Ensemble learning-based approach for the global minimum variance portfolio

Anh-Tuan Tran

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à École Pratique des Hautes Études

Ensemble learning-based approach for the Global Minimum Variance Portfolio

Soutenue par
Anh-Tuan TRAN
Le 20 Mars 2024

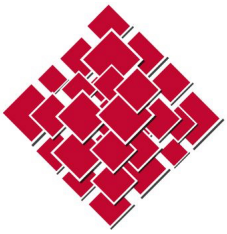
École doctorale n°472
École Pratique des Hautes Études

Spécialité
Informatique, mathématique et applications

Préparée au
EA 4004 - Cognition humaine et artificielle

Composition du jury :

Isis TRUCK University of Paris 8	*Président du jury*
Mhand HIFI University of Picardie Jules Verne	*Rapporteur*
Soufian BEN AMOR University of Versailles Saint-Quentin-en-Yvelines	*Rapporteur*
Vu DUONG Nanyang Technological University	*Examinateur*
Marc BUI Ecole Pratique des Hautes Etudes - PSL	*Directeur de thèse*
Hi-Duc PHAM École centrale d'électronique	*Co-encadrant*



École Pratique des Hautes Études | PSL

Acknowledgements

This work has been partially funded by The Youth Incubator for Science and Technology Programe, under grant number 16/2021/HĐ-KHCNT-VU.

Résumé

Ensemble Learning est une méthode puissante pour améliorer les performances des modèles d'apprentissage automatique en combinant les prédictions de plusieurs modèles de base. L'idée derrière l'apprentissage d'ensemble est qu'en combinant les points forts de différents modèles de base, l'ensemble dans son ensemble peut obtenir de meilleures performances que n'importe quel modèle de base unique. Des études empiriques ont montré que la méthode d'ensemble est particulièrement efficace lorsque les modèles de base sont diversifiés, un exemple réussi étant les arbres de décision aléatoires. En raison de ses avantages, Ensemble Learning est largement utilisé dans diverses applications, notamment les problèmes de détection de fraude. Plus en détail, les avantages d'Ensemble Learning tiennent à deux points principaux : i) l'ensemble combine les points forts de ses modèles de base, rendant chaque modèle complémentaire l'un de l'autre, et ii) il neutralise le bruit et les valeurs aberrantes parmi tous les modèles de base, réduire leur impact sur les prévisions finales. Dans cette thèse, nous utilisons ces deux idées d'Ensemble Learning pour différentes applications dans l'apprentissage automatique et l'industrie financière. Nos principales contributions dans cette thèse sont triples. Tout d'abord, nous démontrons comment l'apprentissage d'ensemble et les techniques de sous-échantillonnage peuvent être utilisés pour traiter efficacement le scénario difficile du problème de déséquilibre des données dans le domaine de l'apprentissage automatique, en particulier dans le cas de mégadonnées extrêmement déséquilibrées. Deuxièmement, nous proposons de manière appropriée l'utilisation de la validation croisée des séries chronologiques et de l'apprentissage d'ensemble pour résoudre un problème de sélection d'estimateurs de matrice de covariance dans le commerce quantitatif. Enfin, nous montrons comment l'apprentissage d'ensemble peut être utilisé pour réduire l'impact des valeurs aberrantes dans les estimations de la matrice de covariance, augmentant ainsi la stabilité des portefeuilles. Dans l'ensemble, nos recherches mettent en évidence le potentiel de l'apprentissage d'ensemble pour améliorer les performances de diverses applications dans le domaine de l'apprentissage automatique et de la finance.

Mots clés : apprentissage d'ensemble, sous-échantillonnage, validation croisée, estimation de matrice de covariance

Abstract

Ensemble Learning is a powerful method for improving the performance of machine learning models by combining the predictions of multiple base models. The idea behind ensemble learning is that by combining the strengths of different base models, the ensemble as a whole can achieve better performance than any single base model. Empirical studies have shown that the ensemble method is particularly effective when the base models are diversified, one successful example is random decision trees. Because of its advantages, Ensemble Learning is widely used in various applications, including fraud detection problems. In more detail, the advantages of Ensemble Learning are due to two main points: i) the ensemble combines the strengths of its base models, making each model complementary to one another, and ii) it neutralizes the noise and outliers among all base models, reducing their impact on the final predictions. In this thesis, we use these two ideas of Ensemble Learning for different applications in the machine learning and the finance industry. Our main contributions in this thesis are threefold. Firstly, we demonstrate how ensemble learning and undersampling techniques can be used to efficiently deal with the hard scenario of imbalance data problem in the machine learning field, particularly in the case of extremely imbalance big data. Secondly, we propose appropriately the use of time-series cross-validation and ensemble learning to resolve a covariance matrix estimator selection problem in quantitative trading. Lastly, we show how ensemble learning can be used to reduce the impact of outliers in covariance matrix estimations, thereby increasing the stability of portfolios. Overall, our research highlights the potential of ensemble learning for improving the performance of various applications in the field of machine learning and finance.

Keywords : ensemble learning, undersampling, cross-validation, covariance matrix estimation

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

Moore's Law, first proposed by Gordon Moore, states that the number of transistors on a microchip doubles approximately every two years (Moore et al., 1965; Moore et al., 1975). Recent studies suggesting that the growth rate could potentially be even higher in the future (Mack, 2011; Theis et al., 2017). Figure 1.1 shows that the number of transistors on microchips doubles every two years. This exponential increase in computing power over time has enabled computers to solve a wide range of previously unsolvable problems. One such example is Spam Detection in Emails, which involves classifying unwanted emails in order to improve the user experience. With the sheer volume of emails sent and received daily, it is not feasible for humans to handle this task manually. However, with the help of computers, we can classify emails as spam or non-spam with a high degree of accuracy by detecting common words and patterns in spam emails. Machine Learning algorithms, such as Logistic Regression or Decision Tree, are commonly used for this purpose (Cormack, 2008).

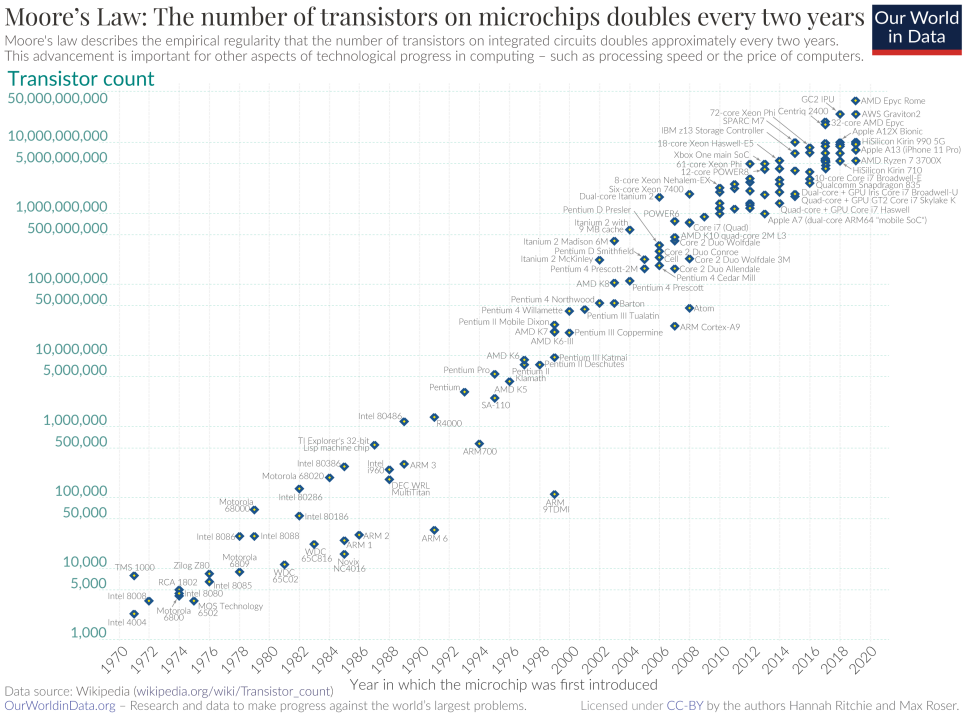


Figure 1.1: A semi-log plot of transistor counts for microprocessors against dates of introduction, nearly doubling every two years. Source: Wikipedia contributors (2022d).

The utilization of Machine Learning techniques in the field of finance has gained significant attention in recent years, due to the exponential increase in computing power and the availability of large amounts of data. Researchers have turned to Machine Learning algorithms to gain insights and make predictions about financial markets. For example, many studies try to predict stock prices in next days by using historical data (Rapach et al., 2013), financial statements (Hirshleifer et al., 2004) or sentiment analysis (Pagolu et al., 2016). However, the financial market is a complex and chaotic system, making accurate prediction a challenging task (Kuhlmann, 2014). There are countless agents who invest in sophisticated multi-layer financial services, even with a small interaction of a single participant, could drastically switch the market regime (Bishop, 2011). Therefore, understanding the characteristics of the financial markets and building predictive models in that chaotic environment are challenging topics.

Despite these challenges, the field of finance continues to explore the use of Machine Learning techniques to gain insights and make predictions about the financial markets. One of the key advantages of Machine Learning in finance is its ability to process large amounts of data and identify patterns that may not be immediately apparent to human analysts. Furthermore, Machine Learning can also be used to improve the efficiency of financial decision-making by automating certain tasks and reducing the need for human intervention. However, there are also limitations to be considered, such as the risk of overfitting and the interpretability of Machine Learning models. Additionally, Machine Learning models can only make predictions based on the patterns they have learned from the data and may not account for unexpected events or changes in market conditions. In order to fully benefit from Machine Learning in

finance, it is essential to have a well-designed and robust approach for the financial machine learning problems.

Applying Machine Learning to Finance is a new field and it has gained significant attention in recent years. It is also known as Financial Machine Learning or Financial Data Science. The Financial Machine Learning field aims to extract order from chaotic environments under the perspective of data. One recent example in this field is industry classification, which involves grouping companies into various sectors based on different criteria, such as production. However, traditional standard classification systems, which have been developed since 1937, may not accurately classify the large and diverse companies present in today's markets. To address this issue, Bonne et al. (2022) have proposed a data-driven industry peer grouping system that clusters similar companies at different levels of granularity by using artificial intelligence to extract features from various data sources and learn relationships. It can identify companies that are similar in terms of their risk-return profile. While these data-driven industry classification systems may not become standard, they have broad applications in finance and economics, as they address the limitations of traditional industry classification systems.

A similar approach to finance is an Econophysic, which is an interdisciplinary research field to solve problems in finance and economics by utilizing theories and methods initially developed from physics. The term Econophysics was started in the 1990s , as physicists recognized the similarities between economics and physics and the availability of large amounts of financial data in the 1980s. They observed that there are many shared characteristics between economics and physics. One example is a Random Matrix Theory, which is used to identify noise in financial correlation matrices and improve their applications. Olivier Ledoit et al. (2012) and Olivier Ledoit et al. (2015) applied an original concept of Random Matrix Theory to deal with noise in large-dimensional covariance matrices and proposed non-linear shrinkage covariance matrix estimations, which have been applied in portfolio optimization. These estimations have demonstrated significant improvements in portfolio performances, and there is potential for further advancements through other approaches.

Portfolio management is a crucial field within finance which includes various subtopics such as asset allocation, portfolio construction, portfolio optimization, risk management, performance measurement and backtesting methodologies. It is also one of the main focuses of Financial Machine Learning, and it is greatly benefited from the advancement of Artificial Intelligence research. In this thesis, one of our research topic focuses on applications of portfolio optimization, particularly a Global Minimum Variance Portfolio which uses only the covariance matrix to estimate a portfolio with the lowest variance. By applying approaches of statistics and machine learning, such as undersampling techniques and ensemble learning, our research demonstrates the effectiveness of these methods in significantly improving portfolio optimization. Our research highlights the potential for further advancements in this area through the application of other advanced methods.

In this thesis, we use two types of data. The first part contains eleven imbalanced datasets that were downloaded from the UCI Machine Learning Repository and various online sources. These datasets cover a wide range of characteristics, including variations in size and degree

of imbalance. These datasets include those from the imbalanced-learn library (Lemaitre et al., 2017), a real-world credit card fraud detection dataset (Dal Pozzolo, 2015) and a network intrusion detector KDD-99 dataset (Bay et al., 2000). A second part makes use of historical stock prices from the Ho Chi Minh Stock Exchange (HOSE), an emerging market in Asia. These stock prices are publicly available on the HOSE website and our research focuses on a normal period of the market, spanning from 2011 to the end of 2019. It is important to note that market crises such as the COVID-19 pandemic in 2020 fall outside of the scope of this thesis as they represent a different context and require a separate analysis.

1.2 Scientific challenges and contributions

In this thesis, we explore the use of Machine Learning techniques and their applications to financial problems. The specific topics of our research include:

First of all, we start with our observation in a Fraud Detection problem which tries to classify fraudulent transactions in credit card transactions datasets. Financial institutions, such as banks, lose billions of dollars annually due to credit card fraud, and this issue has been on the rise in recent years. Therefore this is an important for the banking industry to develop effective methods for detecting and preventing fraud. In our previous work, we have observed that an imbalance ratio in this case is abnormally higher than in common cases (T. Tran, 2022). Several studies show their imbalance ratios, which is a ratio of the number of majority class instances to the number of minority class, from 19 to 99. It means for every 100 data points, there are 1 to 5 positive data points. While in a dataset of two-day credit card transactions captured from a European bank in September 2013, there are only 492 fraudulent transactions out of 284807 transactions, and its imbalance ratio is roughly 578. In other problems, it could be higher than 10^6. Most research studies for imbalance problems only deal with an easy case, i.e. low imbalance ratio. Only a few studies mentioned the extreme case in their experiments, however imbalance ratios of their extreme imbalance datasets usually not excess 100. A higher imbalance ratio easily leads to a bigger dataset, it presents a unique challenge for traditional approaches that are designed to handle low-imbalance and small-dataset problems. For example, the credit card transactions dataset above is two-day data, roughly 50 million transactions per year and containing only 0.17% fraudulent transactions. To the best of our knowledge, this gap between the extreme imbalance problem and big dataset problem has not been addressed in previous research. A challenge in this gap is that we not only deal with the problematic imbalance data but also process the large-scale dataset in order to classify the minority class accurately and quickly. In other words, we need an approach more efficiently for this new problem.

In our study, we consider the issue of extreme class imbalance, characterized by an imbalance ratio of 100 or more, where there is only one positive data point for every 100 negative data points or more. Instead of building more complex models, we propose the use of an undersampling technique to reduce the complexity of the problem, which is the imbalance ratio in this case. This is achieved by building models using lower-imbalance subsets and then combining these models through ensemble learning techniques such as Voting Ensemble. Our approach involves

the use of a small number of subsets, such as 3, 5 or 10, to reduce training time, and the sampling process is done without replacement to further reduce the size of the training set. This sampling approach in the context of big data does not affect the accuracy of the models. We conduct experiments on a variety of datasets, from minuscule to exceedingly large datasets and imbalance ratios from minimal to exceptionally high. The highest imbalance ratio in our study is nearly 10^5 and the dataset contains nearly 5 million data points. Our results demonstrate that our approach is effective not only in the extreme-imbalance big-data cases but also in less challenging scenarios. We presented this new gap of the extreme imbalance problem in big data and our experimental results in a paper titled "K-Segments Under Bagging approach: An experimental Study on Extremely Imbalanced Data Classification" (T. Tran, L. Tran, et al., 2019).

Secondly, we address the problem of model selection in the context of portfolio optimization, which is a key topic in finance. Modern Portfolio Theory, as proposed by Markowitz (H. Markowitz, 1952; H. M. Markowitz, 1968), provides a framework for investors to optimize their portfolios based on the returns and covariance of assets, known as the Mean-Variance portfolio. However, the returns of assets are known to be difficult to predict and more volatile than the covariance matrix. As a result, Markowitz's portfolio performs poorly during market crash periods, such as the 2007-2008 Financial Crisis. Among all feasible portfolios on the efficient frontier of Modern Portfolio Theory, there is a special case with minimum variance, known as the Global Minimum Variance portfolio. This portfolio uses only the covariance matrix to optimize the portfolio, with the goal of minimizing risk. While the goal of this portfolio is not to maximize profit, it has been shown to have higher long-term returns than the Mean-Variance portfolio over a long-term investment horizon. This phenomenon is known as the low volatility anomaly. There are studies that attempt to explain this anomaly, but it falls outside the scope of this thesis. A standard covariance matrix estimation in Markowitz portfolios is the sample covariance matrix estimation, which is simple but has many problems. One of the most critical issues is its singularity, particularly in high-dimensional data when the number of assets is greater than the number of observations. As a result, the sample covariance matrix is often not invertible, leading to suboptimal Markowitz portfolios. To address this issue, the shrinkage technique in statistics has been proposed, which combines the sample covariance matrix with another invertible matrix, resulting in a significant improvement in the Global Minimum Variance portfolio. However, various matrices have been proposed for use in shrinkage estimation, such as the identity matrix or a constant-correlation matrix, each with different advantages in different market scenarios. This raises the question of estimation selection for investors, and investors need a mechanism to select the best estimation among various possible solutions for a given set of data and investment objectives.

Under the data perspective, the optimal intensity of shrinkage estimations are in-sample results, i.e. the results on the training data, and those results could be overfitted on the training data or underfit on the future data. With only the in-sample results, the covariance matrix estimations are incomparable. Moreover, to avoid the overfitting problem in covariance matrix estimation, we propose to use the Cross-Validation technique to estimate the portfolios' perfor-

mances on the testing data and then let them vote for their best estimator by Voting Ensemble. With the time series data of stock prices, a typical K-Fold method and shuffling in the Cross-Validation are not appropriate. Therefore we use a rolling Leave-One-Out Cross-Validation without shuffling. Within the scope of our study, we consider three kinds of shrinkage estimators which have been proposed by Ledoit and Wolf such as Shrinkage to identity matrix (Olivier Ledoit et al., 2003), Shrinkage to single-index model (Olivier Ledoit et al., 2004b), Shrinkage to constant correlation model (Olivier Ledoit et al., 2004a). We estimate the performances of those estimators by a Sharpe ratio which indicates a tradeoff between profit and risk of a portfolio. In other words, we assume the main objective of investors is to select an estimator with the highest Sharpe ratio. Testing our approach with those shrinkage estimators on the Vietnam stock market, particularly on the HOSE exchange, from 2013 to the end of 2019 shows that our approach could adapt quickly to the market changes and produce better portfolio performances significantly. It suggests that the investors could follow our selection mechanism with several estimations and their objectives to choose a single best estimation for their portfolios. We described our approach and the results in a paper titled "Voting shrinkage algorithm for Covariance Matrix Estimation and its application to portfolio selection" (T. Tran, N. Nguyen, T. Nguyen, and Mai, 2020).

Thirdly, we we address the issue of outliers in the covariance matrix estimations for the Global Minimum Variance Portfolio (GMVP). The GMVP has only one input parameter, the inverse of the covariance matrix, and thus it requires an invertible and stable covariance matrix. The most common covariance matrix estimation for the GMVP is the sample covariance matrix estimation, which has a critical problem of singularity. When the number of observations is greater than the number of variables, the sample covariance matrix is singular. Then the GVMP is non-optimal and unreliable. At the present time, the number of companies is increasing faster than the number of dates. Therefore, we will easily face with a singular problem in many cases. Various research studies focus on the singular problem, such as the linear shrinkage technique in statistics, which optimally combines the sample covariance matrix with another well-structured matrix. The optimal linear shrinkage estimations show significant improvements for portfolio optimization in the GMVP. This approach goes further with many variants or more complex estimations, such as non-linear shrinkage estimations. From another point of view, some studies argue that the significant improvement of shrinkage estimations is because of combining with another matrix, not because of their optimal combinations. They showed that facultative combinations of the sample covariance matrix and different matrices have the same level of portfolio performance with the optimal shrinkage estimations, at least on the portfolio volatility. However, the sample covariance matrix estimation or even more advanced estimations, such as linear shrinkage estimations, are also sensitive to outliers. The problem of outliers receives less attention than the singularity or robustness of the covariance matrix. However, a robust covariance matrix estimation is not robust if it is sensitive to outliers. Therefore, the research community should focus not only on the singular problem but also on the outliers problem.

We address the problem of outliers in covariance matrix estimations for the Global Minimum Variance Portfolio (GMVP) by utilizing Machine Learning techniques such as undersampling and

ensemble learning. These techniques have been shown to be effective in reducing the impact of outliers on the output of Machine Learning models. Specifically, we use the undersampling technique to sample several smaller subsets from the original dataset and train a single model for each subset. These models are then ensembled into a final model. In the context of GMVP, we consider the covariance matrix estimation as a model and the covariance matrix as its output. Additionally, a common procedure in the GMVP is that the daily stocks return dataset is extracted to a weekly stocks returns dataset by taking the last return values in each week. Then the covariance matrix is estimated from this weekly dataset. In our context, we we propose a modification to the traditional GMVP procedure, where we under-sample the daily stock returns dataset to several weekly datasets by randomly and with replacement taking return values in each week. These weekly datasets are then used to estimate the covariance matrix using any given estimation method, such as the sample covariance matrix estimation or linear shrinkage estimations. These estimated covariance matrices are then ensembled into a final covariance matrix and used in the portfolio optimization process of GMVP. Those weekly datasets have the same size as each other and the same as the weekly dataset in the traditional procedure, similarly to those covariance matrices. Our approach reduces the fluctuation of the covariance matrix by manipulating the input data, while other approaches such as shrinkage estimations operate at the model level. Empirical results on the Vietnam stock market from 2013 to the end of 2019 show that applying our approach to the sample covariance matrix estimation improves it to the level of the shrinkage estimations. Similarly, on the Shrinkage to single-index model, it shows that this shrinkage estimation is also impacted by the outliers and other shrinkage estimations are not sensitive to the outliers. We presented those results in a paper titled "k-Covariance: An Approach of Ensemble Covariance Estimation and Undersampling to Stabilize the Covariance Matrix in the Global Minimum Variance Portfolio" (T. Tran, N. Nguyen, and T. Nguyen, 2022).

1.3 Outlines

This thesis consists six chapters, as follows:

- Chapter 1: We introduce the field of Financial Machine Learning, which is an interdisciplinary area that encompasses both machine learning and finance. We then outline the scientific challenges and the contributions of our research to both machine learning and financial problems.

- Chapter 2: In this chapter, we provide a background on the topic of Financial Machine Learning, covering both machine learning and finance concepts. Specifically, we present formal definitions, terms and formulas that are essential for understanding the later chapters.

- Chapter 3: In this chapter, we explore the new challenge of dealing with extreme imbalance data and big data, and propose an approach using undersampling and ensemble learning to effectively address this issue.

- Chapter 4: In this chapter, we examine the problem of selecting estimators for covariance matrix estimation in quantitative investment, and propose an approach using cross-validation and ensemble learning to predict the best estimator for portfolio optimization.
- Chapter 5: In this chapter, we investigate the impact of outliers on covariance matrix estimation and the resulting portfolio performance. We propose an approach using under-sampling and ensemble learning to reduce the impact of outliers and stabilize the covariance matrix.
- Chapter 6: Finally, we summarize our contributions and discuss the limitations of our research. Based on these limitations, we suggest potential directions for future work.

References

Bay, Stephen D, Dennis Kibler, Michael J Pazzani, and Padhraic Smyth (2000). "The UCI KDD archive of large data sets for data mining research and experimentation". In: *ACM SIGKDD explorations newsletter* 2.2, pp. 81–85 (cit. on p. 4).

Bishop, Robert C (2011). "Metaphysical and epistemological issues in complex systems". In: *Philosophy of complex systems*. Elsevier, pp. 105–136 (cit. on p. 2).

Bonne, George, Andrew W Lo, Abilash Prabhakaran, Kien Wei Siah, Manish Singh, Xinxin Wang, et al. (2022). "An Artificial Intelligence-Based Industry Peer Grouping System". In: *The Journal of Financial Data Science* 4.2, pp. 9–36 (cit. on p. 3).

Cormack, Gordon V (2008). "Email spam filtering: A systematic review". In: (cit. on p. 1).

Dal Pozzolo, Andrea (2015). "Adaptive machine learning for credit card fraud detection". In: (cit. on p. 4).

Hirshleifer, David, Kewei Hou, Siew Hong Teoh, and Yinglei Zhang (2004). "Do investors overvalue firms with bloated balance sheets?" In: *Journal of Accounting and Economics* 38, pp. 297–331 (cit. on p. 2).

Kuhlmann, Meinard (2014). "Explaining financial markets in terms of complex systems". In: *Philosophy of Science* 81.5, pp. 1117–1130 (cit. on p. 2).

Ledoit, Olivier and Michael Wolf (2003). "Improved estimation of the covariance matrix of stock returns with an application to portfolio selection". In: *Journal of empirical finance* 10.5, pp. 603–621 (cit. on pp. viii, 6, 28–30, 32, 33, 89–91, 103, 105, 137, 138).

— (2004a). "A well-conditioned estimator for large-dimensional covariance matrices". In: *Journal of multivariate analysis* 88.2, pp. 365–411 (cit. on pp. viii, 6, 28, 32, 89, 91, 103, 105, 137).

— (2004b). "Honey, I shrunk the sample covariance matrix". In: *The Journal of Portfolio Management* 30.4, pp. 110–119 (cit. on pp. 6, 34, 89, 91, 103, 106, 137).

— (2012). "Nonlinear shrinkage estimation of large-dimensional covariance matrices". In: *The Annals of Statistics* 40.2, pp. 1024–1060 (cit. on pp. 3, 101, 142).

— (2015). "Spectrum estimation: A unified framework for covariance matrix estimation and PCA in large dimensions". In: *Journal of Multivariate Analysis* 139, pp. 360–384 (cit. on pp. 3, 101, 142).

Lemaitre, Guillaume, Fernando Nogueira, and Christos K Aridas (2017). "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning". In: *The Journal of Machine Learning Research* 18.1, pp. 559–563 (cit. on p. 4).

Mack, Chris A (2011). "Fifty years of Moore's law". In: *IEEE Transactions on semiconductor manufacturing* 24.2, pp. 202–207 (cit. on p. 1).

Markowitz, Harry (1952). "Portfolio selection". In: *The journal of finance* 7.1, pp. 77–91 (cit. on pp. 5, 22).

Markowitz, Harry M (1968). *Portfolio selection*. Yale university press (cit. on pp. 5, 101, 142).

Moore, Gordon E et al. (1965). *Cramming more components onto integrated circuits* (cit. on p. 1).

Moore, Gordon E et al. (1975). "Progress in digital integrated electronics". In: *Electron devices meeting*. Vol. 21. Washington, DC, pp. 11–13 (cit. on p. 1).

Pagolu, Venkata Sasank, Kamal Nayan Reddy, Ganapati Panda, and Babita Majhi (2016). "Sentiment analysis of Twitter data for predicting stock market movements". In: *2016 international conference on signal processing, communication, power and embedded system (SCOPES)*. IEEE, pp. 1345–1350 (cit. on p. 2).

Rapach, David and Guofu Zhou (2013). "Forecasting stock returns". In: *Handbook of economic forecasting*. Vol. 2. Elsevier, pp. 328–383 (cit. on p. 2).

Theis, Thomas N and H-S Philip Wong (2017). "The end of moore's law: A new beginning for information technology". In: *Computing in Science & Engineering* 19.2, pp. 41–50 (cit. on p. 1).

Tran, Tuan (2022). "On some studies of Fraud Detection Pipeline and related issues from the scope of Ensemble Learning and Graph-based Learning". In: *arXiv preprint arXiv:2205.04626* (cit. on p. 4).

Tran, Tuan, Nhat Nguyen, and Trung Nguyen (2022). "k-Covariance: An Approach of Ensemble Covariance Estimation and Undersampling to Stabilize the Covariance Matrix in the Global Minimum Variance Portfolio". In: *Applied Sciences* 12.13, p. 6403 (cit. on p. 7).

Tran, Tuan, Nhat Nguyen, Trung Nguyen, and An Mai (2020). "Voting shrinkage algorithm for Covariance Matrix Estimation and its application to portfolio selection". In: *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*. IEEE, pp. 1–6 (cit. on pp. 6, 101, 142).

Tran, Tuan, Loc Tran, and An Mai (2019). "K-Segments Under Bagging approach: An experimental Study on Extremely Imbalanced Data Classification". In: *2019 19th International Symposium on Communications and Information Technologies (ISCIT)*. IEEE, pp. 492–495 (cit. on pp. 5, 104, 144).

Wikipedia contributors (2022d). *Moore's law — Wikipedia, The Free Encyclopedia*. [Online; accessed 23-April-2022] (cit. on pp. vii, 2).

Chapter 2

Preliminaries

Objectives

In this chapter, we will provide a comprehensive background on the fundamental concepts of both Machine Learning and Quantitative Finance. This will serve as a foundation for the later chapters.

Contents

2.1 Machine Learning

Artificial Intelligence enables computers to perform tasks that normally require human intelligence. And a subfield of Artificial Intelligence that allows computers to learn from data with little human involvement is called Machine Learning (Samuel, 2000; Koza et al., 1996). ML has many applications in domains where it is difficult or impractical to write explicit rules, such as email filtering or recommendation systems. Machine Learning also overlaps with computational statistics, which uses statistical methods to discover patterns in data and build predictive models. Machine Learning algorithms can be formally defined as follows, according to Mitchell (Mitchell et al., 1997):

> "A computer program learns from experience E with respect to a class of tasks T and a performance measure P if its performance on tasks in T, measured by P, improves with experience E."

Machine Learning methods can be grouped into three main categories based on the availability of the "label" or the desired output in the data used by the learning algorithm (Russell et al., 1995):

- Supervised learning: the data have input features and their corresponding output values as labels. The supervised learning algorithm aims to learn a general rule that maps the input features to the labels. Supervised learning algorithms can be of various types, such as regression, classification, active learning or similarity learning.
- Unsupervised learning: the data does not have any output values or labels. In this case, the learning algorithms explore and discover hidden structures in the data. If some output values are missing in the data, it is called semi-supervised learning. If the output values are limited, noisy or imprecise, it is called weakly supervised learning instead of semi-supervised learning.
- Reinforcement learning: there is no data or labels in this case. Instead, it interacts with a dynamic environment to achieve a pre-defined goal. Through the interactions, the learning system observes the environment as its input and receives feedback from the environment as its label. The reinforcement learning algorithm tries to maximize the rewards from the environment for its goal.

The most common task in machine learning is supervised learning, where we learn a function that maps input data to output labels (Mohri et al., 2012). A supervised-learning function $g : X \to Y$ maps an input space X (matrix of feature vectors) to an output space Y (matrix of labels) for a set of N data points $(x_1, y_1), \ldots, (x_N, y_N)$, where each x_i is a feature vector and each y_i is its corresponding label. The function g can either belong to a function class G, or it can be expressed as a scoring function $f : X \times Y \to \mathbb{R}$ that gives the highest score to the output label y: $g(x) = \arg\max_y f(x, y)$. For instance, a possible model for conditional probability is $g(x) = P(y \mid x)$, which can be exemplified by logistic regression (Walker et al., 1967; Cox, 1958).

And an example for the function f can be a joint probability model $f(x, y) = P(x, y)$ such as naive Bayes(Russell et al., 1995).

To evaluate the performance of the learned function, we use a loss function $L : Y \times Y \to \mathbb{R}$ to measure how well the learned function performs. The learned function g has a risk function $R(g)$ that represents its expected loss. The risk can be estimated by Equation 2.1.

$$\widehat{R}(g) = \frac{1}{N} \sum_{i} L(y_i, g(x_i)) \tag{2.1}$$

2.1.1 Decision tree learning

Decision tree learning is a popular supervised learning method in Machine Learning because of its simplicity and interpretability (Rokach and Maimon, 2014). The goal is to create a model that estimates values based on a hierarchical structure of choices. For example, Figure 2.1 shows a decision tree that predicts the survival of passengers on the Titanic ship based on three attributes (Wikipedia, 2017).

A decision tree learner is a supervised learning method that can divide the input data into smaller groups based on a certain criterion. This splitting process is done recursively from top to bottom until all or most of the data points belong to a specific class label. In data mining, decision trees can be used for two main purposes:

- Classification tree: a technique that predicts discrete classes for the input data,
- Regression tree: a technique that predicts continuous values for the input data.

Both types of techniques are also known as Classification and Regression Tree (CART) analysis (Breiman et al., 1984). Some of the well-known algorithms for decision tree learning are:

- ID3 (Iterative Dichotomiser 3) (J. Ross Quinlan, 1986),
- C4.5: an improvement of ID3 algorithm (J Ross Quinlan, 2014).

Decision tree algorithms use various criteria to decide which feature is the most suitable for dividing the data into smaller groups. These criteria measure how similar the outcomes are within each group. A decision tree learning algorithm can use various criteria to split the data, such as Entropy, Information Gain and Gini.

2.1.2 Ensemble learning

An ensemble learning is a machine learning approach that improves accuracy by combining different learning algorithms is called ensemble learning (Opitz et al., 1999; Rokach, 2010). Using a single base learner, ensemble methods produce various models. Three most popular kinds of ensemble learning are bagging, boosting, and stacking.

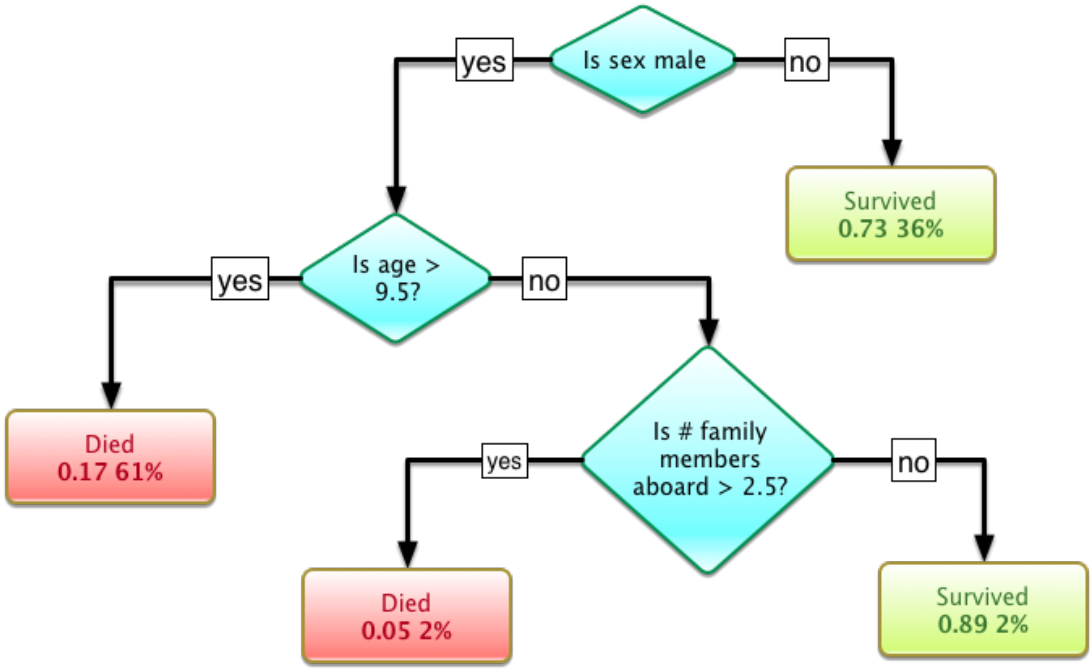


Figure 2.1: One example of decision tree that estimates who lived or died among those on board the Titanic vessel. This is an example with only three attributes such as age, gender and a number of family members. Summarizing: a passenger had a high chance of survival if i) the passenger were female or ii) the passenger were male at least 9.5-year old with strictly greater than 3 siblings. Source: Wikipedia contributors (2017).

Bagging Bagging (see Figure 2.2), also called bootstrap aggregating, is an ensemble technique that combines multiple learnings by their equal weight votes. It trains each model from a subset that is randomly drawn with replacement to lower the model variance. One of the most popular applications of bagging is the Random Forest method that employs decision trees with random features as fundamental learners.

Boosting Boosting is a way of enhancing the performance of machine learning models by using a set of base learners that are trained sequentially on weighted versions of the training dataset (see Figure 2.3). Boosting aims to correct the errors made by previous learners by giving more weight to misclassified data points. Some cases may require boosting to achieve higher accuracy than bagging, but the boosting models tend to fit the training data too closely. Adaboost (Freund, R. Schapire, et al., 1999) is the most common algorithm for boosting.

Stacking Stacking is a way of improving the accuracy of machine learning models by using another model to learn how to best combine the outputs of different base models (see Figure 2.4). These base models are built independently on the given training data, and then a single meta-model is built on their predictions. The meta-model can be any type of model, but often logistic regression or linear regression are used. Stacking can potentially represent any ensemble technique, depending on the choice of base and meta-models.

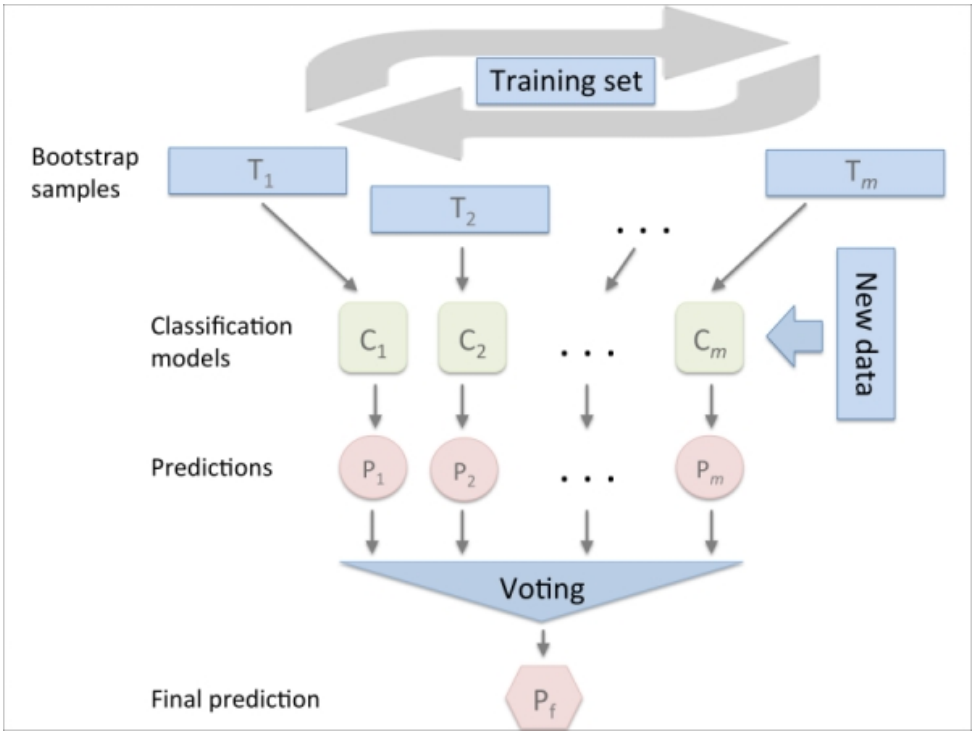


Figure 2.2: Illustration of bagging ensemble (also known as bootstrap aggregating). It draws bootstrap samples $(T_i)_{i=1,\ldots,m}$ randomly with replacement from an original dataset. Each sample is used to train a separate model, it could be regression or classification. Then prediction outputs of those models $(P_i)_{i=1,\ldots,m}$ are ensembled by Majority Voting for classification tasks or by Weighted Average for regression tasks. Source: Raschka (2015)

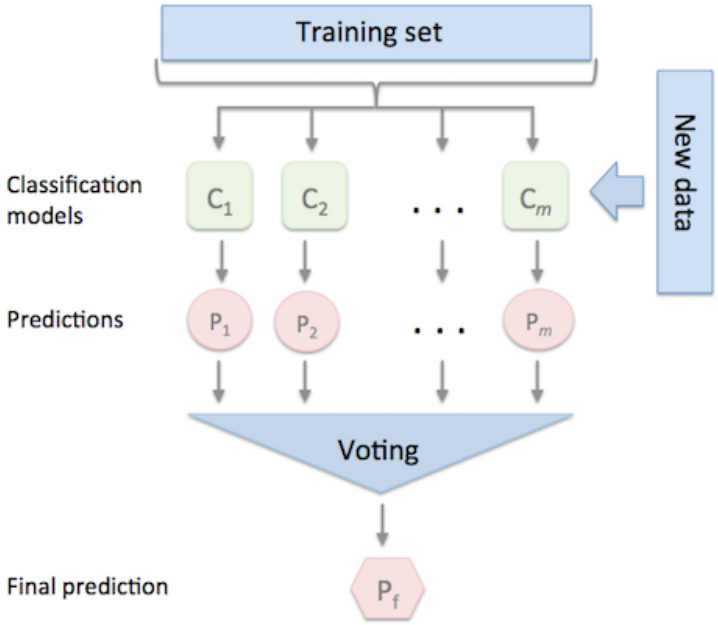


Figure 2.3: Illustration of boosting ensemble. It is an ensemble learning method that combine several weak learners into one strong learner. A sample of data is carefully selected, and is used to train a model. Sequentially, a next sample is selected to improve the performance of previous models. Source: (Raschka, 2015)

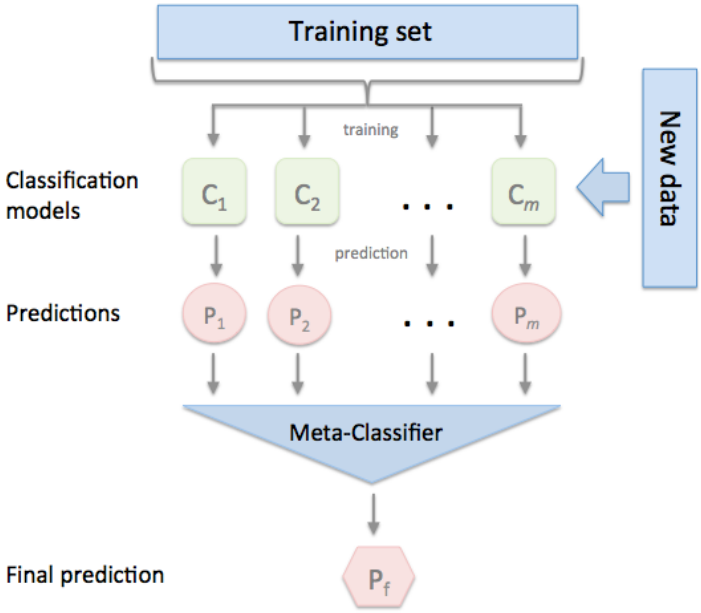


Figure 2.4: Stacking ensemble. Source: (Raschka, 2015)

2.1.3 Random Forest

Random Forest (Breiman, 2001) is a type of ensemble learning that uses many trees, such as decision trees, for classification or regression tasks. It combines the outputs of these trees by taking the most common class for classification task or the average output for regression task. There are two components in the Random Forest model:

1. Base learner: there are several trees in the forest but they are from a same base learner, which is a weak tree-based model with high variability,

2. Ensemble learning: it combines the outputs of multiple base models to produce a final prediction.

Tree-based methods tend to fit too closely to their training sets, meaning that they have low error but high instability. To overcome this problem, Random Forest combines the predictions of many trees that are trained on random samples from the dataset. This way, it reduces the variance and improves the generalization performance of the model. The algorithm can be summarized as follows: given a dataset of N datapoints $X = x_1, \ldots, x_N$ and output values $Y = y_1, \ldots, y_N$ respectively. For B times, randomly choose a sample from X and train a tree on this sample, then aggregate the predictions of all trees (see Algorithm 1).

One way to combine the predictions from different trees in Random Forest is to use an average function for regression tasks or a majority vote for classification tasks, as suggested by the creators of Random Forest (Breiman, 2001). However, other combination functions are also possible. The advantage of bootstrapping is the model's variance decreases without affecting its bias. We can measure how confident the model's prediction is by computing the standard deviation of the predictions from each individual tree. (see Equation 2.2).

Algorithm 1 Tree bagging

Input: X, Y, B
Output: A forest with B trees f
1: **for** $b = 1 : B$ **do**
2: A subset of (X, Y) with N data points is (X_b, Y_b)
3: Build a tree-based model f_b for classification or regression using the sample X_b, Y_b
4: **end for**

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B}(f_b(x) - \hat{f})}{B-1}} \tag{2.2}$$

The Random Forest algorithm is a variant of the bagging method that modifies the decision trees to use a random sample of features at each split. This reduces the correlation among the trees and improves the generalization performance. The dataset has p features and the number of features used for each split is usually $\sqrt{p}$ for classification and $p/3$ for regression, as suggested by the original authors (Friedman et al., 2001).

2.1.4 Evaluation metrics

In this section, we describe some most common evaluation metrics in Machine Learning that are used to measure how well an algorithm performed the task of classification.

Accuracy

Accuracy is a statistical measure that shows how close a single measurement is to the true or accepted value of a quantity (BiPM et al., 2008). In machine learning, accuracy is often used to evaluate the performance of a classifier or a predictor. It is calculated as the ratio of correct predictions to the total number of predictions made by the model. A higher accuracy indicates that the model can correctly identify or classify the input data, while a lower accuracy suggests that the model makes more errors or misclassifications.

Suppose we have 2 sets with N datapoints: the desired values $Y = y_1, \ldots, y_n$ and our predicted values $\hat{Y} = \hat{y_1}, \ldots, \hat{y_n}$, then we can compute accuracy as follows:

$$\text{accuracy} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{y_i = \hat{y_i}} \tag{2.3}$$

Precision and recall

Two common metrics in binary classification, information retrieval and pattern recognition are recall and precision. Recall (also referred to as sensitivity) measures the proportion of relevant data points that are retrieved out of the total number of relevant data points, while precision (also known as positive predictive value) measures the proportion of relevant data points among the given data points (see Figure 2.5).

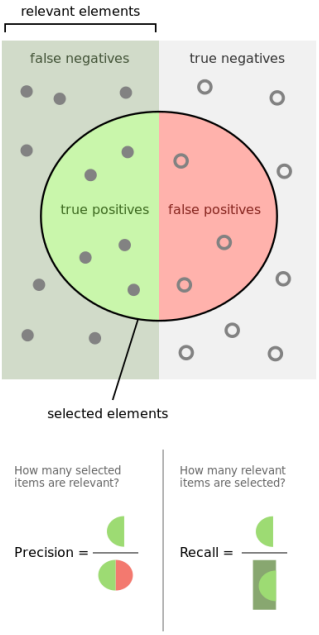


Figure 2.5: Precision and recall. Source: Wikipedia (2018a).

Perry et al. (1955) introduced precision and recall in the context of information retrieval as two sets: one containing the retrieved elements and another containing the relevant elements. The ratio of relevant documents that are retrieved for a query is called precision:

$$\text{precision} = \frac{|\text{ elements that are relevant and retrieved }|}{|\text{ elements that are retrieved }|}. \tag{2.4}$$

The recall measures how many of the relevant elements are retrieved successfully:

$$\text{recall} = \frac{|\text{ elements that are relevant and retrieved }|}{|\text{ elements that are relevant }|}. \tag{2.5}$$

The precision and recall metrics are based on four terms in classification tasks: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). According to Olson et al. (2008), precision is defined as the ratio of TP to TP plus FP, which means how many of the predicted positive cases are actually positive. Recall is defined as the ratio of TP to TP plus FN, which means how many of the actual positive cases are correctly predicted. They are calculated as follows:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.6}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.7}$$

Confusion matrix

A confusion matrix is a tool that helps to evaluate and validate a classification model. It is also known as an error matrix. It shows the number of data points that belong to a certain class (true class) and the number of data points that are predicted to belong to that class (predicted class). The name confusion matrix comes from the fact that it makes it easy to identify where the model is making mistakes between classes (see Figure 2.6).

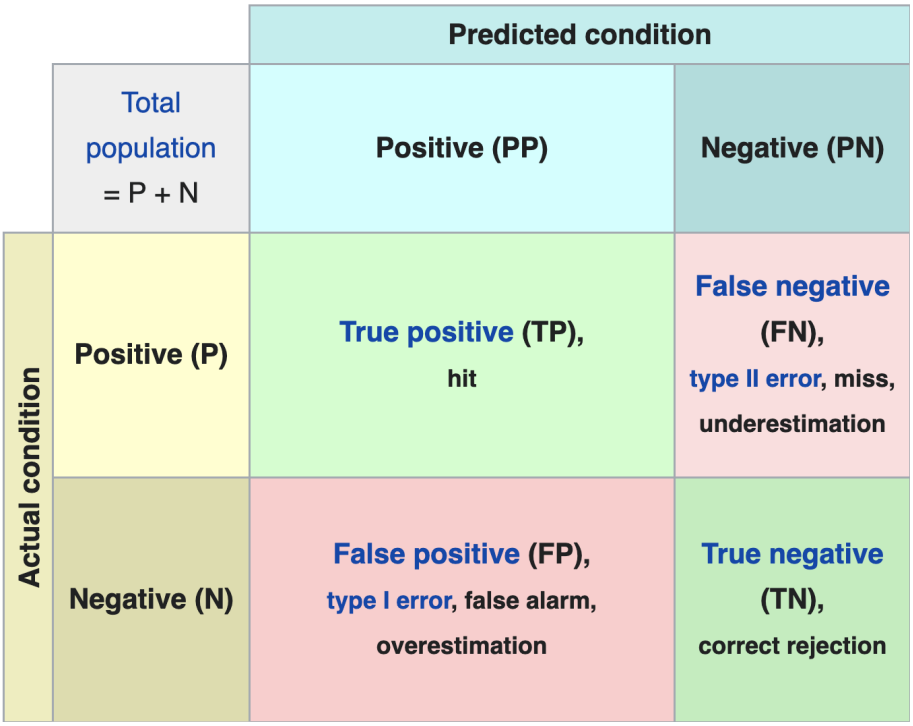


Figure 2.6: A confusion matrix is a special kind of contingency table, also known as an error matrix. It is used to visualize the performance of an Machine Learning algorithm. Source: Wikipedia contributors (2022a).

Receiver operating characteristic (ROC)

A ROC curve, or receiver operating characteristic curve, is a graphical plot that illustrates how well a binary classifier system can distinguish between two classes at various threshold levels. ROC analysis helps to choose a suitable threshold for models by plotting the true positive rate (TPR) versus the false positive rate (FPR). An example of an ROC curve plot for three predictors of peptide cleavage in the proteasome is given in Figure 2.7.

Area under the ROC curve (AUC)

A common way to summarize the performance of a binary classifier system at different threshold levels is to plot the ROC curve, which is a graphical representation of how well the system discriminates between classes. The ROC curve covers a proportion of the unit square, which is called the area under the curve (AUC). The AUC value can vary from 0 to 1, but any realistic

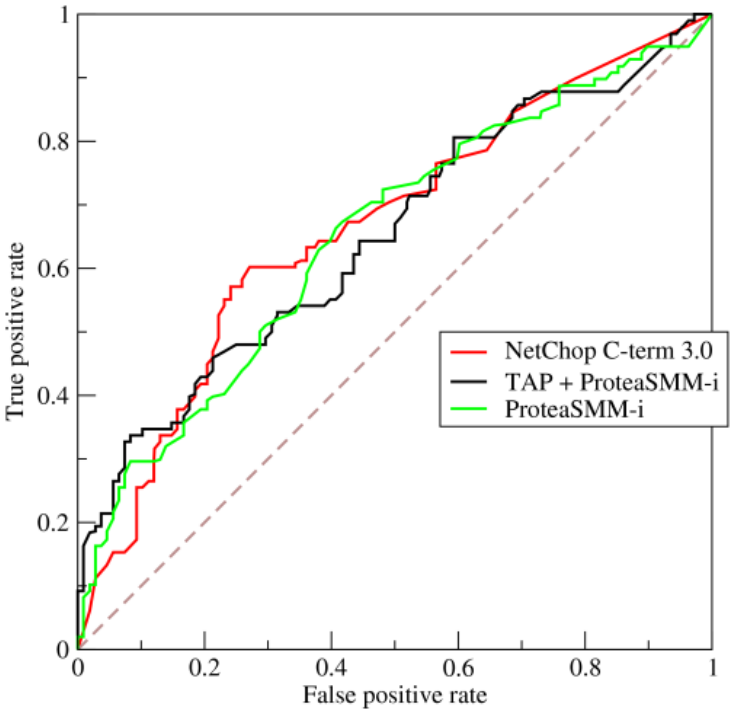


Figure 2.7: A Receiver Operating Characteristic (ROC) curve was utilized to evaluate the performance of three predictors in determining peptide cleavage within the proteasome. Source: Wikipedia (2018b).

classifier should have an AUC above 0.5, which is the baseline performance for random guessing. The AUC metric indicates how close the ROC curve is to the point of perfect discrimination.

The AUC metric has been widely adopted in Machine Learning for comparing models (Hanley et al., 1983). However, some recent studies have identified some limitations and drawbacks of using AUC as a classification measure (Hanczar et al., 2010) and some other studies have highlighted significant problems in comparing models based on AUC (Lobo et al., 2008; Hand, 2009).

F-score

The F-score (also called F-measure or F_1 score) is a metric that combines precision and recall by calculating their harmonic mean.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{2.8}$$

For any positive real number β, a more general formula of F_β-score is:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \tag{2.9}$$

Van Rijsbergen (1979) suggests that the F_β could be interpreted as "measures the effectiveness of retrieval with respect to a user who attaches β times as much importance to recall as precision". When the dataset is imbalanced, the F-score metric is a common choice in machine

learning because it does not suffer from the imbalance problem.

2.2 Financial Machine Learning

Machine Learning is making our lives better with various applications in every aspect of the world. In finance, Machine Learning started to change the financial industry with several disruptive technologies, it is also known as fin-tech (financial technology). More particularly in quantitative investment, Machine Learning is changing the way how we invest our capital. In the stochastic stock markets, it is hard to build a predictive model under a high uncertainty environment even with a group of financial experts. But with the help of Machine Learning, we are able to analyze terabytes of data from various sources and extract undiscovered hidden patterns in the data automatically. Therefore, Financial Machine Learning is an important subject for the investment community.

In another point of view, Financial Machine Learning is a gap between Financial Mathematics and standard Machine Learning. The Financial Mathematics use elegant mathematics with multiple strict assumptions to model the formula for the financial world. That world may not exist in a physical sense even if their theorem is true in a logical sense. The Econophysics which we described above is reversed, they re-use existing theorem in the physical sense and apply to the financial world just because those worlds have multiple shared characteristics. On another side, the Machine Learning community applies their algorithms or techniques to financial applications by an inappropriate way. They misuse mathematical concepts or financial assumptions, which makes their models' results overfit and will fail in the real world. In the Financial Machine Learning topic, it aims to apply the Machine Learning properly to solve financial problems with real assumptions of the world. Without an appropriate approach, every Machine Learning models are non-sense even if they significantly outperform others.

There are many examples of inappropriate Machine Learning applications in finance, such as brute-force search. (De Prado, 2018) showed that we could easily discover a (false) trading strategy, i.e. a false positive case, with a significance level of .05. Typically, with just around 20 iterations to find a small subset in a massive dataset which matches with the above (false) strategy. Therefore, they suggest that a study must report how many trials to discover a strategy, and then they proposed an approach to estimate an overfit probability for that strategy. The brute-force search methodology is considered as a scientific fraud and is mentioned in a ethical guidelines of (Wallman, 1993).

In this thesis, we focus on the applications of the Machine Learning in the Modern Portfolio Theory of Markowitz which is one of the main topics in the Quantitative Finance. Particularly, we will show how investors could use the Machine Learning techniques such as ensemble learning, under-sampling or cross-validation to improve Markowitz's portfolios. In this section, we will describe background knowledge of mathematical finance which is necessary for understanding later chapters.

2.2.1 Modern Portfolio Theory

In a 1952 essay, Harry Markowitz introduced a mathematical framework for portfolio selection (H. Markowitz, 1952) and this groundbreaking work changed the whole financial industry, today it is known as Modern Portfolio Theory. The Modern Portfolio Theory extends and formularizes a diversification in investing with two ideas:

- Investing on different financial assets is less risky than on a single asset. For a definition of the risk, Markowitz uses the standard deviation of asset's returns as a proxy of the asset's risk,
- Both risk and return of an asset should not be evaluated by themselves. In a portfolio of different financial assets, they should be assessed by how much that asset contributes to the overall risk and return of the portfolio.

In the Modern Portfolio Theory, Markowitz assumes that all investors are risk aversion which means that between two portfolios with similar expected returns, they will prefer the less risky portfolio. Therefore, if investors expect a higher expected return then they must accept more risk in their portfolios. Consequently, a rational investor will prefer a portfolio with the lowest risk among all portfolios with the same expected returns. The investors assess a risk-expected return profile of a portfolio by two components:

- Portfolio expected return $\mathbb{E}[R_P] = \sum_i w_i \mathbb{E}[R_i]$,
- Portfolio volatility $\sigma_P = \sqrt{\sigma_P^2}$,

where R_P is the return of a portfolio P, R_i is the return of an asset i in the portfolio P, w_i is the weighting of the asset i, σ_P^2 is the variance of portfolio P's return values which is calculated as follows:

$$\begin{aligned}\sigma_P^2 &= \sum_i w_i^2 \sigma_i^2 + \sum_i \sum_{j \neq i} w_i w_j \sigma_i \sigma_j \rho_{i,j} \\ &= \sum_i \sum_j w_i w_j \sigma_i \sigma_j \rho_{i,j} \\ &= \sum_i \sum_j w_i w_j \sigma_{i,j}\end{aligned} \tag{2.10}$$

where σ_i is the standard deviation of the asset i's returns, $\rho_{i,j}$ is the correlation coefficient of the returns between asset i and asset j and $\sigma_{i,j} = \sigma_i \sigma_j \rho_{i,j}$ is the covariance of the returns between asset i and asset j.

2.2.1.1 Efficient Frontier

With all possible combinations of assets, each of them is a different portfolio, if we plot them in a risk-expected return space then a collection of those portfolios specifies a region in this 2-D

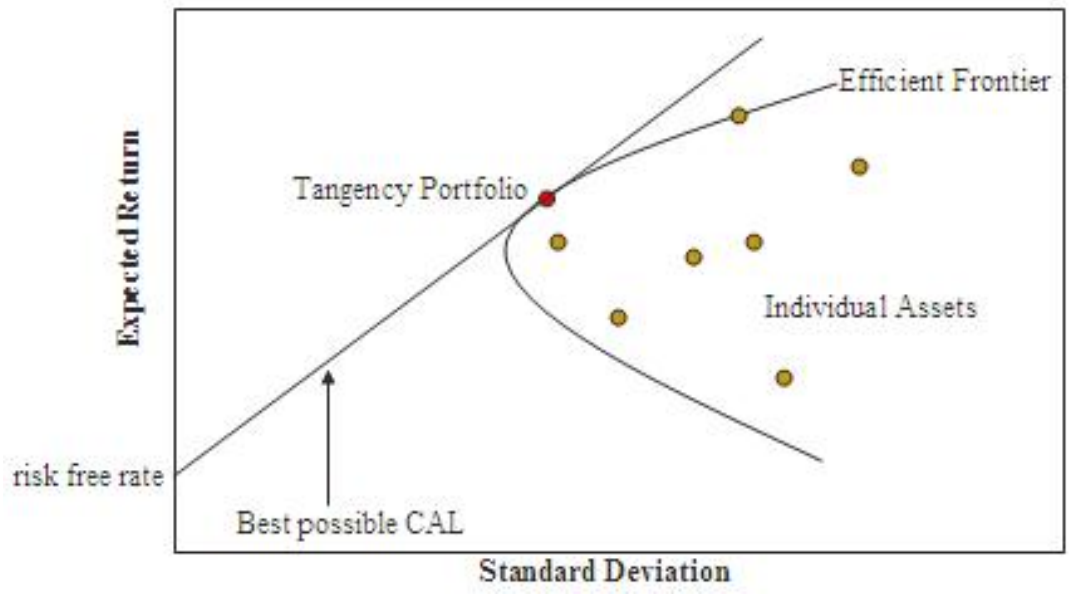


Figure 2.8: Efficient Frontier, also known as "Markowitz bullet". Source: Wikipedia contributors (2022b).

space. In this space, an efficient frontier is the set of portfolios such that with the same level of risk then there is no other portfolio with a higher expected portfolio return. Similarly, given an expected return for a portfolio, there is only one "efficient" portfolio which has lowest risk and that portfolio lies on the efficient frontier curve. The efficient frontier is illustrated in Figure 2.8, a left boundary of the region is parabolic.

2.2.1.2 Mean-Variance Portfolio

The Modern Portfolio Theory, also known as a mean-variance portfolio, which compares the expected return, i.e. mean, with the risk of the portfolio, i.e. variance (or equivalent, standard deviation). All feasible portfolios lying on the efficient frontier curve indicate the "efficient" combinations which offer the best possible expected return for the given risk level.

Given an expected risk $q \in [0, \infty)$, i.e. risk tolerance of an investor, the mean-variance portfolio for a universe of N assets is calculated by minimizing the following equation:

$$\vec{w}^T \mathbf{\Sigma} \vec{w} - q\vec{R}^T \vec{w} \tag{2.11}$$

where:

- $\vec{R}$ is a vector of assets' expected returns,
- $\vec{w} = [w_1, w_2, ..., w_N]$ is a vector of portfolio weights such that $\sum_i^N w_i = 1$,
- Σ is a covariance matrix of the returns of N assets,
- $\vec{w}^T \mathbf{\Sigma} \vec{w}$ is a portfolio variance,
- $\vec{R}^T \vec{w}$ is a portfolio expected return.

Another way to calculate the mean-variance portfolio is by using the investor's expectation of portfolio return $\vec{w}^T\vec{R}$. Given an expected portfolio return p, the mean-variance portfolio is calculated by minimizing the following equation:

$$\vec{w}^T\mathbf{\Sigma}\vec{w} \tag{2.12}$$

subject to

$$\begin{cases} \vec{w}^T\vec{R} = p \\ \vec{w}^T\mathbf{1} = 1 \end{cases} \tag{2.13}$$

A tangent to the upper part of the efficient frontier curve is a special portfolio of Markowitz's mean-variance portfolios, it is named as a tangency portfolio. The tangency portfolio not only is the "efficient" portfolio but also has the highest Sharpe ratio (i.e. a trade-off between portfolio return and portfolio volatility). A most left point in the efficient frontier curve is a portfolio with minimum variance (i.e. portfolio's risk), it is named as global minimum variance portfolio.

2.2.2 Global Minimum Variance Portfolio

Input parameters in Markowitz's mean-variance portfolio are the return, the risk and also the covariance matrix. The calculations for those parameters are based on the assets' expected returns to represent statistical features in the future. However, estimated parameters from the historical data often do not capture the true features due to the massive fluctuation in the chaotic financial markets. Large estimation error leads to an unstable mean-variance portfolio and also degrades significantly the portfolio performances.

Several studies point out the impact of those estimation errors. For example, Michaud (1989) and Chopra et al. (2013) show that mean-variance portfolio optimization is sensitive to the estimation errors, and small changes in the input parameters could lead to large changes in portfolio weights of the optimal mean-variance portfolio. Especially, the estimation error in the expected returns is greater substantially than in the variances and covariances (see Figure 2.10). They concluded that for a low-risk tolerance investor, minimizing the variance of the portfolio is more crucial than looking for the additional expected return. Therefore, a portfolio with lowest risk (i.e. variance) is an important topic for those low-risk tolerance level investors. And that portfolio is a starting point for all other portfolios and also known as the global minimum variance portfolio (see Figure 2.9).

In practice, the mean-variance portfolio has shown a worse performance in the 2008-2009 crisis period while the global minimum variance portfolio showed a stable and better performance in the same period. After that time, the applicability of the mean-variance portfolio is questioned by both the research community and fund managers that why the optimized expected-return portfolio has a lower return than the non-optimized one. In most stock markets, we observe that low-volatility financial assets have higher returns than high-volatility financial assets over the long investment horizon (see Figure 2.11). This market anomaly is also known as the low-volatility anomaly. There are several studies trying to explain this phenomenon but there is no

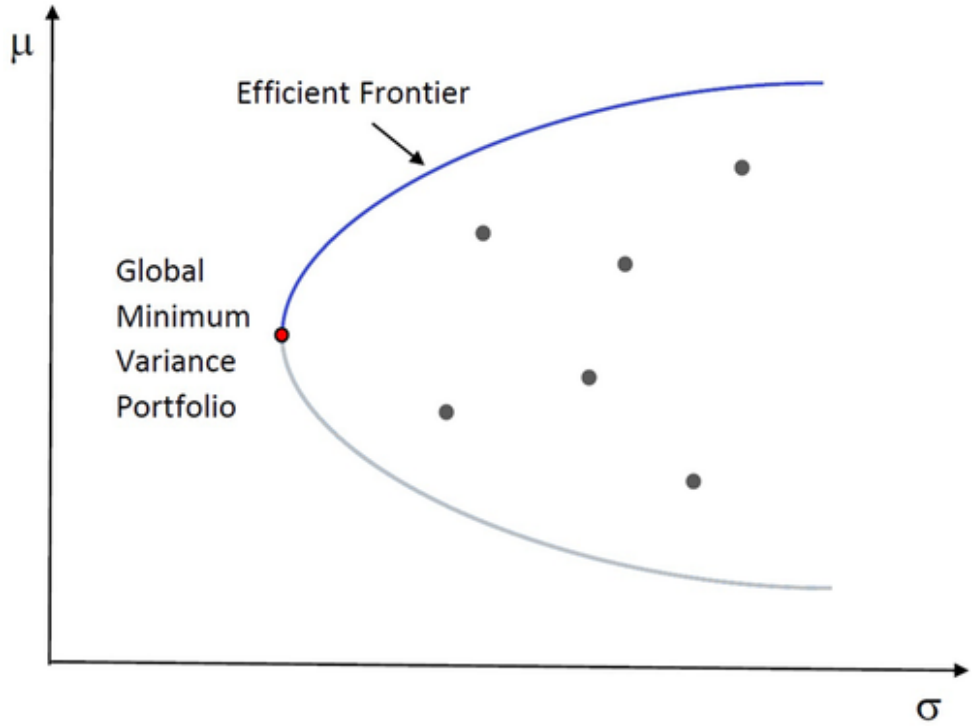


Figure 2.9: The Global Minimum Variance Portfolio is a starting point for all other portfolios in Markowitz's portfolio selection. It is on the Efficient Frontier curve and is the most left point. The y-axis is the portfolio expected return and the x-axis is the portfolio volatility. Source: Golosnoy, Gribisch, et al. (2022).

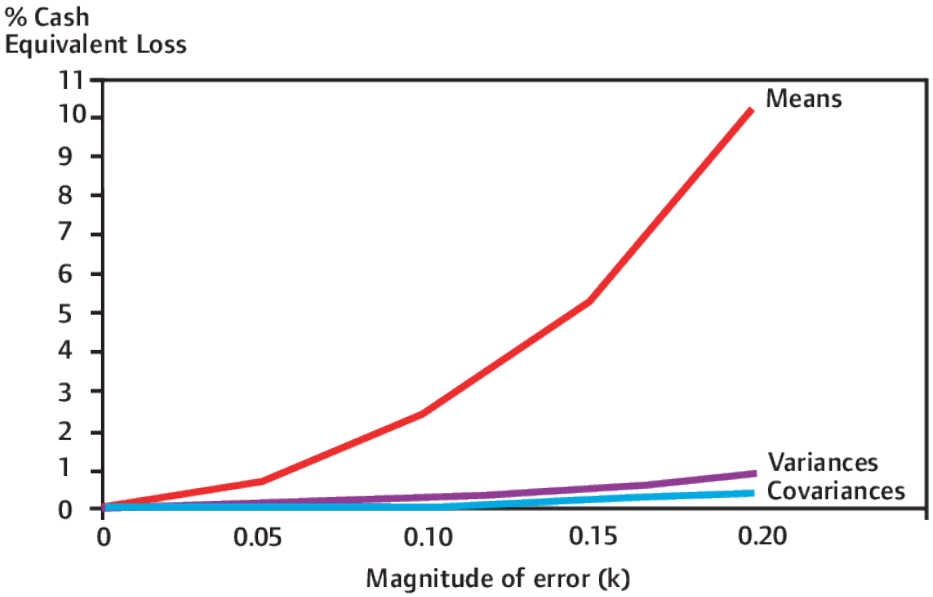


Figure 2.10: Cash loss due to estimation errors in the input parameters of the Markowitz portfolio. The estimation error in the means (expected returns) is higher several times than in the variance or in the covariance. Among these input parameters of the Markowitz portfolio, the covariance estimation has lowest estimation error. Source: Chopra et al. (2013).

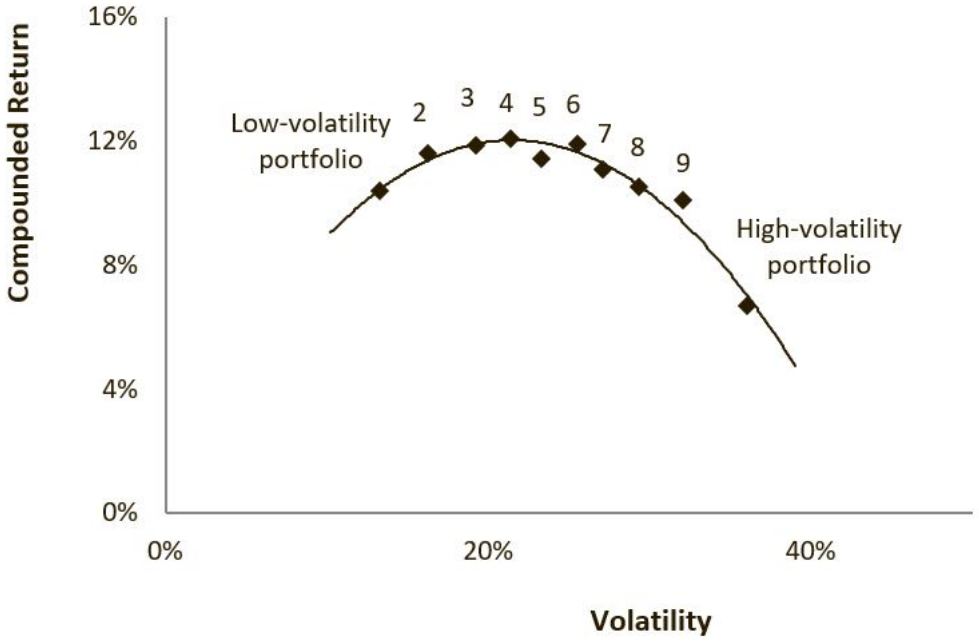


Figure 2.11: Sorting all stocks in the U.S stock market from 1929 to 2020 by their volatilities. Low-volatility stocks surprisingly yield higher returns than high-volatility stocks. This observation of low risk but high return is known as low-volatility anomaly. Source: Wikipedia contributors (2022c).

clear answer yet.

In the global minimum variance portfolio, there is only one parameter: the covariance matrix. Therefore, it does not face with the estimation error problem in the mean. Computing portfolio weights for the global minimum variance portfolio is slightly different from the calculation of the mean-variance portfolio, there is no desired portfolio return and we are only concern about minimizing the portfolio variance. Similarly to the Equation 2.12, the optimization problem for the global minimum variance portfolio is defined as follows:

$$\begin{aligned} \min_{\vec{w}} \quad & \vec{w}^T \Sigma \vec{w} \\ \text{s.t.} \quad & \vec{w}^T \mathbf{1} = 1. \end{aligned} \tag{2.14}$$

This minimization problem could be solved by using the Lagrange form as follows:

$$\mathcal{L}(\vec{w}, \lambda_1) = \frac{1}{2} \vec{w}^T \Sigma \vec{w} - \lambda_1 (\vec{w}^T \mathbf{1} - 1). \tag{2.15}$$

We will derive two first order conditions of the above minimization problem as follows:

$$\frac{\partial \mathcal{L}}{\partial \vec{w}} = \sum \vec{w} - \lambda_1 \mathbf{1} = 0, \tag{2.16}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda_1} = \vec{w}^T \mathbf{1} - 1 = 0. \tag{2.17}$$

From Equation 2.16, we will have:

$$\begin{aligned} \vec{w} &= \lambda_1 \mathbf{1} \Sigma^{-1} \\ \Leftrightarrow \vec{w} \mathbf{1}^T &= \lambda_1 \mathbf{1}^T \mathbf{1} \Sigma^{-1}. \end{aligned} \tag{2.18}$$

From Equation 2.17 and Equation 2.18, we will have:

$$\begin{aligned} 1 &= \lambda_1 \mathbf{1}^T \mathbf{1} \Sigma^{-1} \\ \Leftrightarrow \lambda_1 &= \frac{1}{\mathbf{1}^T \Sigma \mathbf{1}}. \end{aligned} \tag{2.19}$$

Finally, let us substitute Equation 2.19 into Equation 2.18 to get a solution of the optimization problem:

$$\vec{w} = \frac{\Sigma^{-1} \mathbf{1}}{\mathbf{1}^T \Sigma^{-1} \mathbf{1}} \tag{2.20}$$

where Σ is the covariance matrix of assets' returns. However, the true covariance matrix of stocks is unknown and it is usually replaced by the sample covariance matrix $\mathbf{S}$.

Let us denote $\mathbf{X}$ as an $N \times T$ matrix of T observations of N random variables with entries are denoted by x_{it}. In the financial context, $\mathbf{X}$ represents a dataset of T returns on a universe of N assets. Assets' returns are assumed as independent and identically distributed random variables, even though it is not true in the real financial world but it is typically acceptable for mathematical calculations. A sample mean μ and a sample covariance matrix $\mathbf{S}$ are defined as follows:

$$\begin{aligned} \overline{x} &= \frac{1}{T} \mathbf{X} \mathbf{1}, \\ \mathbf{S} &= \frac{1}{T} \mathbf{X} \left(\mathbf{I} - \frac{1}{T} \mathbf{1} \mathbf{1}^\intercal \right) \mathbf{X}^\intercal \end{aligned} \tag{2.21}$$

where $\mathbf{1}$ denotes a vector of ones and $\mathbf{I}$ denotes an identity matrix. A sample average of the returns of asset i is $\overline{x}_{i\cdot} = \frac{1}{T} \sum_{t=1}^{T} x_{it}$.

One critical problem of the sample covariance matrix is a singular problem. In the Equation 2.20, the solution of the global minimum variance portfolio requires an inverse of the covariance matrix. However, in a high-dimensional space, i.e. when $N \geq T$, the sample covariance matrix is not invertible. Equation 2.21 shows that the rank of $\mathbf{S}$ is less than or equals $T - 1$. If N exceeds $T - 1$ then the sample covariance matrix is singular, in other words, there is not enough information in the input data to estimate the covariance matrix. In this case, the inverse of this ill-conditioned covariance matrix will amplify the estimation error into a solution of the portfolio

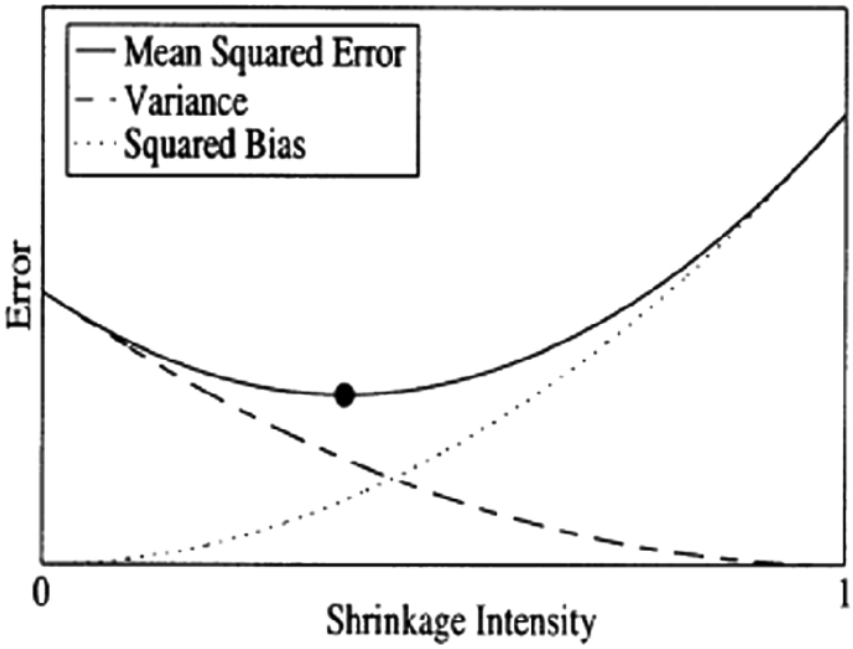


Figure 2.12: The shrinakge estimation is interpreted as a trade-off between bias and variance. The shrinkage intensity is from zero to one. The shrinkage intensity zero means it uses only the sample covariance matrix. And the shrinkage intensity one means it uses only the target matrix. Source: Olivier Ledoit et al. (2004a).

selection, i.e. portfolio weights. In a typical stock market, there are thousands of companies to select from, but usually using maximum of ten years of monthly data, i.e. $N >> 1000$ and $T = 120$ (although $N >> T$, N is assumed to be fixed and finite while $T \to \infty$). Therefore, covariance matrix estimation is a central research topic of the global minimum variance portfolio which focus on building a robust and invertible covariance matrix.

2.2.3 Shrinkage covariance matrix estimations

In order to improve the estimation of the covariance matrix for the portfolio selection in the high-dimensional space, Olivier Ledoit et al. (2003) propose to combine two estimators, the sample covariance matrix and another covariance matrix, by an optimal weighted average. This approach is known as a shrinkage technique in statistics and is used in many domains such as decision theory. A main idea of this shrinkage approach is that the sample covariance matrix is ill-structured and ill-conditioned in a large-dimensional estimation, therefore we could impose its structure to make it be well-conditioned and also invertible. It will shrink an unbiased but has a lot of estimation error (e.g. the sample covariance matrix) to a biased but less estimation error. Thus, the output is an invertible and well-conditioned covariance matrix which is well-defined for estimating the inverse covariance matrix. A beauty of this method is that a proper combination of two extreme estimators could perform better than either of them, this is a fundamental principle in statistics and machine learning is that there is an interior optimum in the trade-off between approximation error and estimation error (bias-variance tradeoff). Figure 2.12 illustrates this trade-off and the shrinkage intensity.

There are three components in the shrinkage estimator, the sample covariance matrix $\mathbf{S}$ with entries $(s_{ij})_{i,j=1,\dots,N}$, a pre-defined target matrix $\mathbf{F}$ with entries $(f_{ij})_{i,j=1,\dots,N}$ and an optimal shrinkage intensity $\alpha \in [0,1]$. The linear shrinkage covariance matrix $\hat{\mathbf{\Sigma}}$ is defined as a linear combination below:

$$\hat{\mathbf{\Sigma}} = \alpha\mathbf{F} + (1-\alpha)\mathbf{S}. \tag{2.22}$$

The pre-defined target matrix $\mathbf{F}$ is an invertible, well-conditioned and biased covariance matrix estimator. Also, it has the same shape as the sample covariance matrix $\mathbf{S}$. It is invertible while the sample covariance matrix $\mathbf{S}$ maybe not, therefore, the shrinkage matrix $\hat{\mathbf{\Sigma}}$ will be invertible definitely. It is well-conditioned while the sample covariance matrix $\mathbf{S}$ is numerically ill-conditioned (even in the case that $\mathbf{S}$ is invertible), therefore, the shrinkage matrix $\hat{\mathbf{\Sigma}}$ could inherit the good conditioning properties of the target matrix. Typically, the target matrix is domain-specific. Different applications have different feasible target matrices, and each of them is used to express its perspective of the true (unobservable) structure of the covariance matrix for a particular domain. The target matrix $\mathbf{F}$ has some weak assumptions such as:

- Asset returns are independent and identically distributed (i.i.d.) through time,
- The number of variables (i.e. assets) is fixed and finite: $N \in \mathbb{N}^+$,
- The number of observations goes to infinity: $T \to \infty$,
- First fourth moments of the asset returns are finite: $\mathbb{E}[|x_{it}x_{jt}x_{kt}x_{lt}|] < \infty \ \forall i,j,k,l,t \in \mathbb{N}^+ | i \leq N, j \leq N, k \leq N, l \leq N, t \leq T$.

The shrinkage intensity α is "optimal" in a sense of similarity between the estimated shrinkage covariance matrix and the true (unobservable) covariance matrix. To measure that similarity, Olivier Ledoit et al. (2003) used a mean squared error which is a common loss function in statistics and machine learning. Using a Frobenius norm of the distance between those two matrices, we have a quadratic loss function below:

$$\begin{aligned} \mathrm{L}(\alpha) &= ||\hat{\mathbf{\Sigma}} - \mathbf{\Sigma}||_F^2 \\ &= ||\alpha\mathbf{F} + (1-\alpha)\mathbf{S} - \mathbf{\Sigma}||_F^2 \end{aligned} \tag{2.23}$$

in which, $||\cdot||_F$ denotes the Frobenius norm of a squared matrix. For example, with a $N \times N$ matrix $\mathbf{A}$ with entries $(a_{ij})_{i,j=1,\dots,N}$, the Frobenius norm is calculated by:

$$||\mathbf{A}||_F = \sqrt{\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}^2} \tag{2.24}$$

Obtaining the optimal shrinkage intensity by minimizing an expected value of the loss function $\mathrm{L}(\alpha)$, we define a risk function $\mathrm{R}(\alpha)$ as follows:

$$\begin{aligned}
\mathrm{R}(\alpha) &= \mathbb{E}\left[\mathrm{L}(\alpha)\right] \\
&= \mathbb{E}\left[||\alpha\mathbf{F} + (1-\alpha)\mathbf{S} - \mathbf{\Sigma}||_F^2\right] \\
&= \sum_{i=1}^{N}\sum_{j=1}^{N} \mathbb{E}\left[\alpha f_{ij} + (1-\alpha)s_{ij} - \sigma_{ij}\right]^2 \\
&= \sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{Var}(\alpha f_{ij} + (1-\alpha)s_{ij}) + \left(\mathbb{E}\left[\alpha f_{ij} + (1-\alpha)s_{ij} - \sigma_{ij}\right]\right)^2 \\
&= \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha^2\mathrm{Var}(f_{ij}) + (1-\alpha)^2\mathrm{Var}(s_{ij}) + 2\alpha(1-\alpha)\mathrm{Cov}(f_{ij}, s_{ij}) + \alpha^2(\phi_{ij} - \sigma_{ij})^2.
\end{aligned} \tag{2.25}$$

Then, the first and second derivatives of the risk function $\mathrm{R}(\alpha)$ are

$$\begin{aligned}
\frac{\partial R}{\partial \alpha} &= 2\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha\mathrm{Var}(f_{ij}) + (1-\alpha)\mathrm{Var}(s_{ij}) + (1-2\alpha)\mathrm{Cov}(f_{ij}, s_{ij}) + \alpha(\phi_{ij} - \sigma_{ij})^2, \\
\frac{\partial^2 R}{\partial \alpha^2} &= 2\sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{Var}(f_{ij} - s_{ij}) + (\phi_{ij} - \sigma_{ij})^2.
\end{aligned} \tag{2.26}$$

We have an optimum value α^* by letting $\dfrac{\partial R}{\partial \alpha} = 0$ and it is verified as a minimum value because $\frac{\partial^2 R}{\partial \alpha^2} > 0 \ \forall \alpha \in \mathbb{R}$.

$$\begin{aligned}
\alpha^* &= \arg\min_{\alpha \in \mathbb{R}} \mathrm{R}(\alpha) \\
&= \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{Var}(s_{ij}) - \mathrm{Cov}(f_{ij}, s_{ij})}{\sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{Var}(f_{ij} - s_{ij}) + (\phi_{ij} - \sigma_{ij})^2}
\end{aligned} \tag{2.27}$$

With the assumption of $T \to \infty$ while N is fixed and finite, Olivier Ledoit et al. (2003) proved that the "optimal" shrinkage intensity α^* has a form of κ/T and by using an assumption that the asset returns have finite fourth moments, they showed that the constant κ converges to:

$$\kappa = \frac{\pi - \rho}{\gamma} \tag{2.28}$$

where

$$\pi = \sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{aVar}[\sqrt{T}s_{ij}], \tag{2.29}$$

$$\rho = \sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{aCov}[\sqrt{T}f_{ij}, \sqrt{T}s_{ij}], \tag{2.30}$$

$$\gamma = \sum_{i=1}^{N}\sum_{j=1}^{N}(\phi_{ij} - \sigma_{ij})^2 \tag{2.31}$$

in which aVar and aCov denote an asymptotic variance and an asymptotic covariance operators respectively. However, the κ is unknown in practice and must be estimated through estimating the triple elements $< \pi, \rho, \gamma >$.

Firstly, an estimated element for π is

$$\hat{\pi} = \sum_{i=1}^{N}\sum_{j=1}^{N}\hat{\pi}_{ij} \tag{2.32}$$

with

$$\hat{\pi}_{ij} = \frac{1}{T}\sum_{t=1}^{T}\big((x_{it} - \overline{x}_{i\cdot})(x_{jt} - \overline{x}_{j\cdot}) - s_{ij}\big)^2. \tag{2.33}$$

Secondly, an estimated element for ρ is

$$\hat{\rho} = \sum_{i=1}^{N}\hat{\pi}_{ii} + \sum_{i=1}^{N}\sum_{i=1,j\neq i}^{N}\frac{\overline{r}}{2}\Big(\sqrt{\frac{s_{jj}}{s_{ii}}}\hat{\vartheta}_{ii,ij} + \sqrt{\frac{s_{ii}}{s_{jj}}}\hat{\vartheta}_{jj,ij}\Big). \tag{2.34}$$

in which, $\overline{r}$ is an average sample correlation and $\hat{\vartheta}_{ii,ij}$ is an estimated value for $\text{aCov}[\sqrt{T}s_{ii}, \sqrt{T}s_{ij}]$. They are calculated as follows:

$$\overline{r} = \frac{2}{(N-1)N}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} r_{ij}, \tag{2.35}$$

$$\hat{\vartheta}_{ii,ij} = \frac{1}{T}\sum_{t=1}^{T}\big((x_{it} - \overline{x}_{i\cdot})^2 - s_{ii}\big)\big((x_{it} - \overline{x}_{i\cdot})(x_{jt} - \overline{x}_{j\cdot}) - s_{ij}\big), \tag{2.36}$$

where r_{ij} is a sample correlation between the returns of asset i and j and it is defined as:

$$r_{ij} = \frac{s_{ij}}{\sqrt{s_{ii}s_{jj}}}. \tag{2.37}$$

Thirdly, an estimated element for γ is

$$\hat{\gamma} = \sum_{i=1}^{N}\sum_{j=1}^{N}(f_{ij} - s_{ij})^2. \tag{2.38}$$

Finally, an estimated constant κ is

$$\hat{\kappa} = \frac{\hat{\pi} - \hat{\rho}}{\hat{\gamma}}. \tag{2.39}$$

Under the assumption of finite sample, there are scenarios that the ratio $\hat{\kappa}/T \notin [0, 1]$. In those scenarios, we could simply clip the ratio in the interval of $[0, 1]$. Therefore, the optimal shrinkage intensity is

$$\hat{\alpha}^* = \max\left(0, \min\left(\frac{\hat{\kappa}}{T}, 1\right)\right). \tag{2.40}$$

2.2.3.1 Shrinkage to the identity matrix

A simplest model for covariance matrix estimation is just a scalar multiple of the identity matrix $\mu\mathbf{I}$. In which, it assumes that all variances are the same to one another and all covariances are the same to one another. The identity matrix is one of the most well-structured, well-conditioned and always invertible. Therefore, it is a prefered target matrix in various domains. In general, the identity matrix is used in general cases in any domain because of its simplicity for mathematical computations or whenever we have no perspective about the true structure of the covariance matrix.

Using the identity matrix as the target matrix, the shrinkage covariance matrix imposes the structure of the sample covariance matrix toward the identity matrix (Olivier Ledoit et al., 2004a). From another point of view, this shrinkage estimation is a linear combination of the sample portfolio and an equally-weighted portfolio which is an extreme example of a diversified portfolio(DeMiguel, Garlappi, et al., 2009). Although the identity matrix seems to be a non-optimal choice for the target matrix because of its little information, but interestingly, it yields a good shrinkage matrix or at least significantly better than the sample covariance matrix. Therefore, the shrinkage to the identity matrix is used as a baseline for other studies.

2.2.3.2 Shrinkage to the single-index model

In a single-index model, Sharpe (1963) assumed that the asset returns x_{it} are correlated with market returns x_{0t} as follows:

$$x_{it} = \alpha_i + \beta_i x_{0t} + \epsilon_{it} \tag{2.41}$$

in which residuals ϵ_{it} are independent of one another and a variance of each asset is assumed as constant ($\mathrm{Var}(\epsilon_{it}) = \delta_{ii}$ and $\delta = (\delta_{ii})_{i=1,\dots,N}$). The implied covariance matrix is:

$$\mathbf{\Phi} = \sigma_{00}^2 \beta\beta^\intercal + \mathbf{\Delta} \tag{2.42}$$

in which σ_{00}^2 is the variance of the market returns x_{0t}, β is the vector of slopes and $\mathbf{\Delta} = \mathrm{diag}(\delta)$ is a square diagonal matrix containing residual variances.

Applying ordinary least square regression, we obtain the estimated slope vector $\mathbf{b}$ with entries $(b_i)_{i=1,\dots,N}$ and the estimated residual variance vector $\mathbf{d}$ with entries $(d_{ii})_{i=1,\dots,N}$. Then, the covariance matrix estimated from the single-index model is:

$$\mathbf{F} = s_{00}^2 \mathbf{b}\mathbf{b}^\intercal + \mathbf{D} \tag{2.43}$$

in which s_{00}^2 is the sample variance of the market returns and $\mathbf{D} = \mathrm{diag}(\mathbf{d})$.

Using the target matrix $\mathbf{F}$ above, this shrinkage estimator is known as the shrinkage to the single-index model (Olivier Ledoit et al., 2003). This target matrix contains a perspective of

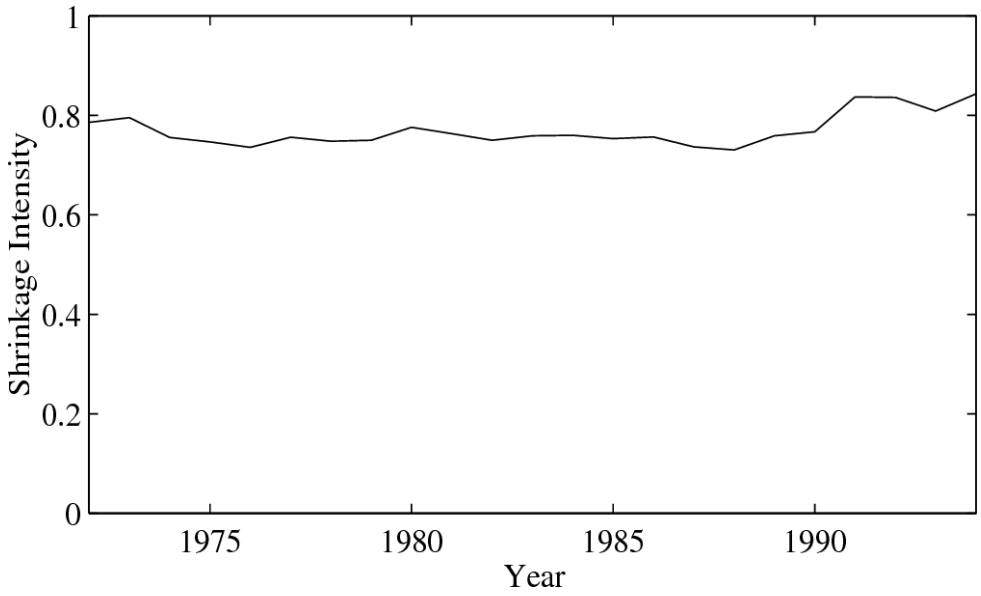


Figure 2.13: Optimal shrinkage intensity of the linear shrinkage to single-index model on the U.S stock market through 23-year data. This is the weight placed on the target matrix, which is the covariance matrix of the single-index model in this case. On the U.S stock market, it is stably high (around 80%). Source: Olivier Ledoit et al. (2003).

the whole stock market, but it is specific for the financial domain. Because of containing the information of the market, it is usually better than the shrinkage to the identity matrix.

Figure 2.13 shows the estimated shrinkage intensity of the U.S stock market through the 23-year data by the shrinkage to the single-index model. It is remarkably stable through time and fairly high (around 80%). It means it uses 80% structure of the single-index model covariance matrix and uses only 20% structure of the sample covariance matrix. It suggests that the estimation error in the sample covariance matrix is four times as there is bias in the single-index model.

2.2.3.3 Shrinkage to the constant-correlation model

The target matrix has two requirements: i) a small number of free parameters to estimate a lot of structure of the covariance matrix and ii) represent (multiple) important characteristics of the true (unobservable) covariance matrix. The single-index model above is a single-factor model to represent the whole stock market, while the industry standard in finance is multiple-factor models such as three, five or fifty factors. The more factors in the model, the more flexibility and accuracy for it. Its bias is reduced while estimation error is increased.

Similarly, another choice for the target matrix is a constant-correlation model. Simply taking an average of all the sample correlations and that constant is assumed as the correlation for all pairwise assets. Let us define the target matrix $\mathbf{F}$ with entries $(f_{ij})_{i,j=1,\dots,N}$ by means of the sample variances and the average sample correlation:

$$f_{ij} = \begin{cases} s_{ii} & \text{if } i = j \\ \bar{r}\sqrt{s_{ii}s_{jj}} & \text{if } i \neq j \end{cases} \tag{2.44}$$

Using the constant-correlation model as the target matrix is known as the shrinkage to the constant-correlation model (Olivier Ledoit et al., 2004b). An advantage of the constant-correlation model is that it yields comparable performance but is easier to implement than the single-index model.

2.2.4 Portfolio performance metrics

In this section, we describe three most common metrics to evaluate a portfolio's performance that how well did it perform in a given period of time: portfolio annual return, portfolio annual volatility and Sharpe ratio. However, there is no single metric that could indicate which portfolio is better. Therefore, in order to have a bigger view for the investors, we also describe in details various portfolio performance metrics. They are useful to analyze the portfolio's advantages and disadvantages.

Let us consider a portfolio in a period $[T_0, T]$ $(T_0 < T)$ with its portfolio values $\mathbf{P}$ and its portfolio returns $\mathbf{R}$, the portfolio return $\mathbf{R}_t$ at date t is:

$$\mathbf{R}_t = \frac{\mathbf{P}_t - \mathbf{P}_{t-1}}{\mathbf{P}_{t-1}}. \tag{2.45}$$

Cumulative return

The portfolio's cumulative return is a total profit or loss of the portfolio for a given period of time. In a period of time $[T_0, T]$ where $T_0 < T$, the cummulative return is calculated as follows:

$$\text{Cumulative return } = \frac{\mathbf{P}_T - \mathbf{P}_{T_0}}{\mathbf{P}_{T_0}}. \tag{2.46}$$

In details, we could compute the cummulative return by using daily portfolio returns as follows:

$$\text{Cumulative return } = \prod_{t=T_0+1}^{T} (\mathbf{R}_t + 1) - 1 \tag{2.47}$$

in which $\mathbf{R}_t$ is the portfolio return at date t where $T_0 < t \leq T$.

Annual return

Annual return is a standardized portfolio return to make it comparable to other portfolios. The annual return could be defined as a yearly profit or loss of the portfolio. Its formula is as follows:

$$\text{Annual return } (R_P) = (\text{Cumulative return} + 1)^{\frac{1}{no\_years}} - 1 \tag{2.48}$$

in which *no_years* is a number of years of a backtesting period. Assuming this is a daily backtesting and there is 252 trading days in one year, then $no\_years = \dfrac{T - T_0}{252}$.

In detail, the portfolio return of a portfolio with several financial assets is the proportion-weighted combination of those assets' annual returns by a following equation:

$$R_P = \sum_i \mathbf{w}_i \mathbf{R}_i \tag{2.49}$$

where $\mathbf{R}_i$ is the return of asset i in the period $[T_0, T]$ and $\mathbf{w}_i$ is the proportion (i.e. weighting) of the asset i in the portfolio such that $\sum_i \mathbf{w}_i = 1$.

Annual volatility

Annual volatility indicates the risk of an investment portfolio in a period of time. However, there is no definition for the risk of portfolio. Therefore we need an alternative approach and Markowitz uses the standard deviation on the portfolio returns to show the uncertainty of the portfolio. The annual volatility is calculated as follows:

$$\text{Annual volatility } (\sigma_P) = \sigma(\mathbf{R})\sqrt{252} \tag{2.50}$$

in which $\sigma(\mathbf{R})$ denotes the standard deviation of the portfolio return series $\mathbf{R}$ and it calculated as follows:

$$\begin{aligned}
\sigma_P^2 &= \text{Var}(\mathbf{R}) \\
&= \mathbb{E}\Big[\Big(\sum_i^N w_i \mathbf{R}_i - \mathbb{E}[\mathbf{R}_i]\Big)^2\Big] \\
&= \mathbb{E}\Big[\Big(\sum_i^N w_i (\mathbf{R}_i - \mathbb{E}[\mathbf{R}_i])\Big)^2\Big] \\
&= \mathbb{E}\Big[\Big(\sum_i^N w_i (\mathbf{R}_i - \mathbb{E}[\mathbf{R}_i])\Big)\Big(\sum_j^N w_j (\mathbf{R}_j - \mathbb{E}[\mathbf{R}_j])\Big)\Big] \\
&= \sum_i^N \sum_j^N w_i w_j \mathbb{E}\big[(\mathbf{R}_i - \mathbb{E}[\mathbf{R}_i])(\mathbf{R}_j - \mathbb{E}[\mathbf{R}_j])\big] \\
&= \sum_i^N \sum_j^N w_i w_j \sigma_{i,j}
\end{aligned} \tag{2.51}$$

where $\sigma_{i,j}$ is the co-variance of two assets i and j.

Sharpe ratio

The portfolio annual return and annual volatility are the two most common portfolio metrics to measure the performance of portfolio. However, considering a portfolio with higher volatility

but has a better return than another portfolio, in this case, investors need to evaluate a trade-off between risk and expected return to know that: is an additional return worth for an increasing risk level? In 1994, Sharpe (1994) introduced a Sharpe ratio to measure the performance of an investment compared to its risk, after adjusting by a risk-free asset. This ratio is determined as follows:

$$\begin{aligned} \text{Sharpe ratio (SR)} &= \frac{\mathbb{E}[\mathbf{R} - \mathbf{R}_f]}{\sigma} \\ &= \frac{\mathbb{E}[\mathbf{R} - \mathbf{R}_f]}{\sqrt{\text{Var}[\mathbf{R} - \mathbf{R}_f]}} \end{aligned} \tag{2.52}$$

where $\mathbf{R}$ is the investment's return, $\mathbf{R}_f$ is the risk-free rate in the same investment horizon such as a return of a short-term government treasury bond. $\mathbf{R} - \mathbf{R}_f$ is an excess return series. $\mathbb{E}[\mathbf{R} - \mathbf{R}_f]$ and σ are the expected value and the standard deviation of the excess return. A portfolio with a higher Sharpe ratio is more attractive to investors.

Max drawdown

The maximum drawdown is also an important indicator for portfolio efficiency evaluation. This indicator reflects the portfolio's level of risk in the impoverished and complicated market situation. The maximum drawdown can be calculated as follows:

$$\text{Max drawdown } (MDD) = \underset{t^* \in [T_0, T]}{\arg\max} \left[\underset{t \in [T_0, T]}{\arg\max} \left(\frac{\mathbf{P}_t - \mathbf{P}_{t^*}}{\mathbf{P}_t} \right) \right]. \tag{2.53}$$

Clearly, a lower maximum drawdown will attract investors because it shows that the investment strategy is less risky.

Daily Value at Risk

Value at Risk (VaR) estimates the risk of loss of the portfolio. It answers a question "How much a set of investments might lose". Its formula is as follows:

$$\text{VaR}_\alpha(X) = -\inf \{x \in \mathbb{R} : \boldsymbol{F}_X(x) > \alpha\} \tag{2.54}$$

in which, X is a distribution of the portfolio's profit and loss, $\boldsymbol{F}_X$ is a cumulative distribution function of random variable X and $\alpha \in (0, 1)$ is a level of risk we are considering. Common levels of α are 1% and 5% in a time horizon of one day or two weeks (Pearson, 2011). In default of the VaR in our system, we use daily Value at Risk at level 5%.

Daily turnover

This indicator shows the stability of the portfolio at a a time when the portfolio is changing its status according to an optimal strategy. Therefore, the investors will prefer a lower turnover, because this shows that the liquidity risks will reduce and the transaction costs are also going lower. The portfolio turnover is defined as follows:

$$\text{Portfolio Turnover} = \frac{1}{T - T_0 - 1} \sum_{t=T_0}^{T-1} \sum_{i=1}^{N} \left(|\mathbf{w}_{t+1,i} - \mathbf{w}_{t,i}| \right) \tag{2.55}$$

in which, $\mathbf{w}_{t,i}$ is a weight value of asset i at date t.

Alpha

This metric is a measure of the portfolio's superior return to the theoretical expected return. The theoretical expected return is calculated by Capital Asset Pricing Model (CAPM), which relies on the Beta coefficient and the average market return. The metric is also generally known as Jensen's alpha and is identified as follows:

$$\alpha = R_P - [R_f + \beta(R_M - R_f)] \tag{2.56}$$

in which R_P is the average return of the portfolio, R_f is the risk-free rate, β is beta coefficient of the portfolio and often estimated by Ordinary Least Square and R_M is the average market return.

Beta

In finance, a Beta (β) measures a co-movement of a portfolio and the market. In other words, the Beta could be seen as the similarity of the portfolio and the market. We denote the variance and covariance operators as Var and Cov respectively. Its formula is as follows:

$$\beta = \frac{Cov(\mathbf{R}, \mathbf{R_M})}{Var(\mathbf{R_M})}. \tag{2.57}$$

Winning rate

Winning rate is the fraction of bets has won. In a trading context, it is the number of positive returns over the number of returns. Its formula is as follows:

$$\text{Winning rate (WR)} = \frac{\sum_{t=T_0}^{T} [\mathbf{R}_t > 0]}{T - T_0}. \tag{2.58}$$

Running time

This indicator measures a total running time in seconds of a given trading strategy. In the investment industry, a strategy has to be optimized to run faster. Therefore, we also describe this indicator.

2.2.5 Backtesting

In order to evaluate a trading strategy on the time-series of stock data precisely, we rely on our backtesting system which accurately handles every event through the time-series dataset. In this section, we comprehensively describe our backtesting system and show an example of usage.

2.2.5.1 Introduction

Instead of using a simple portfolio evaluation approach with a few lines of code which is easy to implement, easy to use but impractical. Investors need to have a reliable portfolio performance in order to compare trading strategies. Therefore, the backtesting system is important in the investment industry. Given a trading strategy, the backtesting system estimates the portfolio performance and also reveals the disadvantages of that strategy in the future scenarios of the real market. We built our backtesting system to automatically evaluate our research ideas and simplify our processes, in which we only focus on the calculation for portfolio selection and the backtesting system does the rest for us.

Assuming we could invest an infinite amount of money into a "trash" stock which has only one transaction of one share to increase its price from one cent to two cents, definitely we will have an impossible return and significantly beat the market[1]. This assumption and many other simple assumptions are integrated into our backtesting system. There are some assumptions we have to simplify because they need more research on their effects, e.g. market impact, and are out of the scope of this thesis.

There are many aspects that need to be considered in building the backtesting system. For example, a portfolio calculation cannot see the future data[2]. To make our results more reliable, we handle these problems carefully and systematically. In this section, we will discuss about challenges of the backtesting system.

2.2.5.2 Architecture

Similar to the microservice architecture, the backtesting system is designed with multiple modules and each module has different purpose. They communicate with each other or use the results from other modules. Including:

- A source code of trading strategy. In the outside of the backtesting system, the users (i.e. we) focus only on their trading strategy or their portfolio. It has two stages:
  - Initialization stage: The strategy could define its initial parameters or any calculation before actually running the backtesting process. This is an optional stage for the strategy,
  - Handling stage: The strategy has to define a function to calculate its portfolio. From a given dataset as its input, an output of this function is its portfolio weights which will

[1] It is called Paper Trading, but our example is an extreme case.
[2] In short, this is Look-Ahead Bias.

be translated automatically to long/short orders, and those orders will be submitted to the system.

- A core backtesting module: At each time point in the original dataset, the system will prepare a subset from the beginning of the original dataset up to the current time point. Then it feeds this subset to the handling function above and receives the orders. Those orders will be matched with the prices at a next time point then it repeats the backtesting process. Finally, we have all information about the portfolio's behaviours and their results, then we compute various portfolio performance metrics,

- An API module to receive a source code file and other parameters then execute the core module. With the detailed results from the core module, it easily visualizes the portfolio performances.

Formal description

We formally describe the backtesting process and also summarize it as a pseudo-code in Algorithm 0. Moreover, we want to make the backtesting results as reliable, therefore we integrate any regulations of real markets into our system. For example, instead of assuming no impact of the risk-free rate R_f, our backtesting system on the Vietnam stock market will practically consider R_f as a 10-year interest rate of Vietnam government bonds.

Let's consider N assets from the beginning of the market ($t = 0$) to the present time point ($t = T$), we denote the stock prices of all assets $\mathbb{D}^N_{[0:T]}$ and the traded volume $\mathbb{V}^N_{[0:T]}$. Suppose that we want to perform backtesting in the period $[T_1, T_2]$, such that $0 < T_1 < T_2 \leq T$[3], with an initial capital $\mathbf{P}_{T_1}$ and the first return value $\mathbf{R}_{T_1} = 0$, the backtesting process for each timepoint $t \in [T_1, T_2]$, including two phases, is describes as follow.

In the first phase, we calculate all the necessary information for the portfolio. In more detail, with the current actual vector of volume $\mathbf{V}_{t-1}$ (i.e., real holding assets at timepoint $t-1$) and their latest prices data $\mathbb{D}_t$, we compute a current portfolio value by $\mathbf{P}_t = \mathbb{D}_t \cdot \mathbf{V}_{t-1}$ and a current portfolio return $\mathbf{R}_t = \dfrac{\mathbf{P}_t}{\mathbf{P}_{t-1}} - 1$. Please note that at the first timepoint $t = T_1$, the first value of portfolio and portfolio return are $\mathbf{P}_{T_1}$ and 0 respectively. We then derive the matched number of shares by taking the minimum value of the previous ordered volume $\mathcal{V}_{t-1}$ and the current real traded volume $\mathbb{V}_t$[4]. Now the actual volume at present $\mathbf{V}_t$ is determined by the previous actual volume (at timepoint $t-1$) plus the matched shares. The backtesting system automatically handles this phase base on previous input information before switching to the next phase.

In the second phase, the goal is to determine a target portfolio from the currently available data $\mathbb{D}^N_{[0:t]}$ (the data from the beginning to the timepoint t). We start by using the shrinkage weights $\hat{\mathbf{w}}_*$ to estimate the vector of weights at timepoint t of N assets $\mathbf{w}_t$ and then the target volume $\mathbf{v}_t = \mathbf{P}_t\mathbf{w_t}$. To move the portfolio from current position $\mathbf{V}_t$ to the target position $\mathbf{v}_t$, we place an order number of shares $\mathcal{V}_t = \mathbf{v}_t - \mathbf{V}_t$ (negative values mean selling orders).

[3]The $T_1 \neq 0$ because at the first timepoint $t = 0$ there is no data for the calculation.

[4]To simplify the process, we assume that we can buy/sell up to the maximum volume of the real market and no effect of market impact to our trading strategies.

When all computational steps are finished at timepoint t, the backtesting system will repeat the whole process at the next timepoint $t+1$, and so on, until the end of the backtesting period. Finally, once we have all related information at every timepoint, such as returns $\mathbf{R}$, the system will calculate and output the performance indicators of the portfolio. One of the essential advantages of our backtesting system is that it can carry out the whole process sequentially and precisely in a real-time manner to ensure that we do not get any mistakes in time series testing (e.g., Look-Ahead Bias).

Algorithm 2 Backtesting process for trading strategies from historical stock data.

Input: Historical daily prices of N assets $\mathbb{D}^N_{[0:T]}$, historical daily traded volumes of N assets $\mathbb{V}^N_{[0:T]}$, initial portfolio capital $\mathbf{P}_1$, backtesting period $[T_1, T_2]$, historical daily returns of benchmark $\mathbf{B}_{[0:T]}$[5]

Output: List of various portfolio performance indicators

$\mathbf{R}_{T_1} = 0$, $\mathbf{V}_{T_1} = \mathbf{0}$
for $t = T_1 : T_2$ **do**
  if $(t > T_1)$ **then**
    $\mathbf{P}_t = \mathbb{D}_t \cdot \mathbf{V}_{t-1}$
    $\mathbf{R}_t = \dfrac{\mathbf{P}_t}{\mathbf{P}_{t-1}} - 1$
    $\mathbb{V}_t = \mathbb{V}^N_{[0:T]}|_t$
    $\mathbf{V}_{\text{matched}} = \left(\min(\mathbb{V}^1_t, |\mathcal{V}^1_{t-1}|), ..., \min(\mathbb{V}^N_t, |\mathcal{V}^N_{t-1}|)\right)$
    $\mathbf{V}_t = \mathbf{V}_{t-1} + \text{sign}\left(\mathcal{V}_{t-1}\right) \odot \mathbf{V}_{\text{matched}}$
  end if
  $\mathbf{w}_t = \hat{\mathbf{w}}_* \left(\hat{\delta}\left(\mathbb{D}^N_{[0:t]}\right), \mathbb{D}^N_{[0:t]}\right)$
  $\mathbf{v}_t = \mathbf{P}_t \mathbf{w}_t$
  $\mathcal{V}_t = \mathbf{v}_t - \mathbf{V}_t$
end for
Compute respectively the quintuple of portfolio's indicators $(R, \sigma, SR, PT, \alpha)$ by equations 4.6, 4.7, 4.5, 4.8, 4.9.
Return (R, σ, SR, PT, α)

Utility functions

The source code of trading strategy should focus on its calculation part. Therefore, there are some utility functions in our system to help it reduce its length and remove repeated code. Two common utility functions are:

- *get_universe*: at any timepoint in the backtesting period, this function gets all valid asset tickers in the universe[6] then returns them as a list object,
- *rebalance_portfolio*: with a weights object as its input parameters, this function will convert the assets' weights from percentages to their expected positions. Comparing to their

[6]They are listed assets but for some reason, the regulator marks them as untradable. Therefore, their data are still available in the dataset but we have to ignore them.

current positions and by taking into account the commission fee, this function will place long/short orders to fulfil the portfolio's positions.

Code example

We illustrate the efficiency of our backtesting system by using the simplest trading strategy, the Equally-Weighted (EW) portfolio. A Python code of this strategy is less than ten lines as follows:

```
1: from backtest.api import get_universe, rebalance_portfolio


2: def initialize(context):
3:     pass


4: def handle_data(context, data):
5:     universe = get_universe(context, data)
6:     weights = {asset: 1/len(universe) for asset in universe}
7:     rebalance_portfolio(context, data, weights)
```

In which:

- A first line is to import two functions we will use later,
- A second line is a static line to define a function that will be executed before running the backtesting process,
- And a next line is to skip the initialization function above because we don't have any parameter to define or calculate at this time,
- Similarly, a fourth line is a static line to define a handling function which will be executed at each timepoint in the backtesting period. It has two fixed parameters:
  - *context*: this object contains information about current context, e.g. we could get a current timepoint by using command *context.datetime*,
  - *data*: this is a subset that contains the data up to current time point.
- A fifth line is to get all valid assets' tickers in our universe at current timepoint by using function *get_universe*,
- At a sixth line, the EW portfolio' weights for all assets is one over the number of assets,
- Finally, we rebalance our EW portfolio by using the function *rebalance_portfolio* from the *weights* object.

Backtesting parameters

There are many customizable parameters in the backtesting system in which some of them are required. Their descriptions and default values are in Table 2.1.

We will test the above Equally-Weighted portfolio with following parameters:

- Starting date is 2013-01-01,
- Ending date is 2019-12-31,
- Initial capital is 10^9 (i.e. one billion VND),
- Benchmark is VN-Index.

Dataset

We briefly describe a dataset in this experiment for the EW portfolio above. We use daily historical data of the Vietnam stock market, particularly is HOSE exchange. Backtesting period is from 2013 to the end of 2019 with 1744 trading days, prior to 2013 is for computing preparation. The number of assets varies across backtesting period between minimum N = 303 and maximum N = 387.

Output

Table 2.2 reports final portfolio performances of the EW portfolio above. Figure 2.14 visualizes the cummulative returns of the EQ portfolio and the benchmark over time. Figure 2.15 compares annualized returns of them over each year. Figure 2.16 visualizes the maximum drawdown of the EW's returns over time. Figure 2.17 visualize top five periods that have largest drawdown of the EW portfolio. Figure 2.18 and Figure 2.19 visualize daily turnover and portfolio weights of the EW portfolio over time.

Table 2.1: Descriptions of seven parameters in the backtesting system.

Parameter	**Required field**	**Default value**	**Description**
start	Required	N/A	Starting date of the backtesting process, at the first date the prior data up to this date are provided to the portfolio calculation function,
end	Required	N/A	Ending date of the backtesting process,
capital_base	Optional	10^6	Initial capital of the portfolio at the first date, default is one billion VND. This amount is chosen because it is not too high or too low. A portfolio with high capital, e.g. institutional investors, is out of the scope of this thesis because there are many aspects they have to consider, e.g. portfolio liquidity. A portfolio with low capital, e.g. retail investors, is not easy to analyze the portfolio characteristics because they don't have enough money to invest in portfolio tail,
bm_ticker	Optional	VN-Index	Benchmark for the portfolio, default is VN-Index. The benchmark returns are the same period as the portfolio (i.e. from starting date to ending date). This could be another market index or another asset. In this scope of this thesis, we are using the Vietnam stock market, therefore we chose the Vietnam index as our benchmark,
universe	Optional	HOSE	An universe for the portfolio, default is HOSE - the biggest stock exchange in Vietnam. At each date, the portfolio will receive the data of all listing assets in the universe and all delisted assets will be removed from the dataset, therefore the portfolio calculation doesn't have to deal with these changes,
trading_unit	Optional	10	A trading unit of the market. Particularly for the Vietnam stock market, a unit for an order is 10 shares according to their regulations,
commission_fee	Optional	0.03%	A commission fee for each order, default is 0.03% according to Vietnam regulations.

Table 2.2: The out-of-sample performance results of the Equally-Weighted portfolio on the Vietnam stock market from 2013 to the end of 2019.

Indicator	**Result**
Annual return	15.4273%
Cumulative returns	169.9080%
Annual volatility	10.1984%
Sharpe ratio	1.1072
Max drawdown	-19.9252%
Daily value at risk	-1.2259%
Daily turnover	1.3885%
Alpha	0.0654
Beta	0.5056
Winning rate	58.4289%
Running time	2728

Figure 2.14: Visualization of the cumulative returns of the Equally-Weighted (EW) portfolio on all stocks in the HOSE from 2013 to the end of 2019. A red line is the cumulative returns of the benchmark (VN-Index), and a blue line is the cumulative returns of the EW portfolio. The unit of the left axis is the percentage.

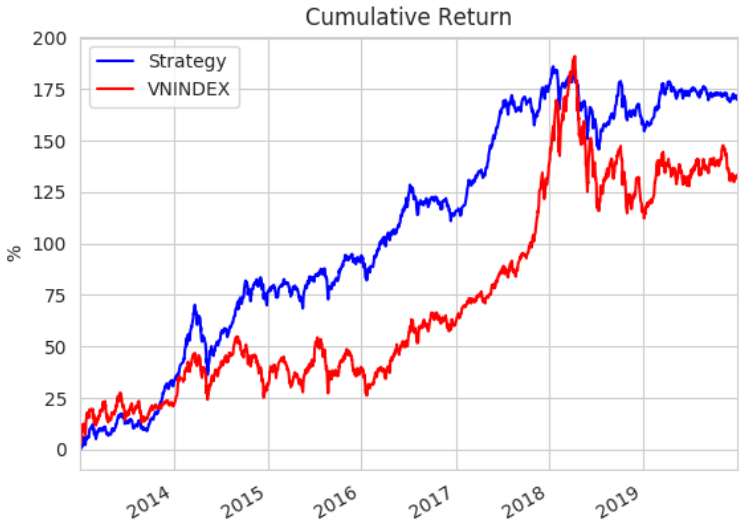


Figure 2.15: Visualization of the yearly Annual returns of the Equally-Weighted portfolio on all stocks in the HOSE from 2013 to the end of 2019. Comparing to the yearly Annual returns of the benchmark (VN-Index). The unit of the left axis is the percentage.

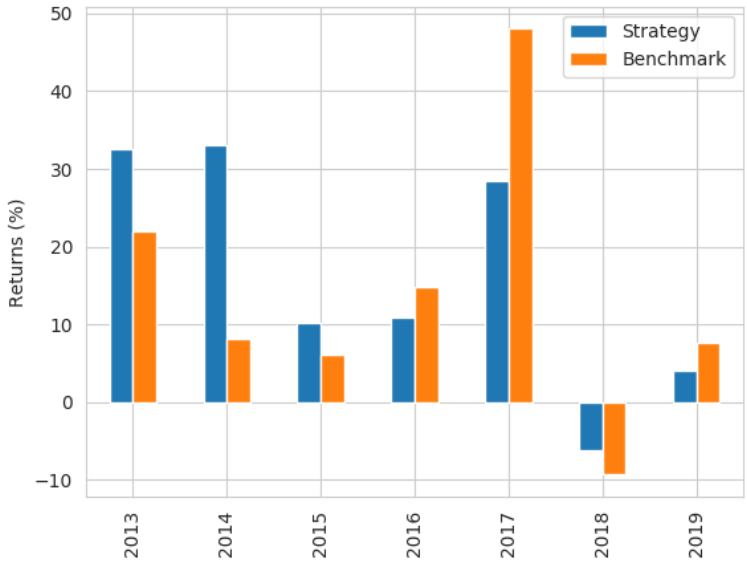


Figure 2.16: Visualization of the Maximum Drawdown of the Equally-Weighted portfolio on all stocks in the HOSE from 2013 to the end of 2019. The unit of the left axis is the percentage.

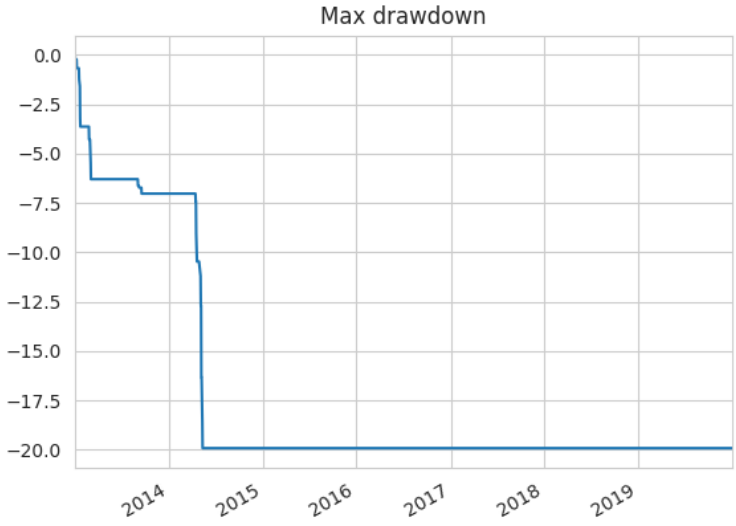


Figure 2.17: Visualization of top five largest Maximum Drawdown of the cumulative returns over the time from 2013 to the end of 2019 of the Equally-Weighted portfolio on all stocks in the HOSE.

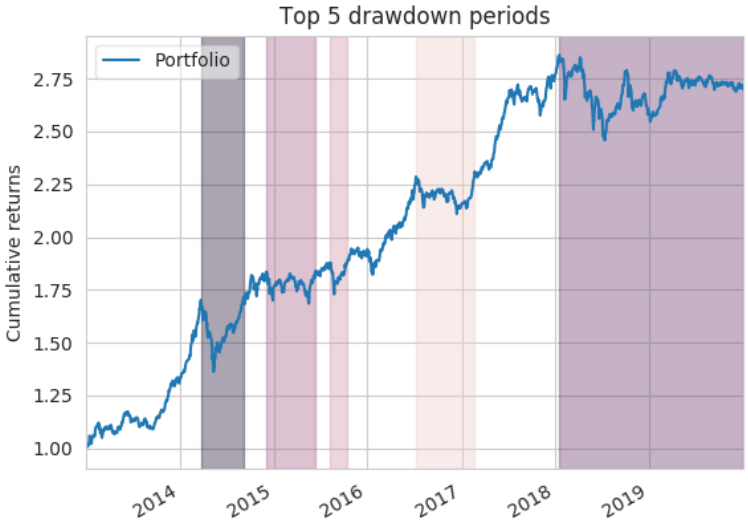


Figure 2.18: Visualization of the daily turnover of the Equally-Weighted portfolio on all stocks in the HOSE from 2013 to the end of 2019. A possible maximum value of the daily turnover is two, i.e. 200%.

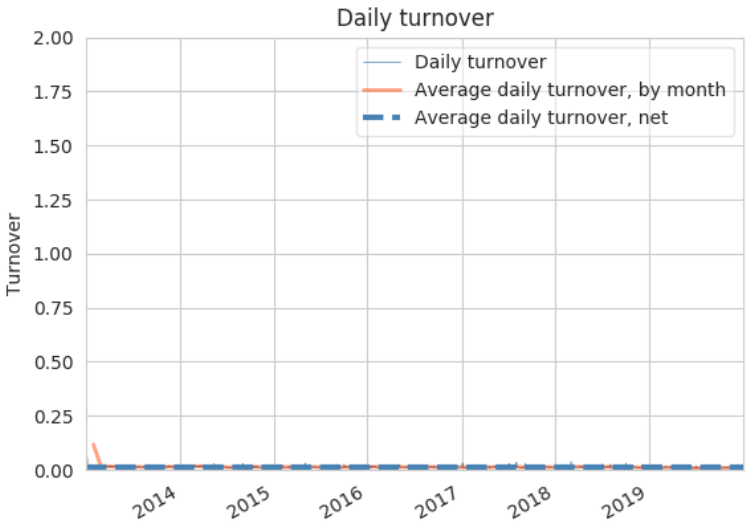
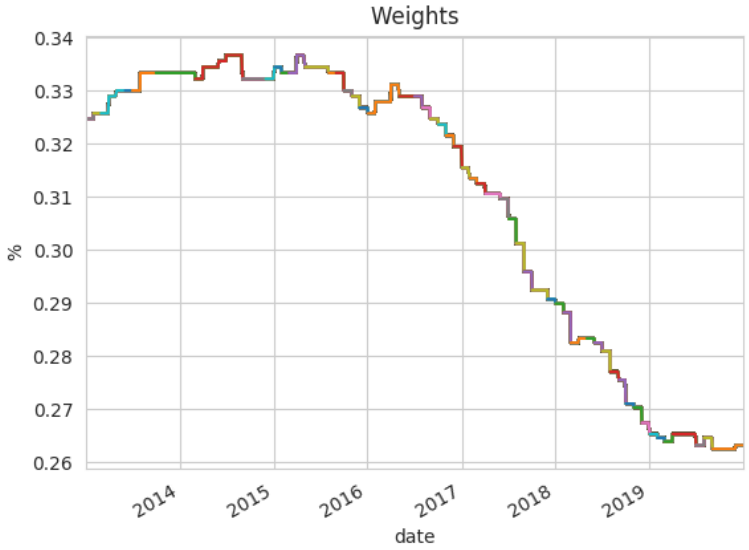


Figure 2.19: Visualization of the weights of all stocks in the Equally-Weighted portfolio from 2013 to the end of 2019. The unit of the left axis is the percentage.



References

BiPM, IEC, ILAC IFCC, IUPAC ISO, and OIML IUPAP (2008). "International vocabulary of metrology–basic and general concepts and associated terms, 2008". In: *JCGM* 200, pp. 99–12 (cit. on p. 17).

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32 (cit. on pp. 16, 104, 143).

Breiman, Leo, Jerome Friedman, Charles J Stone, and Richard A Olshen (1984). *Classification and regression trees*. CRC press (cit. on p. 13).

Chopra, Vijay K and William T Ziemba (2013). "The effect of errors in means, variances, and covariances on optimal portfolio choice". In: *Handbook of the fundamentals of financial decision making: Part I*. World Scientific, pp. 365–373 (cit. on pp. vii, 24, 25).

Cox, David R (1958). "The regression analysis of binary sequences". In: *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 215–242 (cit. on p. 12).

De Prado, Marcos Lopez (2018). *Advances in financial machine learning*. John Wiley & Sons (cit. on p. 21).

DeMiguel, Victor, Lorenzo Garlappi, Francisco J Nogales, and Raman Uppal (2009). "A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms". In: *Management science* 55.5, pp. 798–812 (cit. on pp. 32, 94, 107, 141, 147).

Freund, Yoav, Robert Schapire, and Naoki Abe (1999). "A short introduction to boosting". In: *Journal-Japanese Society For Artificial Intelligence* 14.771-780, p. 1612 (cit. on p. 14).

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani (2001). *The elements of statistical learning*. Vol. 1. Springer series in statistics New York (cit. on p. 17).

Golosnoy, Vasyl, Bastian Gribisch, and Miriam Isabel Seifert (2022). "Sample and realized minimum variance portfolios: Estimation, statistical inference, and tests". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 14.5, e1556 (cit. on pp. vii, 25).

Hanczar, Blaise, Jianping Hua, Chao Sima, John Weinstein, Michael Bittner, and Edward R Dougherty (2010). "Small-sample precision of ROC-related estimates". In: *Bioinformatics* 26.6, pp. 822–830 (cit. on p. 20).

Hand, David J (2009). "Measuring classifier performance: a coherent alternative to the area under the ROC curve". In: *Machine learning* 77.1, pp. 103–123 (cit. on p. 20).

Hanley, James A and Barbara J McNeil (1983). "A method of comparing the areas under receiver operating characteristic curves derived from the same cases." In: *Radiology* 148.3, pp. 839–843 (cit. on p. 20).

Koza, John R, Forrest H Bennett III, David Andre, and Martin A Keane (1996). "Automated design of both the topology and sizing of analog electrical circuits using genetic programming". In: *Artificial Intelligence in Design'96*. Springer, pp. 151–170 (cit. on p. 12).

Ledoit, Olivier and Michael Wolf (2003). "Improved estimation of the covariance matrix of stock returns with an application to portfolio selection". In: *Journal of empirical finance* 10.5, pp. 603–621 (cit. on pp. viii, 6, 28–30, 32, 33, 89–91, 103, 105, 137, 138).

— (2004a). “A well-conditioned estimator for large-dimensional covariance matrices”. In: *Journal of multivariate analysis* 88.2, pp. 365–411 (cit. on pp. viii, 6, 28, 32, 89, 91, 103, 105, 137).

— (2004b). “Honey, I shrunk the sample covariance matrix”. In: *The Journal of Portfolio Management* 30.4, pp. 110–119 (cit. on pp. 6, 34, 89, 91, 103, 106, 137).

Lobo, Jorge M, Alberto Jiménez-Valverde, and Raimundo Real (2008). “AUC: a misleading measure of the performance of predictive distribution models”. In: *Global ecology and Biogeography* 17.2, pp. 145–151 (cit. on p. 20).

Markowitz, Harry (1952). “Portfolio selection”. In: *The journal of finance* 7.1, pp. 77–91 (cit. on pp. 5, 22).

Michaud, Richard O (1989). “The Markowitz optimization enigma: Is ‘optimized’optimal?” In: *Financial analysts journal* 45.1, pp. 31–42 (cit. on p. 24).

Mitchell, Tom M et al. (1997). *Machine learning. WCB* (cit. on p. 12).

Mohri, Mehryar, Afshin Rostamizadeh, and Ameet Talwalkar (2012). *Foundations of machine learning*. MIT press (cit. on p. 12).

Olson, David L and Dursun Delen (2008). *Advanced data mining techniques*. Springer Science & Business Media (cit. on p. 18).

Opitz, David W and Richard Maclin (1999). “Popular ensemble methods: An empirical study”. In: *J. Artif. Intell. Res.(JAIR)* 11, pp. 169–198 (cit. on p. 13).

Pearson, Neil D (2011). *Risk budgeting: portfolio problem solving with value-at-risk*. John Wiley & Sons (cit. on p. 36).

Perry, James W, Allen Kent, and Madeline M Berry (1955). “Machine literature searching x. machine language; factors underlying its design and development”. In: *Journal of the Association for Information Science and Technology* 6.4, pp. 242–254 (cit. on p. 18).

Quinlan, J Ross (2014). *C4. 5: programs for machine learning*. Elsevier (cit. on pp. 13, 60).

— (1986). “Induction of decision trees”. In: *Machine learning* 1.1, pp. 81–106 (cit. on p. 13).

Raschka, Sebastian (2015). *Python machine learning*. Packt Publishing Ltd (cit. on pp. vii, 15, 16).

Rokach, Lior (2010). “Ensemble-based classifiers”. In: *Artificial Intelligence Review* 33.1-2, pp. 1–39 (cit. on p. 13).

Rokach, Lior and Oded Maimon (2014). *Data mining with decision trees: theory and applications*. World scientific (cit. on p. 13).

Russell, Stuart, Peter Norvig, and Artificial Intelligence (1995). “A modern approach”. In: *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs* 25, p. 27 (cit. on pp. 12, 13).

Samuel, Arthur L (2000). “Some studies in machine learning using the game of checkers”. In: *IBM Journal of research and development* 44.1.2, pp. 206–226 (cit. on p. 12).

Sharpe, William F (1963). “A simplified model for portfolio analysis”. In: *Management science* 9.2, pp. 277–293 (cit. on pp. 32, 105).

— (1994). “The sharpe ratio”. In: *Journal of portfolio management* 21.1, pp. 49–58 (cit. on pp. 36, 107, 146).

Van Rijsbergen, CJ (1979). "Information retrieval. dept. of computer science, university of glasgow". In: *URL: citeseer. ist. psu. edu/vanrijsbergen79information. html* 14 (cit. on p. 20).

Walker, Strother H and David B Duncan (1967). "Estimation of the probability of an event as a function of several independent variables". In: *Biometrika* 54.1-2, pp. 167–179 (cit. on p. 12).

Wallman, Katherine K (1993). "Enhancing statistical literacy: Enriching our society". In: *Journal of the American Statistical Association* 88.421, pp. 1–8 (cit. on p. 21).

Wikipedia (2017). *Decision tree learning*. URL: https://en.wikipedia.org/wiki/Decision_tree_learning (visited on 10/07/2017) (cit. on p. 13).

— (2018a). *Precision and recall*. URL: https://en.wikipedia.org/wiki/Precision_and_recall (visited on 08/03/2018) (cit. on pp. vii, 18).

— (2018b). *Receiver operating characteristic*. URL: https://en.wikipedia.org/wiki/Receiver_operating_characteristic (visited on 08/29/2018) (cit. on pp. vii, 20).

Wikipedia contributors (2017). *Titanic Survival Decison Tree*. [Online; accessed 22-July-2022] (cit. on pp. vii, 14).

— (2022a). *Confusion matrix — Wikipedia, The Free Encyclopedia*. [Online; accessed 9-October-2022] (cit. on pp. vii, 19).

— (2022b). *Efficient frontier — Wikipedia, The Free Encyclopedia*. [Online; accessed 27-August-2022] (cit. on pp. vii, 23).

— (2022c). *Low-volatility anomaly — Wikipedia, The Free Encyclopedia*. [Online; accessed 28-August-2022] (cit. on pp. viii, 26).

Chapter 3

Adaptive Extreme Imbalance: A combination of Undersampling and Ensemble Learning for Extreme Imbalance Big Data Classification

Objectives

Imbalanced datasets can be found in various real-world applications, such as fraud detection and cancer detection. While many methods have been proposed in the past to handle imbalanced data classification problems, there is a lack of research specifically addressing the issue of extremely imbalanced data. Additionally, imbalanced data are increasingly prevalent in big data analysis, where the volume of data is increasing rapidly. To address these challenges, we propose a combination of under-sampling and ensemble learning as a method to adapt effectively to different scenarios of extreme imbalance. Through experimental analysis on 11 datasets from various sources, we demonstrate that our proposed method is not only competitive with commonly used methods such as Under Bagging and RUSBoost but also demonstrates superior performance, particularly in the context of extremely imbalanced big data classification problems.

Contents

3.1 Introduction

Imbalanced datasets are a common issue in many real-world domains, such as fraud detection and cancer detection. The task in these cases is to identify a small number of positive data points (minority class) among a large number of redundant data points (majority class). For example, in a classification task with an imbalance ratio (IR) of 99, where only 1 out of 100 samples is a positive sample. This presents a significant challenge for the Machine Learning algorithms. Let us imagine that if they maximize their accuracy, then in the worst case, they always have an accuracy of 99% by doing nothing. This lazy classifier marks all samples in our dataset as the majority class, and it has very high accuracy but misclassifies all minority samples. Several studies have reported that they lose performance with the imbalance datasets (C. Chen et al., 2004; X.-w. Chen et al., 2005; J. Wang et al., 2006; Hong et al., 2007).

To address the challenge of imbalanced data, several methods have been proposed, which can be broadly grouped into two levels: algorithmic level and data level. At the algorithmic level, new classifiers are designed or existing algorithms are modified to handle imbalanced data (Bradford et al., 1998; Cieslak et al., 2008). At the data level, pre-processing techniques are applied to the original imbalanced data before applying it to standard classification algorithms. The three most common techniques are under-sampling, over-sampling (Drummond et al., 2003), and Synthetic Minority Over-sampling Technique (SMOTE) (N. V. Chawla, Bowyer, et al., 2002). Several studies (Weiss and F. Provost, 2001; Laurikkala, 2001; Estabrooks et al., 2004) have shown that using a balanced training set with standard algorithms can give better performance. Another approach is the cost-sensitive approach (Ling, Q. Yang, et al., 2004), which aims to minimize misclassification costs, particularly for data points in the minority class. However, this approach requires additional information about the costs, which may not always be available.

In the era of the computing world, the rapid advancement of computer technologies has led to an exponential increase in the volume of data, such as in genome biology or banking systems. Big data, often consisting of millions or billions of records, pose a significant challenge for traditional solutions, particularly in the context of imbalanced data classification (Del Rio et al., 2014; Triguero, Galar, Vluymans, et al., 2015; Fernandez et al., 2017). Traditional methods have been found to be ineffective in handling such large datasets, either due to resource constraints or poor performance. In recent years, the problem of imbalanced data in the big data context has received more attention (Triguero, Rio, et al., 2015; Krawczyk, 2016).

In many practical applications, datasets are not only large in scale but also highly imbalanced, such as in the case of fraud detection where the imbalance ratio is often greater than 1000 (Juszczak et al., 2008; Shuhao Wang et al., 2017; W. Yang et al., 2019; Mekterović et al., 2021; X. Zhang et al., 2021). This scenario of two challenging problems, namely big and extremely imbalanced data classification, requires an effective solution that addresses both issues simultaneously. In this study, we demonstrate that a simplified combination of the under-sampling technique and ensemble learning can effectively handle this extremely imbalanced big data classification problem. Our proposed method provides a promising solution for addressing the challenges of large and imbalanced datasets, offering improved performance over traditional approaches.

The organization of this chapter is as follows: Section 3.2 surveys traditional approaches for handling imbalanced data. Section 3.3 surveys several approaches for handling imbalanced data in a big data context. Section 3.4 presents an overview of recent research related to the topic of extremely imbalanced data, imbalance big data classification and the gap between them. In Section 3.5, we present our proposed methodology, including detailed explanations of the experimental design and results. The chapter concludes with a summary of the key findings and a discussion of future research directions in Section 3.7.

Table 3.1: A survey of 30 studies and datasets which are highly imbalanced or big data. The COCO dataset is imbalanced because of the extreme imbalance between background and foreground concepts.

Source	Dataset	Data type	Data size	Class count	Min class size	Max class size	Imbalance ratio
Masko et al. (2015)	CIFAR-10	Image	60000	10	2340	3900	2.3
Lee et al. (2016)	WHOI-Plankton	Image	3400000	103	3500	2300000	657
Pouyanfar et al. (2018)	Public-cameras	Image	10000	19	14	6986	499
Liao et al. (2010)	CIFAR-100 (1)	Image	6000	2	150	3000	20
Liao et al. (2010)	CIFAR-100 (2)	Image	1200	2	30	600	20
Liao et al. (2010)	20-News-Group	Text	1200	2	30	600	20
Lin et al. (2017)	COCO	Image	115000	2	10	100000	10000
Bucini et al. (2009)	Building-changes	Image	203358	6	222	200000	900
X.-Y. Liu et al. (2008)	GHW	Structured	2565	2	406	2159	5.3
X.-Y. Liu et al. (2008)	ORP	Structured	700	2	124	576	4.6
Khan et al. (2017)	MNIST	Image	70000	10	600	6000	10
Khan et al. (2017)	CIFAR-100	Image	60000	100	60	600	10
Khan et al. (2017)	CALTECH-101	Image	9144	102	15	30	2
Khan et al. (2017)	MIT-67	Image	6700	67	10	100	10
Khan et al. (2017)	DIL	Image	1300	10	24	331	13
Khan et al. (2017)	MLC	Image	400000	9	2600	196900	76
C. Zhang et al. (2016)	KEEL	Structured	3339	2	26	3313	128
Y. Zhang et al. (2018)	CIFAR-10	Image	60000	10	250	5000	20
Y. Zhang et al. (2018)	CIFAR-100	Image	60000	100	25	500	20
C. Huang et al. (2016)	CelebA	Image	160000	2	3200	156800	49
Ando et al. (2017)	MNIST	Image	60000	10	50	5000	100
Ando et al. (2017)	MNIST-back-rot	Image	62000	10	12	1200	100
Ando et al. (2017)	CIFAR-10	Image	60000	10	5000	5000	1
Ando et al. (2017)	SVHN	Image	99000	10	73	7300	100
Ando et al. (2017)	STL-10	Image	13000	10	500	500	1
Dong et al. (2018)	CelebA	Image	160000	2	3200	156800	49
Ding et al. (2017)	EmotioNet	Image	450000	2	45	449955	10000
Buda et al. (2018)	MNIST	Image	60000	10	1	5000	5000
Buda et al. (2018)	CIFAR-10	Image	60000	10	100	5000	50
Buda et al. (2018)	ImageNet	Image	1050000	1000	10	1000	100

3.2 Imbalance problem in traditional data

One of the main challenges in data mining and machine learning is the problem of imbalanced data, as it is popular in many real-world applications. Such as fraud detection in telephone calls (Fawcett et al., 1997) and credit card transactions (Chan et al., 1999). In these cases, the minority class (i.e. fraudulent transactions) is heavily outnumbered by the majority class (i.e. legitimate transactions). For example, in a dataset where only 1% of the data belongs to the minority class and the remaining belongs to the majority class. This classifier always predicts the majority class will achieve an accuracy of 99%. However, this classifier would be essentially useless as it would not be able to detect any instances of the minority class. This highlights the importance of developing effective methods for addressing imbalanced data classification problems.

There are various methods that have been proposed to address the problem of imbalanced data, which can be broadly categorized into two levels: those that operate at the data level and those that operate at the algorithmic level (N. V. Chawla, Japkowicz, et al., 2004). Algorithms that operate at the algorithmic level are specifically designed to handle the minority class by themselves, while algorithms that operate at the data level use sampling strategies to re-balance the data prior to applying traditional Machine Learning algorithms.

At the data level, methods used to tackle imbalanced datasets can be broadly categorized into five main groups: sampling, ensemble, cost-based, distance-based, and hybrid. The sampling techniques focus on balancing the dataset by either removing or replicating data points. Ensemble methods, on the other hand, combine multiple classifiers to enhance performance. Cost-based methods balance the data by adjusting the sample size of each class in accordance with the costs associated with each class. In contrast, distance-based methods focus on the minority class by reducing the distance between the minority and majority classes. Hybrid methods, as the name implies, combine different techniques to achieve a balanced dataset.

One algorithmic approach to handle imbalanced data is to use classifiers that are specifically designed for this problem. Another approach is to use classifiers that minimize the total cost of classification errors by assigning different costs to the minority and majority classes. These are called cost-sensitive classifiers. Additionally, another way to convert a cost-insensitive classifier into a cost-sensitive one is to use wrapper methods that adjust the classifier's decision threshold.

Both data-level and algorithm-level methods that are sensitive to costs can deal with imbalanced data by assigning different costs to the minority and majority classes at both the data and algorithm levels. At the data level, these methods use cost-based techniques to change the sample size of each class so that the costs of each class are balanced (Zadrozny et al., 2003). At the algorithm level, these methods directly minimize costs by using a loss function that is specific to costs. In addition, wrapper methods can convert a cost-insensitive classifier to a cost-sensitive one by adjusting the decision threshold (Domingos, 1999). These methods have been shown to be effective in handling imbalanced data in various applications.

In the context of the class imbalance problem, we focus on the imbalance in class frequency, particularly the imbalance between different classes. However, the class imbalance can also

exist within a single class due to small clusters of examples (Japkowicz, 2001; Jo et al., 2004). The existence of infrequent instances that are hard to categorize is frequently associated with this (Weiss, 1995). This problem is known as small disjuncts, and it can hinder classification performance (Jo et al., 2004; Japkowicz and Stephen, 2002; Japkowicz, 2003; Prati et al., 2004). Rules that encompass a minor group of instances originating from inadequately represented ideas are known as small disjuncts (Weiss, 2004; Rokach and Maimon, 2005; Holte et al., 1989). Although small disjuncts are not the focus of this study, further information can be found in S. Chen et al. (2013).

In this section, we review the various methods for addressing the imbalanced data problem, including resampling techniques, cost-sensitive learning, and ensemble methods. Our focus is on binary classification, because we could decompose multi-class classification problems into a set of binary classification problems. The strengths and limitations of these methods will be discussed.

3.2.1 Data level methods

In the context of imbalanced datasets, methods at the data level aim to modify the dataset prior to applying any classifiers. These techniques are designed to re-balance the training distribution of classes in order to decrease the level of imbalance or reduce noise, e.g. mislabeled samples or anomalies. Several studies have shown that utilizing a balanced training set can improve the performance of traditional algorithms (Weiss and F. Provost, 2001; Laurikkala, 2001; Estabrooks et al., 2004). In this section, we will review some common techniques at the data level, such as oversampling the minority class, undersampling the majority class, or generating synthetic samples of the minority class (see Figure 3.1).

Sampling techniques

Sampling methods are a popular approach to addressing the class imbalance in datasets. Studies have demonstrated that classifiers tend to perform better when trained on a balanced training set (Weiss and F. Provost, 2001; Laurikkala, 2001; Estabrooks et al., 2004). Sampling techniques are straightforward to implement, as they do not take into account any class information when removing or adding observations. These methods are easy to understand and provide a simple solution for rebalancing datasets.

Van Hulse et al. (2007) performed an experimental comparison of eleven machine learning algorithms and seven sampling techniques. The models were evaluated using six performance metrics on thirty-five benchmark data sets. The results indicated that the performance of the sampling techniques was highly dependent on both the learning algorithm and the evaluation metric used. The study found that random undersampling performed well overall, outperforming random oversampling and other intelligent sampling methods in most cases. However, the study also suggests that no single sampling method is guaranteed to perform best in all problem domains, and it is recommended to use multiple performance metrics when evaluating results.

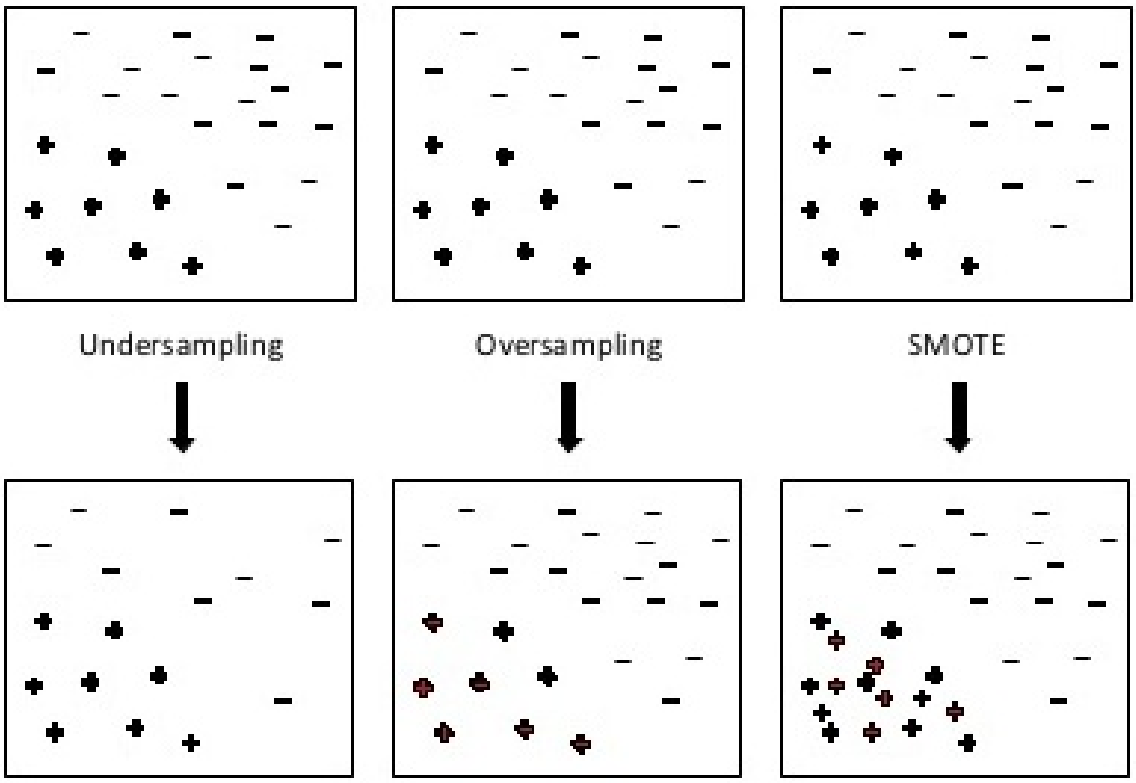
3.2.1.0.1 Undersampling Undersampling (Drummond et al., 2003) is a popular technique used to balance an imbalanced dataset by reducing the number of instances in the majority class. The method works by randomly removing instances from the majority class with the assumption that some of these instances are redundant (see Figure 3.1). However, this method has its limitations. Since the process is unsupervised, there is a risk of dropping important instances which may negatively impact the performance of the classifier. Additionally, a perfectly balanced dataset is not always the optimal solution for undersampling as it may not provide enough information to train the classifier effectively (Dal Pozzolo, Caelen, Johnson, et al., 2015). Despite these limitations, undersampling is widely used in practice due to its simplicity and efficiency in speeding up the training phase.

3.2.1.0.2 Oversampling Oversampling (Drummond et al., 2003) is a method used to address the problem of imbalanced datasets by increasing the number of instances of the minority class. This is done by duplicating instances from the minority class at random, with the goal of balancing the class distribution and making it more similar to the majority class (see Figure 3.1). However, oversampling can also increase the risk of overfitting by biasing the classifier towards the minority class. Additionally, oversampling does not add any new information for minority instances, and it also slows down the training phase. Despite these drawbacks, oversampling is a widely used technique in imbalanced datasets as it can help improve the performance of classifiers by providing more data for the minority class. However, this can be particularly ineffective when the original dataset is fairly large.

3.2.1.0.3 SMOTE SMOTE (Synthetic Minority Over-sampling Technique) is an oversampling method used in imbalanced datasets to generate synthetic samples of the minority class (N. V. Chawla, Bowyer, et al., 2002). It combines both oversampling and undersampling techniques by creating synthetic samples of the minority class that are similar to actual instances but not identical to any of them (see Figure 3.1). This method is effective in increasing the accuracy of classifiers by creating clusters around each minority instance and building larger decision regions. It is a widely-used technique in machine learning and has been shown to be effective in various applications. However, it has some drawbacks, such as the risk of increasing the overlapping area between the classes by creating new minority instances without considering their neighbourhood (B. Wang et al., 2004).

Variations of the SMOTE have been proposed to address its disadvantages. Borderline-SMOTE, as introduced by Han et al. (2005), and Safe-Level-SMOTE, as proposed by Bunkhumpornpat et al. (2009), both consider the majority class neighbours in their approach. Borderline-SMOTE focuses on samples close to the class borders, while Safe-Level-SMOTE defines safe regions to avoid over-sampling in noisy or overlapping regions. These modifications aim to improve the effectiveness of the original SMOTE algorithm.

Figure 3.1: Illustrations of three common sampling methods. These methods include undersampling, oversampling, and SMOTE (Synthetic Minority Over-sampling Technique). The positive and negative signs in the illustrations denote the minority and majority classes respectively, and the new data points created by oversampling methods are represented in red. Source: Dal Pozzolo, Caelen, Waterschoot, et al. (2013).



Cost-based sampling

Cost-based methods are a type of sampling technique that takes into account the misclassification cost associated with each instance. These methods assign different values to each instance based on the cost of misclassifying them. Unlike the random sampling techniques described above, cost-based sampling methods assign a higher weight to the minority class instances than to the majority class instances. This allows for a more balanced dataset, which can improve the performance of classifiers. Cost-based methods are an important approach to consider when dealing with imbalanced datasets, as they allow the user to assign different costs to different instances, taking into account the specific characteristics of the problem at hand.

Zadrozny et al. (2003) proposed Costing, a cost-based undersampling technique that draws instances with an acceptable probability greater than a pre-defined threshold. Furthermore, Klement et al. (2009) proposed a more sophisticated and effective approach which combines cost-based random under-sampling and ensemble learning. The cost-based methods are promising when dealing with imbalanced datasets. However, determining the misclassification cost for each class may not be straightforward in real-life scenarios.

Distance-based sampling

Distance-based methods are a different approach to handle imbalanced datasets compared to cost-based methods. Instead of considering the cost of misclassification, these methods take into account the distance between instances in the imbalanced dataset in order to undersample or remove noise and borderline instances of each class. These methods are computationally more expensive than other techniques as they require the computation of the distance between instances. However, distance-based methods have been found to be effective in some cases.

Tomek (1976) proposed a method to improve class separation by removing instances from the majority class that are close to the minority class. The technique takes into account two examples x_i and x_j from separate classes and computes their distance $d(x_i, x_j)$. A Tomek link is formed between two examples when there is no other example that is closer to either of them. This indicates that one of the examples may be noisy or both are borderline cases. By removing these instances, the majority class is reduced, which can be useful in datasets with noise or overlapping problems as it can prevent misclassifications by the classifier. This approach is particularly useful in noisy datasets or datasets with overlapping issues, as it can prevent the classifier from making misclassifications (Suman et al., 2005).

Condensed Nearest Neighbor (CNN) (Hart, 1968) is a technique used to select a subset of data from an imbalanced dataset that is consistent with the original dataset when using the one-nearest neighbour rule (1-NN). The goal of this technique is to remove instances from the majority class that are far from the decision boundary because they may be deemed less significant for the learning process. However, CNN is highly sensitive to noise in the dataset, as many noisy samples will be added to the subset, leading to the misclassification of subsequent test examples (D. R. Wilson et al., 2000). It is important to note that this method can be used to reduce the size of the dataset, making it more manageable for the classifier to handle and improve performance.

One Sided Selection (OSS) (Kubat et al., 1997) combines the Tomek links and Condensed Nearest Neighbor (CNN). It is particularly useful in datasets that contain a high degree of noise or overlapping problems. The method first uses Tomek links to remove noisy and borderline examples from the majority class. This is followed by using CNN to select a subset of the majority class that is consistent with the original unbalanced set by eliminating examples that are distant from the decision border (Suman et al., 2005). The OSS method is sensitive to noise and aims to remove examples that may not be relevant to learning. The objective of this approach is to remove instances belonging to the majority class that are remote from the decision boundary, as such instances may be deemed less pertinent to the learning process.

Edited Nearest Neighbor (ENN) (D. L. Wilson, 1972) is a method that aims to remove instances from the majority class that are misclassified by at least two of their three nearest neighbours. This approach is particularly advantageous in situations where data is noisy or when datasets exhibit overlapping issues. The ENN technique eliminates instances located within the minority region as well as isolated minority instances. To prevent the loss of pertinent minority instances, the ENN is adapted to only remove negative instances that are incorrectly classified by their three closest neighbors.

Neighborhood Cleaning Rule (NCL) is an extension of the ENN method that places a greater emphasis on data cleaning. NCL initially eliminates negative instances that are incorrectly classified by their three closest neighbors and subsequently removes majority class instances that are neighbors of each positive instance. By removing both noisy instances and points located near the border, the decision boundary is smoothed, thereby reducing the likelihood of overfitting (Suman et al., 2005).

F. Provost (2000) proposed a cluster-based oversampling method that aims to tackle the problem of small disjuncts in the data. The method involves clustering the positive and negative groups using the K-means algorithm and then applying oversampling to each cluster individually. This improves both the imbalance within the class as well as the imbalance between the classes.

Furthermore, alternative hybrid approaches can be developed by combining sampling, ensemble, or distance-based methods to tackle imbalances present within datasets.

3.2.2 Algorithmic level methods

These algorithms are adaptations or extensions of existing methods for classification that can deal with datasets that are imbalanced by increasing the importance of the minority class or decreasing the importance of the majority class. Typically, these modifications involve taking class penalties or weights into consideration or shifting the decision threshold to reduce bias towards the negative class. They can be broadly categorized into three styles: imbalanced learning, cost-sensitive learning and hybrid/ensemble methods. Imbalanced learning algorithms attempt to improve the accuracy of the minority class prediction. On the other hand, cost-sensitive learning algorithms focus on minimizing the cost of wrong predictions by assigning different misclassification costs to different classes. Hybrid/ensemble methods are approaches that combine other methods. However, all of these algorithmic-level methods are often tailored to specific imbalanced datasets, making it important to select the appropriate algorithm based on the characteristics of the data. Additionally, it is worth noting that these algorithms may also be used in combination with data-level techniques, such as under-sampling and over-sampling, to achieve better performance in imbalanced classification tasks.

Imbalance learning

Using Information Gain as a criterion for splitting, Decision Tree (for instance, C4.5 (J Ross Quinlan, 2014)) aims to maximize the number of instances that can be predicted in each node. However, this approach can lead to a bias towards the majority class. To address this issue, Cieslak et al. (2008) proposed splitting with Hellinger Distance (HD), which they found to be skew-insensitive and resulted in improved performance compared to the standard C4.5 algorithm. This bias towards the majority class is not limited to decision trees (He et al., 2009; Japkowicz and Stephen, 2002), but also affects other popular algorithms such as Neural Network (Japkowicz and Stephen, 2002; Visa et al., 2005), k-Nearest Neighbor (kNN) (Kubat et al., 1997; Mani et al., 2003; Batista et al., 2004) and Support Vector Machines (SVMs) (Yan et al., 2003; Wu et al., 2003).

In the family of SVMs, one such method is the optimization of SVMs directly in terms of F-measure, as presented by Callut et al. (2005). Another approach is the use of SVMs with Radial Basis Function (RBF) kernels as the base classifier for AdaBoost, as proposed by X. Li et al. (2008). Within the group of lazy learning classifiers, W. Liu et al. (2011) proposed a k-Nearest Neighbors (kNN) weighting strategy, called CCW-kNN (Class Confidence Weights kNN), designed specifically for handling the problem of class imbalance. This algorithm can rectify the intrinsic unfairness towards the positive class in original kNN classifiers.

Association rule mining is a technique used to discover relationships between variables in large datasets. It can be applied to address the issue of imbalanced classes by setting distinct support thresholds for each class, taking into account the disparities in class distribution (B. Liu et al., 1999). Rule-based classifiers, such as SPARCCC, developed by Verhein et al. (2007), are specifically designed to handle unbalanced classification. These algorithms aim to enhance the performance of the negative class by modifying the original classifier. According to Weiss (2013), these algorithmic solutions should be preferred over data-level methods because they can directly address class imbalance without biasing the classifier towards one class.

Cost-sensitive learning

In classification tasks where imbalanced datasets are present, accurately predicting the minority class is crucial as it often holds greater significance than the majority class. The traditional classifiers may not perform well in identifying the minority class, as they assume that the cost of misclassifying the minority and majority classes is the same. For example, in credit card fraud detection, the cost of a true prediction is zero, while failing to detect a fraudulent transaction results in financial loss. On the other hand, misclassifying a non-fraudulent transaction as fraudulent incurs the cost of investigation and correction. To address this challenge, cost-sensitive learning approaches can be employed, which assign different costs to the prediction of each class (see Table 3.2 for illustration of a cost matrix that can be used to evaluate the cost-sensitive fraud detection system). This allows for the handling of wrong-prediction costs without the need for modifying the dataset. And highlights the importance of utilizing cost-sensitive approaches.

Table 3.2: A sample cost matrix in fraud detection systems to evaluate the cost of different outcomes. It is similar to a confusion matrix used in Machine Learning, but instead of treating all fraud cases as equal, each transaction is evaluated based on its specific cost. Typically, a loss of money from a fraudulent transaction would have a different cost than an investigation fee for a non-fraudulent transaction. The cost matrix helps determine if the cost of investigating a possible fraud is less than the potential loss, in which case it would not be worth further investigation.

	Non-fraud	Fraud
Predict non-fraud	0	amount of money
Predict fraud	investigation fee	0

Integrating cost information into tree-based classifiers is one straightforward approach to cost-sensitive learning. These classifiers can incorporate cost-based criteria during the process

of splitting the tree in order to reduce misclassification costs, as stated in Ling, Q. Yang, et al. (2004). Additionally, tree pruning techniques can also be used to minimize loss, as demonstrated in Bradford et al. (1998).

In cost-sensitive learning, the penalties for each class are defined by the cost matrix. By assigning a higher cost to the minority group, it becomes more important to the algorithm, and thus, the chances of misclassifying instances from this group are reduced (Krawczyk, 2016). In Table 3.3 is an example of binary cost matrix for classification task (Elkan, 2001). Each element of the table, c_{ij}, represents the cost of predicting class i when the true class is j. Typically, the cost for correctly classifying an instance is set to zero on the diagonal of the matrix. By adjusting the costs for false positive and false negative errors, the desired results can be achieved.

According to Ling and Sheng (2008), there are two main categories of cost-sensitive methods: direct methods and meta-learning methods. Instead of minimizing total error, direct methods modify the basic algorithm of a learner to take costs into account during the learning process, which changes the optimization objective to minimizing total cost. On the other hand, any cost-insensitive learner could be transformed to cost-sensitive learner by using meta-learning methods. One example of this is Metacost, a method developed by Domingos (1999), which transforms non-cost-sensitive algorithms into cost-sensitive algorithms.

A study by López et al. (2012) compared how well cost-sensitive learning and over-sampling methods handled class imbalance. The results showed no significant difference between the two approaches. The over-sampling methods were SMOTE and a combination of SMOTE with ENN (Edited Nearest Neighbor by D. L. Wilson (1972)). Their cost-sensitive learners included various adaptations of C4.5, Support Vector Machine, k-Nearest Neighbors, or FHGML methods (Fuzzy Hybrid Genetics-Based Machine Learning). These methods were integrated through a wrapper classifier. In SMOTE+ENN, ENN removed any instances that were incorrectly classified by their three nearest neighbours in the training dataset after applying SMOTE.

Additionally, a new threshold, denoted by δ^*, can be determined by using the cost matrix, for cost-insensitive classifiers that generate posterior probability estimates. This threshold allows for the adjustment of costs associated with false negative and false positive errors, leading to desired results.

$$\delta^* = \frac{c_{10}}{c_{10} + c_{01}} \tag{3.1}$$

One way to convert a cost-insensitive method to a method that considers the costs of prediction errors (and possibly other costs) is to adjust the decision threshold (e.g. by Equation 3.1). This technique, known as thresholding, uses δ^* (Equation 3.1) to change the output threshold for classifying samples as per Ling and Sheng (2008).

A key challenge in cost-sensitive learning is defining a suitable cost matrix which base on previous experiences or through the knowledge of domain experts. Alternatively, the cost of false negatives can be fixed while the cost of false positives is varied and determined through a validation set. This method has the benefit of allowing for the exploration of a range of costs. However, it could be too costly or unrealistic when dealing with large datasets or a large number of features. (Maloof et al., 1997).

Table 3.3: A cost matrix.

	Actual negative	Actual positive
Predicted negative	$C(0,0) = c_{00}$	$C(0,1) = c_{01}$
Predicted positive	$C(1,0) = c_{10}$	$C(1,1) = c_{11}$

Hybrid and ensemble methods

In the literature, various ways of combining data-level and algorithm-level methods have been proposed to tackle class imbalance problems (Krawczyk, 2016). A possible strategy is to first apply sampling methods directly on the training data to lower the imbalance ratio between classes, and then use thresholding techniques or cost-sensitive learning to minimize bias towards the dominant group. Sun et al. (2007) developed three variants of AdaBoost method that combine with the cost-sensitive approach. They progressively amplify the impact of the positive class by adding cost factors to the weight calculations in each iteration of the AdaBoost algorithm. These methods have been proven to perform better than plain boosting methods in most situations.

Two of the most popular methods for aggregating classifiers are Bagging (Breiman, 1996) and Boosting (Freund, R. E. Schapire, et al., 1996). These methods combine an unbalanced strategy with a classifier to explore the distribution of majority and minority classes. BalanceCascade (X.-Y. Liu et al., 2008) is a supervised technique that reduces the size of the majority class by iteratively eliminating instances that are accurately identified by a boosting algorithm. The underlying principle of this method is that observations from the majority class that are easy to classify are not necessary, and by removing them, the algorithm can concentrate on more challenging cases. However, a disadvantage of this approach is that it requires multiple applications of the classification algorithm, increasing computational demands.

EasyEnsemble (X.-Y. Liu et al., 2008) and UnderBagging (Shuo Wang et al., 2009) are methods that integrate multiple models, each of which captures unique characteristics of the original majority class. These techniques operate by generating several balanced training sets through undersampling, training a model on each set, and then combining the predictions in a manner akin to bagging. EasyEnsemble also incorporates boosting, which allows the method to take advantage of both boosting and bagging. Additionally, several studies have integrated undersampling and oversampling in ensembles of support vector machines (SVMs) to improve performance (Vilariño et al., 2005; Kang et al., 2006; Y. Liu et al., 2006; Benjamin X Wang et al., 2010).

SMOTEBoost (N. Chawla et al., 2003) combines boosting with the SMOTE and DataBoost-IM (Guo et al., 2004) generates synthetic samples within the boosting framework to improve the predictive accuracy of both the majority and minority classes. RareBoost (Joshi et al., 2001) modifies the boosting algorithm to increase accuracy on the rare class by emphasizing the difference of true negatives from false negatives and true positives from false positives at each iteration. JOUS-Boost (Mease et al., 2007) generates duplicates of the minority class with oversampling and also introduces perturbations (jittering) by adding independent and identically

distributed noise to minority examples.

Akbani et al. (2004) propose an approach that integrates a modified version of SMOTE and an error cost technique with SVM. They do this to deal with the poor performance of the original SVM when working with highly imbalanced data. They show that their method gave higher performance than random under-sampling (RUS), SMOTE and SVM. Tang et al. (2008) introduce GSVM-RU, which is a modified version of their earlier granular computing-based learning framework Support Vector Machine (GSVM). The method employs under-sampling to address the issues that arise when using an SVM with data that is significantly imbalanced. They compared the GSVM with three other hybrid learners that also use SVMs: SVM-SMOTE, SVM-Weight, and SVM-RANDU. On average, GSVM-RU performs better than the other methods in terms of classification accuracy, as shown by the results. Ahumada et al. (2008) suggest a method that uses clustering in conjunction with an SVM classifier can help to tackle the issue of class imbalance. In the clustering stage, data points from the positive class are split into two distinct clusters. This division is repeated until the datasets produced are either evenly distributed or readily identifiable. The outcome of the clustering process can be represented as a directed acyclic graph, also known as a decision tree. The results indicate that the majority of scenarios show superior performance using their methodology compared to ROS.

Ensemble methods have become a widely used solution for addressing the challenge of class imbalance due to their flexible nature, as reported in previous studies [24, 29]. Some of the well-known methods are Bagging, AdaBoost, and Random Forest (Bekkar et al., 2013; Khoshgoftaar et al., 2007). To improve performance, variations of Bagging such as RUSBagging, Asymmetric Bagging, ROSBagging, and SMOTEBagging have been proposed. Boosting-based ensemble approaches, such as AdaBoost, also have their own variations, such as RUSBoost (Seiffert et al., 2009), ROSBoost (Bekkar et al., 2013), and SMOTEBoost [31]. The Random Forest classifier, which combines bagging with random feature subspace selection, has been adapted to handle class imbalance through variants like Balanced Random Forest and Weighted Random Forest (C. Chen et al., 2004). Additionally, the Balanced Weight Extreme Learning Machine (BWELM) [5] and Extreme Learning Machine (EWELM) [7] are examples of ensemble methods that assign dynamic weights to training samples, with higher weights for misclassified samples and lower weights for correctly classified samples.

Studies suggest that ensemble-based techniques are generally more effective than data sampling methods in addressing the class imbalance, as noted by Galar et al. (2012). They found that SMOTEBagging, RUSBoost, and UnderBagging showed better results compared to other ensemble classifiers, with SMOTEBagging exhibiting a slight advantage. The performance of an ensemble method depends on how well the individual approaches that it combines work together and outperform them individually.

3.2.3 Evaluation measures

Accuracy (see Equation 2.3) is the most common metric for evaluating performance. The error rate (see Equation 3.2) is the opposite of accuracy. However, these metrics are not suitable when the classes are imbalanced, as the larger group, i.e., the negative class, influences the results. For

example, a simple classifier can label all examples as negative and still achieve a 99% accuracy score in a dataset where the positive group makes up only 1% of the data. To prevent misleading results, it's important to use more appropriate evaluation metrics.

$$\text{Error rate} = 1 - \text{Accuracy} \tag{3.2}$$

Precision (Equation 2.6) checks how many positive samples were labelled correctly by a model. It is affected by imbalanced classes because it counts negative samples that were labelled wrongly as positive. Precision alone is not enough to understand the performance of a model, as it does not look at positive samples that were labelled as negative. The model's ability to correctly identify positive samples out of all positive samples is measured by Recall (Equation 2.7). Recall only focuses on the positive group and is not affected by class imbalance. However, it does not account for negative samples that were wrongly classified as positive. Selectivity (Equation 3.3), also known as True Negative Rate (TNR), measures how well your model correctly predicts all possible negative observations. It takes the total number of correctly predicted negative data points and divides it by the total number of all negative data points.

$$\text{Selectivity} = \frac{\text{True Negative}}{\text{True Negative} + \text{False Positive}} \tag{3.3}$$

$$\text{G-Mean} = \sqrt{\text{TNR} * \text{TPR}} \tag{3.4}$$

$$\text{Balanced Accuracy} = \frac{1}{2}(\text{TNR} * \text{TPR}) \tag{3.5}$$

The F-Measure (Equation 2.9), also known as the F_1-score, is a metric that combines Precision and Recall to give a balanced view of the model's performance. The relative weights of both the Precision and the Recall can be adjusted using a coefficient, β. The G-Mean (Equation 3.4) combines the True Negative Rate (TNR) and the True Positive Rate (TPR) by taking the square root of their product to give an overall performance measurement. A performance measure that takes into account both TPR and TNR values and is more attentive to the minority group is Balanced Accuracy (Equation 3.5), which is similar to G-Mean. Despite being superior to Accuracy and Error Rate, these metrics are not effectively evaluating different classifiers and distributions.

A common evaluation method that plots the true positive rate versus the false positive rate is the ROC curve, which was proposed by F. J. Provost et al. (1997). This method illustrates a trade-off between precisely identifying positive instances and wrongly identifying negative instances. Considering a model that outputs continuous probabilities, a threshold could be applied to generate a progression of data points along the ROC curve (He et al., 2009). The area under the ROC curve (AUC) can then be computed as a single metric to compare different models. Weng et al. (2008) also developed a weighted AUC metric that takes into account any cost biases that may affect the calculation of the area.

Davis et al. (2006) argue that ROC curves can overestimate the performance of classifiers on

greatly unbalanced datasets. They suggest using Precision–Recall (PR) curves instead of ROC curves. The authors show that a classifier can only have a higher ROC curve than another if it also has a higher PR curve. This follows from the observation that the false positive rate in ROC, $FPR = FP$, becomes less responsive to changes in FP as the negative class size increases.

Seliya et al. (2009) recommended evaluating machine learning models using multiple performance metrics. In their study, they evaluated two classifiers on 35 different datasets using 22 different performance metrics. To reduce redundancy and improve performance interpretation, they used a statistical technique called common factor analysis to group the metrics. They found that certain metrics complement each other, such as AUC, Brier Inaccuracy (Boichu et al., 2013), and accuracy.

3.3 Imbalance problem in big data

Generally in the class imbalance problem, the strategies for traditional and big data are similar, with solutions implemented either at the algorithm level or the data level. The difference arises from the unique characteristics of big data, such as the need to process a large amount of data using a big data processing framework that distributes computation across many servers. This requires the solution to be implementable in a distributed computing environment. As a result, many class imbalance approaches are not appropriate for big data contexts as they are overly complex for low imbalance problems and cannot be implemented in large-scale systems.

Big data analysis and processing often demand specialized computing systems and algorithms that can leverage parallelism and distributed computing. Some of the well-known frameworks for dealing with big data are MapReduce (Dean et al., 2008), Apache Hadoop (Shvachko et al., 2010) and Apache Spark (Zaharia et al., 2010). MapReduce breaks down the data into smaller chunks that are more manageable to process, and then aggregates the results to obtain the final prediction. Apache Hadoop is a variant and open-source of the Map Reduce. While Apache Spark processes large data sets faster than MapReduce by using in-memory operations instead of the split-and-merge strategy. Spark can operate on Hadoop, but it's not required. Apache Mahout is a distributed large-scale machine learning library that is open-source and developed by the Apache Foundation organization. It can be executed on Apache Spark or Apache Hadoop platforms and provides various classification model implementations. A framework for linear algebra that is distributed in nature and utilizes the Scala language for programming.

In the context of big data, the issue of class imbalance can be particularly challenging when utilizing the MapReduce framework. This is due to the fact that within this framework, certain difficulties such as small disjuncts and insufficient data can become more pronounced. The study compared three techniques for balancing class imbalance: RUS, ROS, and SMOTE. They used two subsets extracted from the ECBDL14 dataset (*Evolutionary Computation for Big Data and Big learning workshop data mining competition 2014: self-deployment track* 2014), with one subset having 12 million instances and the other with 600000 instances, both with a class imbalance ratio of approximately 49. The study used the Decision Tree learners and also the Random Forest algorithm on Apache Hadoop (MapReduce) and Apache Spark frameworks. The

findings indicated that the SMOTE was outperformed by both Random Under-Sampling (RUS) and Random Over-Sampling (ROS). However, RUS yielded better results with a smaller number of partitions while ROS performed better with a larger number of partitions. In general, models using Apache Spark performed better than those using Hadoop, and the best overall performance was achieved by using ROS.

Tsai et al. (2016) compared three different machine learning frameworks: a single machine framework, a learning framework that is distributed and utilizes data parallelism, and a framework for MapReduce that is based in the cloud. The authors tested the frameworks on four datasets, including two datasets with binary classes: Protein Homology and Breast Cancer. They used SVMs in all three frameworks. For the Breast Cancer data, the MapReduce framework had a lower classification accuracy of 58% compared to the baseline and distributed frameworks which both achieved an accuracy of 99.39% for the Breast Cancer dataset. With the Protein Homology dataset, it was found that three different frameworks yielded comparable classification accuracy of around 99%. However, when the quantity of nodes was raised from 10 to 30, both MapReduce and distributed frameworks experienced a slight decline in accuracy. The study emphasizes that relying solely on accuracy as a measure of classification performance may not provide a complete picture since it does not account for True Negative Rate and True Positive Rate values for SVMs. A more informative comparison could have been made by including Apache Spark in the analysis as it has been shown to outperform MapReduce.

Triguero, Galar, Merino, et al. (2016) proposed to use Evolutionary Under-Sampling (EUS) to address the issue of severe class imbalance in big data. They discovered that EUS was effective in traditional data and applied it to the Apache Spark framework, comparing its performance to their previous implementation using the MapReduce and Apache Hadoop frameworks. The C4.5 decision tree algorithm in both implementations was employed as the base learner, with EUS attempting to strike a balance between reducing training data and enhancing classification performance. The researchers modified the Apache Spark framework to handle majority and minority class instances separately, allowing for a larger quantity of instances belonging to the minority class were preserved within each subset. There are two big datasets: KDD Cup 1999 and ECBDL'14, with three variants of the latter being used, each featuring combinations of two distinct classes of DOS vs. U2R or PRB or R2L. The approximate class imbalance ratios for these datasets were 95, 3450, and 74680 respectively. The results indicated that the Apache Spark framework was found to have reduced execution times and the EUS outperformed RUS despite its longer runtime. However, no comparisons were made between EUS and other commonly used techniques (such as SMOTE or ROS) or cost-sensitive or ensemble methods.

In the scope of online learning with big data streams, high sparsity and class imbalance can present significant challenges. To address these issues, two cost-sensitive algorithms were proposed by D. Wang et al. (2015): CS-FSOL and CS-SSOL (first-order and second-order sparse online learning respectively). These methods were evaluated using public datasets obtained from online sources. The datasets had varying feature set sizes (from 7510 to 16079971) and class ratios as high as 99:1. According to the findings, algorithms that take cost into account performed better than those that do not. Additionally, second-order algorithms were found

to have higher classification accuracy than first-order algorithms. However, further research is needed to assess the effectiveness of these algorithms in other domains and to determine whether classification accuracy is an appropriate performance metric in situations with high class imbalance.

In the ECBDL14 workshop, a ROSEFW-RF algorithm of Triguero, Rio, et al. (2015) won that data mining competition in big data. Their ROSEFW-RF used Random Forest algorithm and combined with Evolutionary Feature Weighting and Random Over-Sampling. This algorithm was able to handle a dataset with 32 million data points and the imbalance ratio is 49. This algorithm employed MapReduce techniques and applied it multiple times to address class imbalance. It has six stages sequentially: (1) ROS Map phase; (2) ROS Reduce phase; (3) RF-BigData Map phase for model building; (4) RF-BigData Map phase for classification; (5) DEFW Map phase; (6) DEFW Reduce phase. Although this dataset is large and highly imbalanced, the study's limitations include the fact that MapReduce technologies may not be effective in all cases.

In the field of big data, engineering techniques used to handle large datasets are typically based on distributed computing on a system of multiple and expensive computers. Similarly, in order to address the imbalance problem in big datasets, researchers have implemented distributed versions of common sampling techniques. For example, Del Rio et al. (2014) used a MapReduce framework to implement under-sampling methods and found that they could effectively handle large datasets. Similarly, Triguero, Rio, et al. (2015) faced an extremely imbalanced big data bioinformatics problem and proposed a complex combination of Random Forest and oversampling to balance the class distribution. Furthermore, in a review by Fernandez et al. (2017) on big data and imbalanced classification, it was found that a perfectly balanced sample is not always the best method, and traditional approaches have been used effectively (Hido et al., 2009; Garcia et al., 2009).

3.4 Recent works on the extreme imbalance in big data classification

Recently, researchers have recognized the challenges of dealing with extremely imbalanced datasets, where the imbalance ratio is greater than 100, i.e. the minority class is less than 0.9901% of the dataset (Tang et al., 2009; Triguero, Rio, et al., 2015). In such scenarios, the positive class is a very small fraction of the dataset, making it difficult to find meaningful patterns among a few positive data points. In the big data context, this problem becomes even more pronounced as the size of the dataset increases. To our knowledge, there are only a few studies that have addressed this gap in extremely imbalanced big data classification. This new scenario raises questions about the effectiveness of traditional methods in dealing with such large and highly imbalanced datasets. In this study, we consider a dataset as extreme imbalance big data if it has an imbalance ratio greater than 100 and contains over 100000 instances. For those datasets with IR greater than 50, we consider them as high imbalance data.

When dealing with large datasets that have a significant imbalance between classes, it is

essential to consider the distribution of both positive and negative classes. Chai et al. (2013) explored the issue of class imbalance in medical record text classification. They used Random Under-Sampling to balance the two classes at an equal ratio and compared classification performance between the original imbalanced dataset and the newly balanced one. The training dataset had 516000 samples and 85650 attributes, with only 0.3% of samples belonging to the positive class (i.e., an imbalance ratio of approximately 333). The authors employed Regularized Logistic Regression as their classifier due to its ability to handle a large number of features without overfitting. According to the results, applying under-sampling led to an increase in Recall and a decrease in Precision. However, the F1 score was relatively unaffected by the use of under-sampling. Furthermore, the study did not provide a clear explanation for why equal class ratios with under-sampling were preferred or why under-sampling was chosen over other data-sampling methods. Additionally, it did not examine how different class ratios affected classification performance. Their study was limited to medical data and did not provide generalizable results for other domains.

Wei et al. (2013) designed a system called i-Alertor to identify fraudulent bank transactions in an imbalanced environment. The system combined contrast pattern mining (L. Wang et al., 2005), Decision Forest algorithm and the cost-sensitive neural networks. In which, L. Wang et al. (2005) designed the contrast pattern mining algorithm to handle the high imbalance of classes, making the process of detecting patterns more efficient. The dataset used in the study had about 8 million data points, 130 attributes and its imbalance ratio is 5330. When compared to a widely used rule-based system employed by banks in Australia, i-Alertor demonstrated superior performance in detecting fraudulent activities. On average, i-Alertor achieved a true positive rate of 0.66. However, the study only compared the system to one another, making it difficult to generalize the findings across different domains.

In the research on enhancing DeepQA - the natural language processing system that utilized the power of IBM Watson - Baughman et al. (2013) employed a combination of data-level and algorithm-level approaches to address the issue of class imbalance. The data-level approach involved manual checking of questions and answers and over-sampling, while the algorithm-level approach utilized one of the most common algorithms (i.e. the regularized logistic regression. The inclusion of a regularization term during the fitting stage was found to mitigate the effects of class imbalance, thus improving DeepQA's ability to assist professionals in making quick and informed decisions. Their study used a dataset with approximately 720000 data points, 400 attributes and had the imbalance ratio roughly 7000. Their experiments indicate that over-sampled logistic regression with regularization outperforms over-sampled logistic regression without regularization in terms of accuracy, with a small improvement. The findings indicate that employing data-level approaches such as verification and oversampling enhances the rate of recall. Nonetheless, the proficiency of the individual performing the verification may constrain the technique suggested by Baughman et al. (2013). For instance, verification conducted by an expert is expected to produce more precise outcomes than that carried out by an undergraduate student. Additionally, other classifiers should also be evaluated, not just logistic regression, because there are many classification algorithms more robust than their algorithm.

Del Rio et al. (2014) evaluated the performance of various methods for addressing class imbalance using the MapReduce framework through Apache Hadoop and Apache Mahout and employing random forests as the base classifier. The techniques assessed included Random Over-Sampling (ROS), Random Under-Sampling (RUS), Synthetic Minority Over-sampling Technique (SMOTE), and a cost-sensitive variant of Random Forests. The datasets utilized in the analysis varied in size from roughly 435,000 to 5,700,000 data points and their imbalance ratios are up to 77670. The results were not conclusive, as no single method was the best performer. In their experiment, the best method depends on the quantity of mappers designated for conducting the experiment, which contradicts the results found in other studies. The primary constraint of this investigation is the applicability of its findings, which could be improved by conducting experiments on more adaptable big data frameworks such as Apache Spark.

D'Addabbo et al. (2015) studied the combination of under-sampling techniques and introduced the Parallel Selective Sampling (PSS) technique. The PSS combines Tomek links (Tomek, 1976) and an under-sampling technique, an adapted nearest-neighbor approach that focuses on points in close proximity to the boundaries. They developed the PSS-SVM algorithm by integrating the PSS into the Support Vector Machines. The datasets utilized in this investigation varied from 25000 to 1000000 data points. The ratios between the majority and minority classes ranged from 13:1 to 200:1. When compared to other classifiers such as SVM, RUS-SVM and RUSBoost, PSS-SVM demonstrated superior performance in terms of both prediction accuracy and processing speed. However, this study only combines with the SVM algorithm, and the outcome would be valuable to explore the impact of integrating PSS with other algorithms such as the Naive Bayes algorithm or the logistic regression algorithm. Additionally, it is crucial to compare PSS-SVM with a variety of hybrid and ensemble methods.

Galpert et al. (2015) conducted research on the problem of class imbalance in the identification of orthologs among various yeast species. Orthologs refer to genes that originate from a common ancestor gene and are present in different species. The study compared several big data approaches such as a cost-sensitive version of the Random Forest algorithm implemented with MapReduce (RF-BDCS), Random Over-Sampling with Random Forest using MapReduce (ROS+RF-BD), and an implementation of Support Vector Machines in Apache Spark combined with MapReduce Random Over-Sampling (ROS+SVM-BD). They used several datasets that ranged from 8 million to 29.9 million instances and all of them had six features. The imbalance ratios ranged from 1630 to 10520. The authors compared the supervised techniques to unsupervised techniques, OMA, Reciprocal Smallest Distance, and Reciprocal Best Hits. The supervised techniques outperformed the unsupervised techniques and ROS+SVM-BD had the best performance with the AUC of 0.885 and the GM of 0.879. The performance of the Support Vector Machine algorithm implemented using Spark's MLLib machine learning library was compared to that of Random Over-Sampling implemented within the Hadoop framework. The most effective method, which combined Random Over-Sampling with Support Vector Machines using both MapReduce and Apache Spark (ROS+SVM-BD), was only evaluated against two other supervised learning methods. The inclusion of the SMOTE algorithm in the comparison would have offered a more comprehensive view.

S.-h. Park et al. (2016) developed a scheme for predicting road traffic accidents using big data analysis with the Apache Hadoop framework. They focused on addressing the issue of extreme imbalance in the dataset of traffic accidents using a MapReduce modification of SMOTE, with an imbalance ratio of approximately 370 and 524131 instances with 14 features. In their study, they increased the positive class (accident cases) from 0.27% to 23.5% after the over-sampling approach. The Logistic Regression algorithm resulted in a 76.35% accuracy and a 40.83% true positive rate. Another study by S. H. Park et al. (2014) also used SMOTE in the Apache Hadoop framework and obtained a 0.806 classification accuracy with a 710 imbalance ratio and 13 predictive features when the positive class roughly reached 30% of the training dataset. Due to the high-class imbalance in datasets, MapReduce may not be the most efficient framework. Alternative frameworks such as Apache Spark may provide better results. When using SMOTE in MapReduce to address high-class imbalance, performance may be suboptimal compared to other methods. Finding the best balance between the accuracy of classification and the rate of over-sampling cannot be achieved through a set formula. Instead, it must be determined through practical observation and experimentation. The study suggests that the best over-sampling rate for class balance can vary greatly.

Maurya (2016) introduces the ImbalancedBayesOpt (IBO) algorithm, a novel approach that optimizes the Matthew's Correlation Coefficient (MCC) to measure the imbalance in binary classification problems. The MCC is calculated using the true positive, true negative, false positive, and false negative values from a confusion matrix. The IBO algorithm uses a Gaussian process to find the best weights for both the positive and negative classes in order to maximize the MCC. Furthermore, the author also introduces the ImbalancedGridOpt (IGO) algorithm that could maximizes MCC by performing a uniform grid search on the weight of the positive data points that are in the minority. Their experiment uses a dataset of manufacturing parts moving through production, with the binary class indicating whether the part passes or fails quality control (failures are the minority class). The dataset in question is highly imbalanced with a class ratio of approximately 171 and comprises 1.18 million instances with 6747 attributes. Both IBO and IGO utilized the Gradient Boosting Machine (GBM) algorithm. Sampling methods were used to create smaller subsets with either equal sampling from both classes (unbiased) or only from the majority class (biased). The results of the comparison between IBO, IGO, and GBM showed that IGO and IBO performed better than GBM for intensely imbalanced datasets, with similar performance in terms of accuracy, precision, recall and MCC values. Additionally, IBO operated at a faster speed than IGO.

Infrequent failure occurrences in extensive manufacturing processes can also create an imbalanced class environment. To investigate this issue, Hebert (2016) compared the performance of Random Forest (RF) and XGBoost (a scalable tree-based classification method) with logistic regression. This study's dataset contained approximately 1180000 instances and 4264 features, with a 170 imbalance ratio. Both tree-based classification methods were found to outperform logistic regression. However, the use of tree-based classifiers has some disadvantages, such as the difficulty in understanding the interactions between parameters in a forest of trees. Additionally, other classifiers, such as k-nearest neighbours and neural networks, could also be used

to compare the performance of linear and non-linear systems.

A study by Zhai et al. (2017) explored the use of integrated Extreme Learning Machine classifiers to address class imbalance in a MapReduce environment using Apache Hadoop. The purpose of ELM classifiers is to train feed-forward neural networks with a single hidden layer and randomly choose the hidden nodes (G.-B. Huang et al., 2006). The datasets in the study had sizes ranging from 1500 to 336000 instances and feature sets ranging from 3 to 16. The ratio of majority to minority classes ranged from 11.5 to 2,140. The new algorithm had four stages: alternating oversampling of positive and negative class instances, creating balanced data subsets, training component classifiers using ELM on the subsets, and combining the classifiers using a voting approach. The results indicated that the ELM algorithm outperformed the SMOTE-Bagging, SMOTE-Boost and SMOTE-Vote classification techniques. Further research to compare the ELM algorithm with a wider range of methods is recommended.

The study by Marchant et al. (2017) evaluated the OASIS method on five Entity Resolution datasets, the training datasets had record-pairs ranging from 20,000 to 676,000 and imbalance ratios between 0.99 and 3,380. The OASIS technique was evaluated against three other techniques: Non-Adaptive Importance Sampling, Stratified Sampling and Passive Sampling. The findings indicated that OASIS outperformed the other methods with the lowest F-measure Absolute Error being 10-5. The study also compared OASIS with other classifiers, such as SVM with a Gaussian kernel, Logistic Regression, AdaBoost and Multilayer Perceptron. They found that OASIS performed better than these classifiers as well. However, the study had some limitations, including the limited size of the feature set, which consisted of only two predictive attributes. This is not common in data mining. Further investigation is needed to validate the performance of OASIS in comparison with other methods and classifiers.

3.5 Methodology

K-Segments Under Bagging (K-SUB)

A comprehensive review of different approaches for handling the class imbalance problem is presented by Galar et al. (2012). The authors conducted a comparison of the most important published methods and found that ensemble-based algorithms, like bagging or boosting, combined with undersampling techniques, tend to achieve better results than other complex methods. However, their study primarily focused on datasets with low degrees of imbalance and limited high imbalance datasets. Nonetheless, their conclusion has been used as a foundation for further research.

In this section, we propose an approach to deal with extreme imbalance in big data by combining undersampling and ensemble learning. To overcome this problem, our approach uses each data point in the majority class only once. It first applies an undersampling technique, where it randomly splits all negative data points into several disjoint subsets of equal size. Then it merges each subset with all positive data points and uses it to train a classifier. Finally, it combines these classifiers by a voting ensemble to obtain the final prediction. This approach is illustrated in Figure 3.2. Unlike existing methods that use repeated sampling or weighting

schemes, our approach ensures that each data point in the majority class contributes equally to the learning process and reduces the risk of overfitting.

Let us consider a dataset D with N data points and an imbalance ratio of IR. We use the symbol K to denote the number of subsets that we divide the dataset into. We define D^{major} and D^{minor} to be the majority and minority classes in the dataset D respectively. To ensure that each data point in the majority class is used only once, we sample each subset to have a size of $\left\lfloor \frac{\mid D^{major} \mid}{K} \right\rfloor$. We primarily employ the Random Forest algorithm for the main part of our approach due to its robustness and strong performance in practical applications. The process is summarized in detail in Algorithm 3.

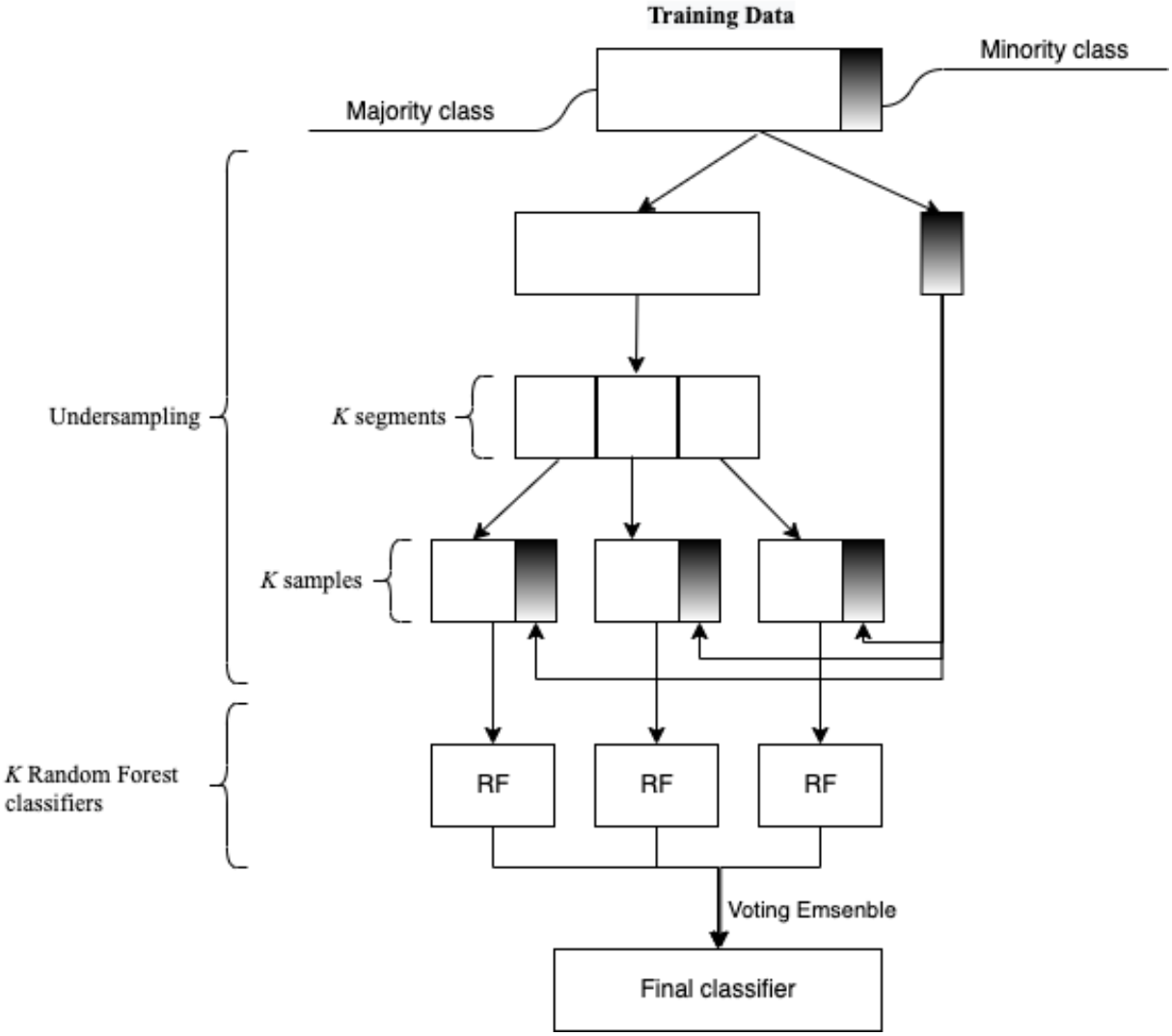


Figure 3.2: The figure illustrates the K-Segments Under Bagging (K-SUB) approach for handling highly imbalanced data. The majority class is split into K segments, and each segment is combined with the whole minority class to create a new sample. This results in K samples with a reduced imbalance ratio of $\frac{IR}{K}$ and smaller data size roughly by $\frac{1}{K}$, allowing for more effective training of the model.

Algorithm 3 K-Segments Under Bagging (K-SUB)

Input:

D^{major} is the majority class in the dataset D
D^{minor} is the minority class in the dataset D
K is the number of segments

Output: An combined model F

for $k = 1 : K$ **do**
  K_k^{major} is drawed without replacement from D^{major} with $\mid K_k^{major} \mid = \left\lfloor \frac{\mid D^{major} \mid}{K} \right\rfloor$
  Let $K_k = K_k^{major} \bigcup D^{minor}$
  Train a classifier f^k by applying the Random Forest algorithm to the subset K_k
end for
Ensemble all f_k into an aggregated model F
Return F

Evaluation measure

In the imbalance problem, many evaluation measures are not meaningful when used independently since they are affected by the majority class. In order to evaluate the classifier, it requires such a measure that is able to balance the classes, and a commonly used measure for this case is the F_1 score, which is derived from the confusion matrix (see Figure 2.6). The F_1 score is the harmonic average of Precision and Recall, and is computed by Equation 3.8. Therefore, in this study, we will use the F_1 score as the main measure for evaluation.

Table 3.4: A confusion matrix is a table that is used to define the performance of a classification algorithm. The table is composed of four different cells, which are true positives, false positives, true negatives, and false negatives. Each of these cells represents the number of times the algorithm predicted a certain outcome and compares it to the actual outcome.

	predicted positives	predicted negatives
true positives	true positive (TP)	false negative (FN)
true negatives	false positive (FP)	true negative (TN)

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.6}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3.7}$$

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{3.8}$$

Datasets

To illustrate the effectiveness of our proposed algorithm, we apply it to 11 different datasets with various aspects, e.g. from small to large datasets and from low to very high imbalance ratios. In each experiment, we also tune the parameter K to achieve the best accuracy. Here, we consider

Table 3.5: Summary of eleven imbalanced datasets and the experimental results.

dataset	IR	sample	feature	UB		K-SUB			RUSBoost	
				F_1	time	F_1	time	K	F_1	time
spectrometer	10	531	93	0.7642	1	**0.8430**	1	5	0.6391	1
isolet	11	7797	617	0.6084	5	**0.7702**	3	3	0.6790	1
libras_move	14	360	90	0.6797	2	**0.7389**	1	3	0.5400	1
webpage	34	344780	300	0.3783	14	**0.4496**	5	10	0.4391	1
mammography	42	11183	6	0.4033	5	**0.6478**	1	5	0.4391	1
protein_homo	111	145751	74	0.4877	24	**0,8125**	7	10	0.6344	11
10% kddcup R2L	443	494021	41	0.2396	643	**0.9350**	46	5	0.8413	133
fraud_detection	577	284807	29	0.2341	148	**0.8113**	30	3	0.3209	59
kdd SF PROBE	7988	703067	30	0.0126	3980	**0.8242**	156	10	0.0351	1857
10% kdd U2R	9499	494021	41	0.0075	7008	**0.6034**	39	5	0.0573	3416
kdd U2R	94199	4940219	41	0.0007	112285	**0.5617**	737	10	0.0061	57291

running a stratified 5-fold cross-validation, then compute the measure F_1 score as an average of five folds and the running time is reported up to only three digits of seconds, which proves that our proposed algorithm is very time-saving. This test is carried out in one single machine with 24 cores and 128 gigabytes of memory. All details of the final results are summarized in Table 3.5.

3.6 Results and discussions

The results in bold indicate the best F_1 scores we can achieve from using the Under Bagging (UB, also known as RUS-Bagging), RUSBoost and our method KSegments Under Bagging (K-SUB) with corresponding tuned K values. We can see that with datasets have low imbalance ratio (i.e. IR < 50), the UB and RUSBoost approaches could handle the problem with acceptable results, while our proposed K-SUB has a little bit better outputs. However, from high to extreme imbalance cases, i.e. IR ≥ 50 and IR ≥ 100 respectively, our method can outperform both the UB and the RUSBoost significantly. Furthermore, the running time is dramatically slow for the case of these two popular methods, but with our method, it only takes us a few minutes in the same machine. Especially in the two biggest datasets with the highest IR, the F_1 scores of the Under Bagging tend to become zero, but our method still achieve the good F_1 scores.

3.7 Conclusion

In this chapter, we have presented the K-Segments Under Bagging approach (K-SUB) in an attempt to tackle the problem of extremely imbalanced data classification. The experimental results show that in the case of extremely imbalanced data, our method not only outperforms the previous method but also runs quickly. While in the case of the low imbalanced data, with the suitable tuned K value, the K-SUB algorithm is still competitive with state-of-the-art methods. With this new and challenging scenario of the extremely imbalanced data, we have shown that the simplified combination of the undersampling technique and the ensemble learning is able to

give better results instead of using other complicated methods addressed in the past. Moreover, it is experimentally seen that we can always tune the parameter K for acceptable accuracy, which is very important in industrial applications. In future work, we suggest to investigate deeper on this approach as well as other related issues regarding to extremely imbalanced and high-dimensional data.

References

Ahumada, Hernán, Guillermo L Grinblat, Lucas C Uzal, Pablo M Granitto, and Alejandro Ceccatto (2008). "REPMAC: A new hybrid approach to highly imbalanced classification problems". In: *2008 Eighth International Conference on Hybrid Intelligent Systems*. IEEE, pp. 386–391 (cit. on p. 64).

Akbani, Rehan, Stephen Kwek, and Nathalie Japkowicz (2004). "Applying support vector machines to imbalanced datasets". In: *Machine Learning: ECML 2004: 15th European Conference on Machine Learning, Pisa, Italy, September 20-24, 2004. Proceedings 15*. Springer, pp. 39–50 (cit. on p. 64).

Ando, Shin and Chun Yuan Huang (2017). "Deep over-sampling framework for classifying imbalanced data". In: *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I 10*. Springer, pp. 770–785 (cit. on p. 54).

Batista, Gustavo EAPA, Ronaldo C Prati, and Maria Carolina Monard (2004). "A study of the behavior of several methods for balancing machine learning training data". In: *ACM Sigkdd Explorations Newsletter* 6.1, pp. 20–29 (cit. on p. 60).

Baughman, Aaron K, Wesley Chuang, Kevin R Dixon, Zachary Benz, and Justin Basilico (2013). "Deepqa jeopardy! gamification: a machine-learning perspective". In: *IEEE transactions on computational intelligence and AI in games* 6.1, pp. 55–66 (cit. on p. 69).

Bekkar, Mohamed and Taklit Akrouf Alitouche (2013). "Imbalanced data learning approaches review". In: *International Journal of Data Mining & Knowledge Management Process* 3.4, p. 15 (cit. on p. 64).

Boichu, Marie, Laurent Menut, Dmitry Khvorostyanov, Lieven Clarisse, Cathy Clerbaux, Solène Turquety, et al. (2013). "Inverting for volcanic SO 2 flux at high temporal resolution using spaceborne plume imagery and chemistry-transport modelling: the 2010 Eyjafjallajökull eruption case study". In: *Atmospheric Chemistry and Physics* 13.17, pp. 8569–8584 (cit. on p. 66).

Bradford, Jeffrey P, Clayton Kunz, Ron Kohavi, Cliff Brunk, and Carla E Brodley (1998). "Pruning decision trees with misclassification costs". In: *European Conference on Machine Learning*. Springer, pp. 131–136 (cit. on pp. 52, 62, 131, 133).

Breiman, Leo (1996). "Bagging predictors". In: *Machine learning* 24.2, pp. 123–140 (cit. on p. 63).

Bucini, Gabriela, Sassan Saatchi, Niall Hanan, Randall B Boone, and Izak Smit (2009). "Woody cover and heterogeneity in the savannas of the Kruger National Park, South Africa". In: *2009 IEEE International Geoscience and Remote Sensing Symposium*. Vol. 4. IEEE, pp. IV–334 (cit. on p. 54).

Buda, Mateusz, Atsuto Maki, and Maciej A Mazurowski (2018). "A systematic study of the class imbalance problem in convolutional neural networks". In: *Neural networks* 106, pp. 249–259 (cit. on p. 54).

Bunkhumpornpat, Chumphol, Krung Sinapiromsaran, and Chidchanok Lursinsap (2009). "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem". In: *Advances in Knowledge Discovery and Data Mining: 13th Pacific-Asia Conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 Proceedings 13*. Springer, pp. 475–482 (cit. on p. 57).

Callut, Jérôme and Pierre Dupont (2005). "F/sub/spl beta//support vector machines". In: *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* Vol. 3. IEEE, pp. 1443–1448 (cit. on p. 61).

Chai, Kevin EK, Stephen Anthony, Enrico Coiera, and Farah Magrabi (2013). "Using statistical text classification to identify health information technology incidents". In: *Journal of the American Medical Informatics Association* 20.5, pp. 980–985 (cit. on pp. 69, 134).

Chan, Philip K, Wei Fan, Andreas L Prodromidis, and Salvatore J Stolfo (1999). "Distributed data mining in credit card fraud detection". In: *IEEE Intelligent Systems and Their Applications* 14.6, pp. 67–74 (cit. on pp. 55, 131).

Chawla, Nitesh, Aleksandar Lazarevic, Lawrence Hall, and Kevin Bowyer (2003). "SMOTEBoost: Improving prediction of the minority class in boosting". In: *Knowledge Discovery in Databases: PKDD 2003*, pp. 107–119 (cit. on p. 63).

Chawla, Nitesh V, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer (2002). "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16, pp. 321–357 (cit. on pp. 52, 57, 131, 133).

Chawla, Nitesh V, Nathalie Japkowicz, and Aleksander Kotcz (2004). "Special issue on learning from imbalanced data sets". In: *ACM Sigkdd Explorations Newsletter* 6.1, pp. 1–6 (cit. on pp. 55, 132).

Chen, Chao, Andy Liaw, and Leo Breiman (2004). "Using random forest to learn imbalanced data". In: *University of California, Berkeley* 110, pp. 1–12 (cit. on pp. 52, 64, 131, 132).

Chen, Sheng and Haibo He (2013). "Nonstationary stream data learning with imbalanced class distribution". In: *Imbalanced learning: Foundations, algorithms, and applications*, pp. 151–186 (cit. on p. 56).

Chen, Xue-wen, Byron Gerlach, and David Casasent (2005). "Pruning support vectors for imbalanced data classification". In: *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*. Vol. 3. IEEE, pp. 1883–1888 (cit. on pp. 52, 131, 132).

Cieslak, David A and Nitesh V Chawla (2008). "Learning decision trees for unbalanced data". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 241–256 (cit. on pp. 52, 60, 131, 133).

D'Addabbo, Annarita and Rosalia Maglietta (2015). "Parallel selective sampling method for imbalanced and large data classification". In: *Pattern Recognition Letters* 62, pp. 61–67 (cit. on p. 70).

Dal Pozzolo, Andrea, Olivier Caelen, Reid A Johnson, and Gianluca Bontempi (2015). "Calibrating probability with undersampling for unbalanced classification". In: *Computational Intelligence, 2015 IEEE Symposium Series on*. IEEE, pp. 159–166 (cit. on p. 57).

Dal Pozzolo, Andrea, Olivier Caelen, Serge Waterschoot, and Gianluca Bontempi (2013). "Racing for unbalanced methods selection". In: *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, pp. 24–31 (cit. on pp. viii, 58).

Davis, Jesse and Mark Goadrich (2006). "The relationship between Precision-Recall and ROC curves". In: *Proceedings of the 23rd international conference on Machine learning*, pp. 233–240 (cit. on p. 65).

Dean, Jeffrey and Sanjay Ghemawat (2008). "MapReduce: simplified data processing on large clusters". In: *Communications of the ACM* 51.1, pp. 107–113 (cit. on p. 66).

Del Rio, Sara, Victoria Lopez, Jose Manuel Benitez, and Francisco Herrera (2014). "On the use of MapReduce for imbalanced big data using Random Forest". In: *Information Sciences* 285, pp. 112–137 (cit. on pp. 52, 68, 70, 131, 133, 134).

Ding, Wan, Dong-Yan Huang, Zhuo Chen, Xinguo Yu, and Weisi Lin (2017). "Facial action recognition using very deep networks for highly imbalanced class distribution". In: *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, pp. 1368–1372 (cit. on p. 54).

Domingos, Pedro (1999). "Metacost: A general method for making classifiers cost-sensitive". In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 155–164 (cit. on pp. 55, 62, 132).

Dong, Qi, Shaogang Gong, and Xiatian Zhu (2018). "Imbalanced deep learning by minority class incremental rectification". In: *IEEE transactions on pattern analysis and machine intelligence* 41.6, pp. 1367–1381 (cit. on p. 54).

Drummond, Chris, Robert C Holte, et al. (2003). "C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling". In: *Workshop on learning from imbalanced datasets II*. Vol. 11. Citeseer Washington DC (cit. on pp. 52, 57, 131, 133).

Elkan, Charles (2001). "The foundations of cost-sensitive learning". In: *International joint conference on artificial intelligence*. Vol. 17. Lawrence Erlbaum Associates Ltd, pp. 973–978 (cit. on p. 62).

Estabrooks, Andrew, Taeho Jo, and Nathalie Japkowicz (2004). "A multiple resampling method for learning from imbalanced data sets". In: *Computational intelligence* 20.1, pp. 18–36 (cit. on pp. 52, 56, 131, 133).

Evolutionary Computation for Big Data and Big learning workshop data mining competition 2014: self-deployment track (2014). `http://cruncher.ico2s.org/bdcomp/`. Accessed 4 Sept 2018 (cit. on p. 66).

Fawcett, Tom and Foster Provost (1997). "Adaptive fraud detection". In: *Data mining and knowledge discovery* 1.3, pp. 291–316 (cit. on pp. 55, 131).

Fernandez, Alberto, Sara del Rio, Nitesh V Chawla, and Francisco Herrera (2017). "An insight into imbalanced Big Data classification: outcomes and challenges". In: *Complex & Intelligent Systems* 3.2, pp. 105–120 (cit. on pp. 52, 68, 131, 133).

Freund, Yoav, Robert E Schapire, et al. (1996). "Experiments with a new boosting algorithm". In: *Icml*. Vol. 96, pp. 148–156 (cit. on p. 63).

Galar, Mikel, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera (2012). "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42.4, pp. 463–484 (cit. on pp. 64, 72).

Galpert, Deborah, Sara Del Río, Francisco Herrera, Evys Ancede-Gallardo, Agostinho Antunes, and Guillermin Agüero-Chapin (2015). "An effective big data supervised imbalanced classification approach for ortholog detection in related yeast species". In: *BioMed research international* 2015 (cit. on pp. 70, 134).

Garcia, Salvador and Francisco Herrera (2009). "Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy". In: *Evolutionary computation* 17.3, pp. 275–306 (cit. on p. 68).

Guo, Hongyu and Herna L Viktor (2004). "Learning from imbalanced data sets with boosting and data generation: the databoost-im approach". In: *ACM Sigkdd Explorations Newsletter* 6.1, pp. 30–39 (cit. on p. 63).

Han, Hui, Wen-Yuan Wang, and Bing-Huan Mao (2005). "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning". In: *Advances in intelligent computing*, pp. 878–887 (cit. on p. 57).

Hart, Peter (1968). "The condensed nearest neighbor rule (Corresp.)" In: *IEEE transactions on information theory* 14.3, pp. 515–516 (cit. on p. 59).

He, Haibo and Edwardo A Garcia (2009). "Learning from imbalanced data". In: *IEEE Transactions on knowledge and data engineering* 21.9, pp. 1263–1284 (cit. on pp. 60, 65).

Hebert, Jeff (2016). "Predicting rare failure events using classification trees on large scale manufacturing data with complex interactions". In: *2016 IEEE international conference on big data (big data)*. IEEE, pp. 2024–2028 (cit. on pp. 71, 134).

Hido, Shohei, Hisashi Kashima, and Yutaka Takahashi (2009). "Roughly balanced bagging for imbalanced data". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 2.5-6, pp. 412–426 (cit. on p. 68).

Holte, Robert C, Liane Acker, Bruce W Porter, et al. (1989). "Concept Learning and the Problem of Small Disjuncts." In: *IJCAI*. Vol. 89, pp. 813–818 (cit. on p. 56).

Hong, Xia, Sheng Chen, and Chris J Harris (2007). "A kernel-based two-class classifier for imbalanced data sets". In: *IEEE Transactions on neural networks* 18.1, pp. 28–41 (cit. on pp. 52, 131, 132).

Huang, Chen, Yining Li, Chen Change Loy, and Xiaoou Tang (2016). "Learning deep representation for imbalanced classification". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5375–5384 (cit. on p. 54).

Huang, Guang-Bin, Qin-Yu Zhu, and Chee-Kheong Siew (2006). "Extreme learning machine: theory and applications". In: *Neurocomputing* 70.1-3, pp. 489–501 (cit. on p. 72).

Japkowicz, Nathalie (2001). "Concept-learning in the presence of between-class and within-class imbalances". In: *Advances in Artificial Intelligence: 14th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2001 Ottawa, Canada, June 7–9, 2001 Proceedings 14*. Springer, pp. 67–77 (cit. on p. 56).

— (2003). “Class imbalances: are we focusing on the right issue”. In: *Workshop on learning from imbalanced data sets II*. Vol. 1723, p. 63 (cit. on p. 56).

Japkowicz, Nathalie and Shaju Stephen (2002). “The class imbalance problem: A systematic study”. In: *Intelligent data analysis* 6.5, pp. 429–449 (cit. on pp. 56, 60).

Jo, Taeho and Nathalie Japkowicz (2004). “Class imbalances versus small disjuncts”. In: *ACM Sigkdd Explorations Newsletter* 6.1, pp. 40–49 (cit. on p. 56).

Joshi, Mahesh V, Vipin Kumar, and Ramesh C Agarwal (2001). “Evaluating boosting algorithms to classify rare classes: Comparison and improvements”. In: *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*. IEEE, pp. 257–264 (cit. on p. 63).

Juszczak, Piotr, Niall M Adams, David J Hand, Christopher Whitrow, and David J Weston (2008). “Off-the-peg and bespoke classifiers for fraud detection”. In: *Computational Statistics & Data Analysis* 52.9, pp. 4521–4532 (cit. on pp. 52, 131, 133).

Kang, Pilsung and Sungzoon Cho (2006). “EUS SVMs: Ensemble of under-sampled SVMs for data imbalance problems”. In: *Neural Information Processing*. Springer, pp. 837–846 (cit. on p. 63).

Khan, Salman H, Munawar Hayat, Mohammed Bennamoun, Ferdous A Sohel, and Roberto Togneri (2017). “Cost-sensitive learning of deep feature representations from imbalanced data”. In: *IEEE transactions on neural networks and learning systems* 29.8, pp. 3573–3587 (cit. on p. 54).

Khoshgoftaar, Taghi M, Moiz Golawala, and Jason Van Hulse (2007). “An empirical study of learning from imbalanced data using random forest”. In: *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*. Vol. 2. IEEE, pp. 310–317 (cit. on p. 64).

Klement, William, Peter Flach, Nathalie Japkowicz, and Stan Matwin (2009). “Cost-Based Sampling of Individual Instances”. In: *Canadian Conference on Artificial Intelligence*. Springer, pp. 86–97 (cit. on p. 58).

Krawczyk, Bartosz (2016). “Learning from imbalanced data: open challenges and future directions”. In: *Progress in Artificial Intelligence* 5.4, pp. 221–232 (cit. on pp. 52, 62, 63, 131, 133).

Kubat, Miroslav, Stan Matwin, et al. (1997). “Addressing the curse of imbalanced training sets: one-sided selection”. In: *ICML*. Vol. 97. Nashville, USA, pp. 179–186 (cit. on pp. 59, 60).

Laurikkala, Jorma (2001). “Improving identification of difficult small classes by balancing class distribution”. In: *Artificial Intelligence in Medicine*, pp. 63–66 (cit. on pp. 52, 56, 131, 133).

Lee, Hansang, Minseok Park, and Junmo Kim (2016). “Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning”. In: *2016 IEEE international conference on image processing (ICIP)*. IEEE, pp. 3713–3717 (cit. on p. 54).

Li, Xuchun, Lei Wang, and Eric Sung (2008). “AdaBoost with SVM-based component classifiers”. In: *Engineering Applications of Artificial Intelligence* 21.5, pp. 785–795 (cit. on p. 61).

Liao, Yu-Lin, Che-Cheng Kuo, and Ya-Fu Peng (2010). “Prediction and identification using recurrent wavelet-based cerebellar model articulation controller neural networks”. In: *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–6 (cit. on p. 54).

Lin, Tsung-Yi, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár (2017). "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988 (cit. on p. 54).

Ling, Charles X and Victor S Sheng (2008). "Cost-sensitive learning and the class imbalance problem". In: *Encyclopedia of machine learning* 2011, pp. 231–235 (cit. on p. 62).

Ling, Charles X, Qiang Yang, Jianning Wang, and Shichao Zhang (2004). "Decision trees with minimal costs". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM, p. 69 (cit. on pp. 52, 62, 131, 133).

Liu, Bing, Wynne Hsu, and Yiming Ma (1999). "Mining association rules with multiple minimum supports". In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 337–341 (cit. on p. 61).

Liu, Wei and Sanjay Chawla (2011). "Class confidence weighted k NN algorithms for imbalanced data sets". In: *Advances in Knowledge Discovery and Data Mining: 15th Pacific-Asia Conference, PAKDD 2011, Shenzhen, China, May 24-27, 2011, Proceedings, Part II 15*. Springer, pp. 345–356 (cit. on p. 61).

Liu, Yang, Aijun An, and Xiangji Huang (2006). "Boosting Prediction Accuracy on Imbalanced Datasets with SVM Ensembles." In: *PAKDD*. Vol. 6. Springer, pp. 107–118 (cit. on p. 63).

Liu, Xu-Ying, Jianxin Wu, and Zhi-Hua Zhou (2008). "Exploratory undersampling for class-imbalance learning". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.2, pp. 539–550 (cit. on pp. 54, 63).

López, Victoria, Alberto Fernández, Jose G Moreno-Torres, and Francisco Herrera (2012). "Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics". In: *Expert Systems with Applications* 39.7, pp. 6585–6608 (cit. on p. 62).

Maloof, Marcus, Pat Langley, Stephanie Sage, and T Binford (1997). "Learning to detect rooftops in aerial images". In: *Proceedings of the Image Understanding Workshop*, pp. 835–845 (cit. on p. 62).

Mani, Inderjeet and I Zhang (2003). "kNN approach to unbalanced data distributions: a case study involving information extraction". In: *Proceedings of workshop on learning from imbalanced datasets*. Vol. 126 (cit. on p. 60).

Marchant, Neil G and Benjamin IP Rubinstein (2017). "In search of an entity resolution oasis: optimal asymptotic sequential importance sampling". In: *arXiv preprint arXiv:1703.00617* (cit. on p. 72).

Masko, David and Paulina Hensman (2015). *The impact of imbalanced training data for convolutional neural networks* (cit. on p. 54).

Maurya, Abhinav (2016). "Bayesian optimization for predicting rare internal failures in manufacturing processes". In: *2016 IEEE international conference on big data (big data)*. IEEE, pp. 2036–2045 (cit. on p. 71).

Mease, David, Abraham J Wyner, and Andreas Buja (2007). "Boosted classification trees and class probability/quantile estimation". In: *Journal of Machine Learning Research* 8.Mar, pp. 409–439 (cit. on p. 63).

Mekterović, Igor, Mladen Karan, Damir Pintar, and Ljiljana Brkić (2021). "Credit card fraud detection in card-not-present transactions: Where to invest?" In: *Applied Sciences* 11.15, p. 6766 (cit. on pp. 52, 131, 133).

Park, Seong Hun and Young Guk Ha (2014). "Large imbalance data classification based on mapreduce for traffic accident prediction". In: *2014 Eighth international conference on innovative mobile and internet services in Ubiquitous computing*. IEEE, pp. 45–49 (cit. on p. 71).

Park, Seong-hun, Sung-min Kim, and Young-guk Ha (2016). "Highway traffic accident prediction using VDS big data analysis". In: *The Journal of Supercomputing* 72, pp. 2815–2831 (cit. on p. 71).

Pouyanfar, Samira, Yudong Tao, Anup Mohan, Haiman Tian, Ahmed S Kaseb, Kent Gauen, et al. (2018). "Dynamic sampling in convolutional neural networks for imbalanced data classification". In: *2018 IEEE conference on multimedia information processing and retrieval (MIPR)*. IEEE, pp. 112–117 (cit. on p. 54).

Prati, Ronaldo C, Gustavo EAPA Batista, and Maria Carolina Monard (2004). "Class imbalances versus class overlapping: an analysis of a learning system behavior". In: *MICAI 2004: Advances in Artificial Intelligence: Third Mexican International Conference on Artificial Intelligence, Mexico City, Mexico, April 26-30, 2004. Proceedings 3*. Springer, pp. 312–321 (cit. on p. 56).

Provost, Foster (2000). "Machine learning from imbalanced data sets 101". In: *Proceedings of the AAAI'2000 workshop on imbalanced data sets*, pp. 1–3 (cit. on p. 60).

Provost, Foster J, Tom Fawcett, et al. (1997). "Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions." In: *KDD*. Vol. 97, pp. 43–48 (cit. on p. 65).

Quinlan, J Ross (2014). *C4. 5: programs for machine learning*. Elsevier (cit. on pp. 13, 60).

Rokach, Lior and Oded Maimon (2005). "Top-down induction of decision trees classifiers-a survey". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 35.4, pp. 476–487 (cit. on p. 56).

Seiffert, Chris, Taghi M Khoshgoftaar, Jason Van Hulse, and Amri Napolitano (2009). "RUSBoost: A hybrid approach to alleviating class imbalance". In: *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 40.1, pp. 185–197 (cit. on p. 64).

Seliya, Naeem, Taghi M Khoshgoftaar, and Jason Van Hulse (2009). "A study on the relationships of classifier performance metrics". In: *2009 21st IEEE international conference on tools with artificial intelligence*. IEEE, pp. 59–66 (cit. on p. 66).

Shvachko, Konstantin, Hairong Kuang, Sanjay Radia, and Robert Chansler (2010). "The hadoop distributed file system". In: *2010 IEEE 26th symposium on mass storage systems and technologies (MSST)*. Ieee, pp. 1–10 (cit. on p. 66).

Suman, Sanjeev, Kamlesh Laddhad, and Unmesh Deshmukh (2005). "Methods for Handling Highly Skewed Datasets". In: *Part I-October* 3 (cit. on pp. 59, 60).

Sun, Yanmin, Mohamed S Kamel, Andrew KC Wong, and Yang Wang (2007). "Cost-sensitive boosting for classification of imbalanced data". In: *Pattern recognition* 40.12, pp. 3358–3378 (cit. on p. 63).

Tang, Yuchun, Yan-Qing Zhang, Nitesh V Chawla, and Sven Krasser (2008). "SVMs modeling for highly imbalanced classification". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.1, pp. 281–288 (cit. on p. 64).

— (2009). "SVMs modeling for highly imbalanced classification". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39.1, pp. 281–288 (cit. on pp. 68, 133).

Tomek, Ivan (1976). "Two modifications of CNN". In: *IEEE Trans. Systems, Man and Cybernetics* 6, pp. 769–772 (cit. on pp. 59, 70).

Triguero, Isaac, Mikel Galar, D Merino, Jesus Maillo, Humberto Bustince, and Francisco Herrera (2016). "Evolutionary undersampling for extremely imbalanced big data classification under apache spark". In: *2016 IEEE congress on evolutionary computation (CEC)*. IEEE, pp. 640–647 (cit. on p. 67).

Triguero, Isaac, Mikel Galar, Sarah Vluymans, Chris Cornelis, Humberto Bustince, Francisco Herrera, et al. (2015). "Evolutionary undersampling for imbalanced big data classification". In: *Evolutionary Computation (CEC), 2015 IEEE Congress on*. IEEE, pp. 715–722 (cit. on pp. 52, 131, 133).

Triguero, Isaac, Sara del Rio, Victoria Lopez, Jaume Bacardit, Jose M Benitez, and Francisco Herrera (2015). "ROSEFW-RF: the winner algorithm for the ECBDL'14 big data competition: an extremely imbalanced big data bioinformatics problem". In: *Knowledge-Based Systems* 87, pp. 69–79 (cit. on pp. 52, 68, 131, 133).

Tsai, Chih-Fong, Wei-Chao Lin, and Shih-Wen Ke (2016). "Big data mining with parallel computing: A comparison of distributed and MapReduce methodologies". In: *Journal of Systems and Software* 122, pp. 83–92 (cit. on p. 67).

Van Hulse, Jason, Taghi M Khoshgoftaar, and Amri Napolitano (2007). "Experimental perspectives on learning from imbalanced data". In: *Proceedings of the 24th international conference on Machine learning*, pp. 935–942 (cit. on p. 56).

Verhein, Florian and Sanjay Chawla (2007). "Using significant, positively associated and relatively class correlated rules for associative classification of imbalanced datasets". In: *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. IEEE, pp. 679–684 (cit. on p. 61).

Vilariño, Fernando, Panagiota Spyridonos, Jordi Vitrià, and Petia Radeva (2005). "Experiments with SVM and stratified sampling with an imbalanced problem: detection of intestinal contractions". In: *Pattern Recognition and Image Analysis*, pp. 783–791 (cit. on p. 63).

Visa, Sofia and Anca Ralescu (2005). "Issues in mining imbalanced data sets-a review paper". In: *Proceedings of the sixteen midwest artificial intelligence and cognitive science conference*. Vol. 2005. sn, pp. 67–73 (cit. on p. 60).

Wang, Benjamin X and Nathalie Japkowicz (2010). "Boosting support vector machines for imbalanced data sets". In: *Knowledge and Information Systems* 25.1, pp. 1–20 (cit. on p. 63).

Wang, BX and Nathalie Japkowicz (2004). "Imbalanced data set learning with synthetic samples". In: *Proc. IRIS Machine Learning Workshop*. Vol. 19 (cit. on p. 57).

Wang, Dayong, Pengcheng Wu, Peilin Zhao, and Steven CH Hoi (2015). "A framework of sparse online learning and its applications". In: *arXiv preprint arXiv:1507.07146* (cit. on p. 67).

Wang, Juanjuan, Mantao Xu, Hui Wang, and Jiwu Zhang (2006). "Classification of imbalanced data by using the SMOTE algorithm and locally linear embedding". In: *Signal Processing, 2006 8th International Conference on*. Vol. 3. IEEE (cit. on pp. 52, 131, 132).

Wang, Lusheng, Hao Zhao, Guozhu Dong, and Jianping Li (2005). "On the complexity of finding emerging patterns". In: *Theoretical Computer Science* 335.1, pp. 15–27 (cit. on p. 69).

Wang, Shuhao, Cancheng Liu, Xiang Gao, Hongtao Qu, and Wei Xu (2017). "Session-based fraud detection in online e-commerce transactions using recurrent neural networks". In: *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part III 10*. Springer, pp. 241–252 (cit. on pp. 52, 131, 133).

Wang, Shuo, Ke Tang, and Xin Yao (2009). "Diversity exploration and negative correlation learning on imbalanced data sets". In: *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*. IEEE, pp. 3259–3266 (cit. on p. 63).

Wei, Wei, Jinjiu Li, Longbing Cao, Yuming Ou, and Jiahang Chen (2013). "Effective detection of sophisticated online banking fraud on extremely imbalanced data". In: *World Wide Web* 16, pp. 449–475 (cit. on p. 69).

Weiss, Gary M (1995). "Learning with rare cases and small disjuncts". In: *Machine Learning Proceedings 1995*. Elsevier, pp. 558–565 (cit. on p. 56).

— (2004). "Mining with rarity: a unifying framework". In: *ACM Sigkdd Explorations Newsletter* 6.1, pp. 7–19 (cit. on p. 56).

— (2013). "Foundations of imbalanced learning". In: *Imbalanced Learning: Foundations, Algorithms, and Applications*, pp. 13–41 (cit. on p. 61).

Weiss, Gary M and Foster Provost (2001). "The effect of class distribution on classifier learning: an empirical study". In: *Rutgers Univ* (cit. on pp. 52, 56, 131, 133).

Weng, Cheng G and Josiah Poon (2008). "A new evaluation measure for imbalanced datasets". In: *Proceedings of the 7th Australasian Data Mining Conference-Volume 87*, pp. 27–32 (cit. on p. 65).

Wilson, D Randall and Tony R Martinez (2000). "Reduction techniques for instance-based learning algorithms". In: *Machine learning* 38.3, pp. 257–286 (cit. on p. 59).

Wilson, Dennis L (1972). "Asymptotic properties of nearest neighbor rules using edited data". In: *IEEE Transactions on Systems, Man, and Cybernetics* 3, pp. 408–421 (cit. on pp. 59, 62).

Wu, Gang and Edward Y Chang (2003). "Class-boundary alignment for imbalanced dataset learning". In: *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, pp. 49–56 (cit. on p. 60).

Yan, Rong, Yan Liu, Rong Jin, and Alex Hauptmann (2003). "On predicting rare classes with SVM ensembles in scene classification". In: *Acoustics, Speech, and Signal Processing, 2003.*

Proceedings.(ICASSP'03). 2003 IEEE International Conference on. Vol. 3. IEEE, pp. III–21 (cit. on p. 60).

Yang, Wensi, Yuhang Zhang, Kejiang Ye, Li Li, and Cheng-Zhong Xu (2019). "Ffd: A federated learning based method for credit card fraud detection". In: *Big Data–BigData 2019: 8th International Congress, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, June 25–30, 2019, Proceedings 8*. Springer, pp. 18–32 (cit. on pp. 52, 131, 133).

Zadrozny, Bianca, John Langford, and Naoki Abe (2003). "Cost-sensitive learning by cost-proportionate example weighting". In: *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, pp. 435–442 (cit. on pp. 55, 58, 132).

Zaharia, Matei, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica (2010). "Spark: Cluster computing with working sets." In: *HotCloud* 10.10-10, p. 95 (cit. on p. 66).

Zhai, Junhai, Sufang Zhang, and Chenxi Wang (2017). "The classification of imbalanced large data sets based on MapReduce and ensemble of ELM classifiers". In: *International Journal of Machine Learning and Cybernetics* 8, pp. 1009–1017 (cit. on p. 72).

Zhang, Chong, Kay Chen Tan, and Ruoxu Ren (2016). "Training cost-sensitive deep belief networks on imbalance data problems". In: *2016 international joint conference on neural networks (IJCNN)*. IEEE, pp. 4362–4367 (cit. on p. 54).

Zhang, Xinwei, Yaoci Han, Wei Xu, and Qili Wang (2021). "HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture". In: *Information Sciences* 557, pp. 302–316 (cit. on pp. 52, 131, 133).

Zhang, Yulu, Liguo Shuai, Yali Ren, and Huilin Chen (2018). "Image classification with category centers in class imbalance situation". In: *2018 33rd youth academic annual conference of Chinese association of automation (YAC)*. IEEE, pp. 359–363 (cit. on p. 54).

Chapter 4

Voting Ensemble for linear Shrinkage Covariance Matrix Estimations in the Portfolio Optimization

Objectives

Reducing errors of covariance matrix estimation plays a very important role in many optimization problems, e.g., portfolio optimization. In this chapter, we propose a data-driven approach which basically combines the original framework of a popular approach named shrinkage estimation and cross-validation technique to adapt the shrinkage intensity with different levels of uncertainty of real data. Particularly, this approach can be applied well to asset management to enhance the quality of portfolio selection which is known as a huge area of research in modern portfolio theory. Experimental results carried out using the prices and volumes data of the Vietnamese stock market show that our proposed method can practically improve the quality and robustness of portfolio selection. Last but not least, we introduce an automatic backtesting system that can help us evaluate the portfolio based on various financial indicators in a real-time manner.

Contents

4.1 Introduction

In quantitative finance, portfolio selection is a framework for picking the best portfolio among all feasible portfolios according to a predefined objective which can be modelled mathematically. One of the well-known objectives is minimum variance, in which we try to model the portfolio selection problem as an optimization problem that minimizes the risk of our selection. This kind of portfolio is suitable for investors who want to seek for such a strategy having the lowest risk with a given level of return. However, practically proved, it often leads to an outstanding return over long-term periods (Baker et al., 2011). Together with the rising advantages of computer power, plenty of research in quantitative finance which use the combined framework of mathematics and data science techniques progressively become an attractive trend in portfolio selection (Deboeck, 1994; Y. Li et al., 2016).

Fundamentally, the core of Minimum Variance portfolio research relies solely on how to estimate reliably the covariance matrix. However, almost traditional approaches of covariance matrix estimation such as using the sample covariance matrix (SCM) and ordinary least squares (OLS) face with many technical problems in the case of high-dimensional portfolio selection. Having large dimensionality means that it is easier to get unexpected and uncontrollable errors in some computational steps, and the sample data may not be adequate for the estimation of the true covariance matrix. These factors make the estimated covariance matrix become ill-conditioned or even singular, which is a very popular limitation in matrix computation research. Consequently, the portfolios selected from considering the sample covariance matrix often perform poorly and fail in generating profit. To solve this problem systematically, there's a brilliant research direction of shrinkage estimators for covariance matrix estimation. The general idea of this approach is to pose a control on the tradeoff between the bias and the variance of covariance matrix estimation via considering a pre-defined and well-conditioned target matrix $\mathbf{F}$ and a shrinkage intensity δ that can be computed from sample covariance matrix and $\mathbf{F}$ mathematically. However, the main research on the improvement of shrinkage methodology until now mostly focuses on the mathematical point of view (i.e., mathematical formulation for shrinkage estimation), but there are very few ones starting from a data-driven manner. The purpose of this chapter is to propose an adaptive method to improve shrinkage intensity selection by a combination of the original shrinkage framework with the leave-one-out cross-validation technique. In addition, we design and implement careful experiments to illustrate the effectiveness of our proposed algorithm in portfolio selection using the data from the Vietnam stock market.

This chapter includes five sections which are organized as follows: We start with the introduction and our motivation in Section 4.1. Then move to a brief summary of related works in Section 4.2. For the main content described in Section 4.3, we will sequentially present the original shrinkage framework, the main evaluation measure and our proposed method. The illustrative experiments will be provided in Section 4.4. Finally, we will summarize our contribution to the conclusion in Section 4.5.

4.2 Related works

In machine learning, Cross-Validation (CV) is a common technique to evaluate a model approach on out-of-sample data and then incur a risk generated on testing data. Its applications are broadly in various domains, such as variable selection or parameter tuning. Particularly in finance and economics, the CV technique has been gradually applied in both research and application. A very early study in stock trading (Conway et al., 1988) tried to use CV to stabilize stock returns' factor structure via identifying first a smaller number of stable factors than the popular likelihood ratio test, then suggesting one dominant factor. After that, Upton (1994) proposed an improved version in order to analyze the results of those factors in the sense of statistics and economics. One of the main goals of CV is to test the out-of-sample accuracy of the predictive models, for example, Picard et al. (1984) used CV for assessments of predictive ability of regression models. Recently, L. Zhang (2012) studied a critical nonparametric relationship between implied and realized volatilities by estimating quadratic variation via using CV. In an important survey, Arlot et al. (2010) reviewed many existing results on model selection performances of cross-validation procedures, and related them to the most advanced results of model selection theory. They also indicated the most promising direction for future research regarding precise quantitative measures of the variance of CV estimators. For an example of CV application in time series forecasting, Bergmeir et al. (2018) concentrated on evaluating Autoregressive time series prediction and showed that the use of a normal K-fold CV procedure is possible and useful if the residuals of their considered model are uncorrelated. By doing an experiment in real-world data, they indicate that CV can adequately control the overfitting issue.

In another story, in quantitative finance, reliably estimating covariance matrix is an important step in many econometric and financial applications, in particular, portfolio selection. For instance, in the Markowitz portfolio selection framework, when the number of assets is the same as the number of data-points, it leads to a big issue in many computational steps. In fact, there are a lot of useful approaches have been proposed to reduce the estimation error of the covariance matrix, for example, the approach of shrinkage estimators. It should be figured out the work of Golosnoy and Okhrin (2007) which presented a multivariate shrinkage approach for shrinking the weights of Markowitz-based portfolio to the non-stochastic target weights. In the next research of Golosnoy and Okhrin (2009), they proposed the shrinkage estimator, called flexible shrinkage, which can adjust the portfolio's structure dynamically. In another research direction, Olivier Ledoit et al. (2003), Olivier Ledoit et al. (2004b), and Olivier Ledoit et al. (2004a) promoted the linear shrinkage approach for a class of large-dimensional covariance matrices by shrinking the sample covariance matrix toward well-structured matrices. Due to its reasonable and strong results, it becomes a benchmark for portfolio selection in recent years. Additionally, inheriting the potentials of Random Matrix Theory, they extended their previous works in a next study by using nonlinear transformations of the eigenvalues taking into account solely the sample data (Olivier Ledoit et al., 2010). As an extension, DeMiguel, Martin-Utrera, et al. (2013) gave a thorough review of shrinkage methods in portfolio optimization and studied

a new class of shrinkage techniques for the vector of means, the covariance matrix and also the weights in the portfolio. To complement for this research interest, Candelon et al. (2012) introduced a double shrinkage method to improve the stability of estimation on small sample sizes via using ridge regression to shrink the weights towards the equally-weighted portfolio. However, as mentioned above, generally there is still a lack of research which leverages the CV approach to enhance the analysis of portfolio selection.

4.3 Our proposed approach

4.3.1 Shrinkage Intensity in Covariance Estimation

There are different ways to select a good asset portfolio and one of the common ways in modern portfolio theory is to consider a global minimum variance portfolio with a universe of N stocks with the weights of stocks $\mathbf{w} = (w_1, w_2, ..., w_N)$. The sum of stocks weight must equal one ($\Sigma_{i=1}^{N} w_i = 1$), with the conditions of $w_i > 0$ means there is no short selling. The problem of portfolio selection is defined as:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^T \boldsymbol{\Sigma} \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^T \mathbf{1} = 1 \\ & w_i > 0 \quad \forall i = \overline{1, N} \end{aligned} \tag{4.1}$$

where $\mathbf{1}$ denotes a vector of ones, and $\boldsymbol{\Sigma}$ is the covariance matrix of prices of N stocks. The feasible analytical solution of the problem (4.1) is:

$$\mathbf{w}_* = \boldsymbol{\Sigma}^{-1} \mathbf{1} \left(\mathbf{1}^T \boldsymbol{\Sigma}^{-1} \mathbf{1} \right)^{-1}. \tag{4.2}$$

As we can see, the solution (4.2) relates to the inverse of a covariance matrix. The common method to compute $\mathbf{w}_*$ numerically (denoted $\hat{\mathbf{w}}_*$) is to use the sample covariance matrix $\mathbf{S}$ (can be computed directly from data) other than the true covariance matrix $\boldsymbol{\Sigma}$ (which is unknown). However, this approach often gets into trouble technically regarding the computational aspect, especially in the high-dimensional portfolios since the sample covariance is typically not well-conditioned and may not even be invertible.

One of the reliable methods to estimate $\boldsymbol{\Sigma}$ is based on the shrinkage estimators proposed by Olivier Ledoit et al. (2003). In their studies, they presented a general formula for the shrinkage estimators as follows:

$$\hat{\boldsymbol{\Sigma}} = \delta \mathbf{F} + (1 - \delta) \mathbf{S} \tag{4.3}$$

where $\delta \in [0, 1]$ is the shrinkage intensity, and $\mathbf{F}$ is target matrix. In formula (4.3), $\mathbf{F}$ can be interpreted as the belief of domain experts about the shape of the true covariance matrix, while the parameter δ is a kind of balanced factor which reflects the tradeoff of estimation errors and target's bias. Once we can derive a good couple $(\mathbf{F}, \delta)$, it can be existed an invertible matrix $\hat{\boldsymbol{\Sigma}}$, and then we can rewrite a feasible numerical solution for the problem (4.1) as follow:

$$\hat{\mathbf{w}}_* = \hat{\mathbf{\Sigma}}^{-1}\mathbf{1}\left(\mathbf{1}^T\hat{\mathbf{\Sigma}}^{-1}\mathbf{1}\right)^{-1}. \tag{4.4}$$

It is easy to see that we can have different versions of shrinkage estimation if we change $\mathbf{F}$. In the series of Ledoit and Wolf's studies (Olivier Ledoit et al., 2003; Olivier Ledoit et al., 2004b; Olivier Ledoit et al., 2004a), they respectively consider three popular (and probably the most effective) versions for target matrices named Shrinkage towards market (**SSIM**), Shrinkage towards constant correlation (**SCCM**) and Shrinkage towards identity matrix (**STIM**). Based on these targets, they derived the corresponding explicit formulas for estimating δ. In this chapter, we propose a data-driven approach to adapt shrinkage intensity δ with the fluctuation of data, which we call voting shrinkage estimator.

4.3.2 Main evaluation measure

Normally, in order to carry out the voting, we need a good measurement to evaluate how well the shrinkage estimators are, and among many well-known measures for portfolio selection, we count on the Sharpe ratio as our main evaluation measure in this chapter. The Sharpe ratio (SR), introduced by Sharpe (1964), measures the difference (by ratio) between the excess return (return after subtracting the risk-free rate) and risk. The general formula for the Sharpe ratio is defined as:

$$SR = \frac{R - R_f}{\sigma} \tag{4.5}$$

where the return R and volatility σ of a portfolio are calculated by the following formulas:

$$R = ((\mathbf{R} + 1) \cdot \mathbf{1})^{\frac{252}{|\mathbf{R}|}} - 1, \tag{4.6}$$

$$\sigma = \sqrt{252\text{Var}[\mathbf{R}]}, \tag{4.7}$$

with $\mathbf{R}$ is the portfolio's daily return vector. Here the number of trading days of a year is 252, which means the annualization factor for standardizing the result. Moreover, R_f is the risk-free rate for the evaluated period [1]. It should be noted that, in finance, the Sharpe ratio characterizes how well one is compensated for the risk taken. Thus, a higher Sharpe ratio is generally sought, as this implies more return per risk taken. This is a strong measure under the assumption that volatility (standard deviation) as a good proxy for risk holds true. Due to this strong property of the Sharpe ratio, we consider employing it in our algorithm as the voting standard.

4.3.3 Voting algorithm for Shrinkage Intensity selection

As mentioned above, for each version of target matrix $\mathbf{F}$ we have a corresponding version of shrinkage estimation, and there are three popular well-structured target matrices in shrinkage

[1] To simplify the computation in our algorithm, we assume the risk-free rate is zero

estimation investigated by Ledoit and Wolf (**SSIM**, **SCCM** and **STIM**). A key of the Ledoit-Wolf approach is to calculate the shrinkage intensity from the sample covariance matrix and the target matrix. In order to determine the shrinkage intensity which adapts to the out-of-sample data, we propose a modified version of Ledoit-Woft techniques, called voting shrinkage intensity estimation based on leave-one-out cross-validation, which is the common validation process in machine learning research (Celisse et al., 2008; Shao, 1993). The details of our data-driven approach are as below.

Given a fixed timepoint t_0, we denote the price data of N assets $[D(t)]_{(V+W+1)\times N}$ for $t \in [t_0 - V - W, t_0]$, with the sliding window of length W and the validation window of length V. Suppose that we have $M_{\mathbf{F}}$ versions of shrinkage estimators, the main idea of our algorithm is to select among $M_{\mathbf{F}}$ V-dimentional vectors of shrinkage intensities for the "best" one which makes the Sharpe ratio (taken into account a period of length V) achieve the maximum. Then, the algorithm returns the average of these shrinkage intensities as voting intensity, which is employed to build the present portfolio.

An intuitive visualization of our algorithm is shown in Fig. 4.1. In more detail, at each step i, we start to compute daily return vector $\mathbf{r}$ of N assets at the test point $t_0 - i + 1$, and for each version j of shrinkage estimation, we will respectively compute the intensity δ_i^j using the j-th Ledoit-Wolf version LW_j on the training period $D_i = [t_0 - i - W, t_0 - i]$. Then, for each intensity δ_i^j, it is easy to derive the daily return by j-th method of portfolio $\mathbf{R}_i^j$ estimated at test point $t_0 - i + 1$ via the weighted sum of daily return of N asset (here, we consider the estimated weights $\hat{w}_*$ described in sub-section 4.3.1). Now, we can compute a list of $M_{\mathbf{F}}$ estimated Sharpe ratios considered on the validation subset of V test points. Finally, we take the version of shrinkage estimator j_* which gives the highest estimated Sharpe ratio, and the algorithm will return the voting shrinkage intensity $\hat{\delta}^*$ as the average of V shrinkage intensities $(\delta_i^{j_*})_{j=1}^V$. A summary of our algorithm is described in Algorithm 0.

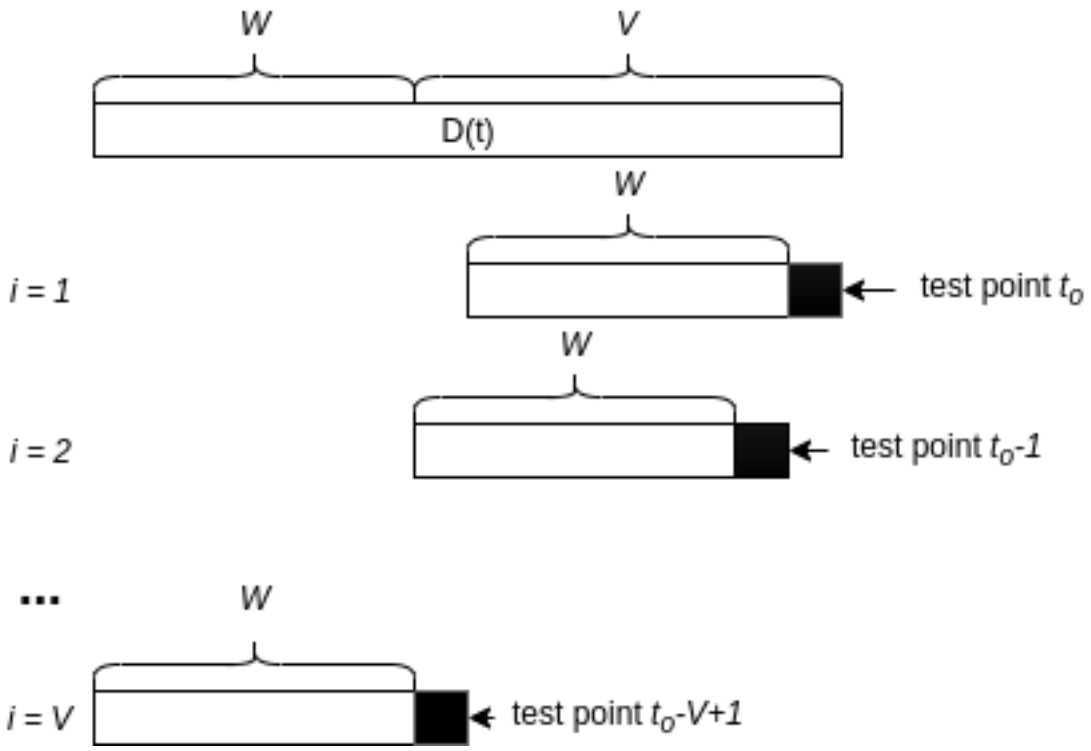


Figure 4.1: An intuitive visualization of the time-series cross-validation process in our approach. From a time-series stock data $D(t)$, we use W data points before a testing point $t_0 - i + 1$ to construct a portfolio then evaluate its performance on the testing point. There are V folds for validation, i.e. V testing points.

Algorithm 4 Voting shrinkage intensity based Ledoit-Wolf framework (Voting-LW)

Input: $D(t)$ for $t \in [t_0 - V - W, t_0]$, t_0, W, V, $M_{\mathbf{F}}$.
Output: Voting-LW shrinkage intensity $\hat{\delta}^*$

$\Delta \leftarrow$ empty list of shrinkage intensities
for $i = 1 : V$ **do**
  Select subset of $D(t)$ from $t_0 - i - W$ to $t_0 - i$: $D_i \leftarrow D(t)\,|_{[t_0-i-W,t_0-i]}$
  $\mathbf{r} = D(t_0 - i + 1)\text{diag}^{-1}\left(D(t_0 - i)\right) - \mathbf{1}$
  for $j = 1 : M_{\mathbf{F}}$ **do**
    $\delta_i^j \leftarrow \text{LW}_j(D_i)$
    $\mathbf{R}_i^j = \mathbf{r} \cdot \hat{\mathbf{w}}_* \left(\delta_i^j, D_i\right)$
  end for
end for
for $j = 1 : M_{\mathbf{F}}$ **do**
  $SR^j = SR\left(\mathbf{R}_1^j, ..., \mathbf{R}_V^j\right)$
end for
Voting: $j_* = \arg\max_{j=1:M_{\mathbf{F}}} \left(SR^j\right)$
Return $\hat{\delta}^* = \frac{1}{V}\sum_{i=1}^{V} \delta_i^{j_*}$.

4.4 Experimental Results

4.4.1 Data

To do the experiments for the illustration of our algorithm performance, we consider using the daily Vietnam stock prices (extracted from Ho Chi Minh City Stock Exchange - HOSE), from January 2011 to October 2019. The whole dataset was taken directly from the HOSE's website. Here, the total observations is $T = 2275$, given the sliding window $W = 250$ and the validation periods is $V = 250$. In the preprocessing step, we faced with some errors due to the data ingestion issues in the server. The two most popular scenarios include missing price and/or volume, and multiple successive days having the same price with the volumes are all zeros. Hence, after crawling and updating the data into our database at the end of a trading day, we need to match the information of stocks' prices and volumes with other sources and use different techniques to impute the data before jumping into all later computational steps. Also, in order to confirm the quality of the analysis, we only consider assets which have at least 250 daily data.

4.4.2 Portfolio Performance measures

To better emphasize on the advantage of the proposed algorithm, we will make a comparison of our work and previous ones achieved by Ledoit-Wolf regarding the performance of portfolio selection by taking into consideration five different indicators, including the common indicators such as Annual Return, Annual Volatility, and Sharpe ratio. Moreover, we use other indicators such as Portfolio Turnover and Alpha.

Annual return & volatility, Sharpe ratio

Two simplest measures in portfolio evaluation are portfolio's annual return and volatility, denoted as R and σ respectively, which are computed directly from portfolio's return values by Equation 4.6 and Equation 4.7. The Sharpe ratio is obtained by the ratio of the excess return and volatility, which is already described in Section 4.3.2.

Portfolio turnover

Following DeMiguel, Garlappi, et al. (2009), portfolio turnover models the level of stability on a portfolio's weights over an investment horizon. It means, at a high level, the turnover indicator measures the changing in portfolio structure over a specific time period. Therefore, a low turnover is often preferable in Minimum Variance based portfolio selection, as it reduces unexpected risks such as liquidity risks and practically implies lower transaction costs. As a matter of fact, its formula on N considered assets in a period of length L is defined as:

$$PT = \frac{1}{L-1} \sum_{t=1}^{L-1} \sum_{i=1}^{N} \left(|w_{t+1,i} - w_{t,i}| \right) \tag{4.8}$$

where $w_{t,i}$ is the weight of an asset i at timepoint t. In another word, the Equation 4.8 measures the average of the total absolute changes of the portfolio weights over the L rebalancing points.

Alpha

Alpha is known as Jensen's alpha (Jensen, 1968), it measures the active performance of the portfolio compared with its benchmark. In our experiment, Vietnam's market index (VNIndex) is taken as our benchmark. For example, an $\alpha = 1\%$ means, in the same period, the portfolio's return is 1% better than the market, therefore a portfolio with a highly positive alpha is more preferred. Theoretically, it is modelled by the following expression:

$$\alpha = 252\mathrm{E}[(\mathbf{R} - \mathbf{R}_f) - \beta(\mathbf{R}_m - \mathbf{R}_f)], \tag{4.9}$$

in which, β is calculated by the below formula:

$$\beta = \frac{\mathrm{Cov}[\mathbf{R} - \mathbf{R}_f, \mathbf{R}_m]}{\mathrm{Var}[\mathbf{R} - \mathbf{R}_f]} \tag{4.10}$$

where $\mathbf{R}_f$ and $\mathbf{R}_m$ are a risk-free rate vector and benchmark's return vector in the same period with the actual portfolio's return vector $\mathbf{R}$.

4.4.3 Analysis of results

In our experimental design, we take three popular Ledoit-Wolf based approaches mentioned above (SSIM, SCCM and STIM) as our benchmark (i.e., $M_{\mathbf{F}} = 3$). Specifically speaking, we will compare our proposed Voting-LW algorithm with them taking into account five performance measures produced by our backtesting system. A completely perfect portfolio among all feasible portfolios will be the one having higher return, lower volatility, higher Sharpe ratio, lower turnover and higher alpha. In real life, we mostly cannot achieve this kind of ideal portfolio. Instead, we mostly rely on a "best" portfolio which can beat the others on at least three indicators (but must include return and Sharpe ratio). We report our experimental results in Table 4.1, Table 4.2 and Table 4.3 for three types of stocks' universes with the number of assets $N = [50, 100, 200]$ respectively. All values in these tables are shown in basis point (bps) which is a common unit of measure in finance (Malkiel, 2013) (a bps is equivalent to 1/100 of 1%).

To be realistic and align with our pre-built backtesting system, we always use the latest price data to construct the portfolio and fit them with future prices at every timepoint. Technically, we compute the optimal weights of N stocks (see 4.3.1) on the universe and then the portfolio will be executed at the next timepoint by following the instruction from the backtesting system. Here, to avoid the possible risk of liquidity, we consider only N assets having the highest liquidity at each timepoint. We also consider the transaction cost in our experiments according to regulations of the Vietnam stocks market.

The results show that the voting-LW has superior results compared to three popular Ledoit-Wolf shrinkage methods over three stocks' universes. Our proposed approach actually can beat

Table 4.1: Experimental results carried out with $N = 50$ assets.

Method	Return	Volatility	Sharpe	Turnover	Alpha
SSIM	684	1680	2198	593	-463
SCCM	1187	1633	4967	**526**	-28
STIM	523	1702	1292	556	-616
Voting-LW	**1404**	**1580**	**6496**	603	**232**

Table 4.2: Experimental results carried out with $N = 100$ assets.

Method	Return	Volatility	Sharpe	Turnover	Alpha
SSIM	1336	1449	6340	626	223
SCCM	1759	1365	9312	**473**	606
STIM	1200	1469	5480	560	86
Voting-LW	**1985**	**1359**	**10976**	580	**720**

at least four indicators compared with the original ones in all three experiments. In general, as we can see, the voting-LW's returns are clearly improved, its volatility is not too volatile and tends to decrease (i.e., competitive or lower than three benchmark methods), which leads to its Sharpe ratios basically increasing. This also explains why the corresponding Alpha is the highest positive and its performance gives better results than VNIndex. The remaining limitation is that the voting LW's turnover is not really stable and tends to increase higher than the other shrinkage methods except the experiment reported in Table 4.3. This implies that the status of the portfolio by the voting LW method has a tendency to change more and cost more transaction fees than the traditional shrinkage methods. However, this issue may come from the default settings in our backtesting process to derive these indicators. By default, our backtesting system uses a fixed time length to change the portfolio's weights, while in our approach, we could flexibly select the rebalancing points in a given validation subset. It means that an investor can proactively limit the number of portfolio rebalances, hence reducing the values of turnover.

Table 4.3: Experimental results carried out with $N = 200$ assets.

Method	Return	Volatility	Sharpe	Turnover	Alpha
SSIM	1789	1096	11515	689	835
SCCM	2073	**1039**	14222	635	1092
STIM	1824	1106	11742	572	836
Voting-LW	**2104**	1042	**14435**	**522**	**1102**

4.5 Conclusions

In this chapter, we present a voting algorithm which can adapt to the fluctuation of the real data to improve the shrinkage intensity selection in covariance matrix estimation. Our approach employed the leave-one-out cross-validation technique, popularly used in machine learning, to

indirectly select the suitable target matrix of shrinkage estimation having the highest Sharpe ratio. Then, we can trace back the list of optimal shrinkage intensities during the evaluation period to derive the final voting intensity. From the experimental results, we can conclude that: firstly it shows our voting method is able to give remarkable results compared with three state-of-the-art shrinkage estimation methods following the comparison of five common financial indicators, and secondly, our voting approach can outperform the market index, which is verified via highly positive Alpha. It should be noted that our proposed method can also be used for the other areas of application related to the covariance matrix estimation-based shrinkage framework. Finally, for future research, we are going to extend our research on target selection based on random validation sets in an attempt to support our backtesting system to select the appropriate periods for executing the stock transactions.

References

Arlot, Sylvain, Alain Celisse, et al. (2010). "A survey of cross-validation procedures for model selection". In: *Statistics surveys* 4, pp. 40–79 (cit. on pp. 89, 137).

Baker, Malcolm, Brendan Bradley, and Jeffrey Wurgler (2011). "Benchmarks as limits to arbitrage: Understanding the low-volatility anomaly". In: *Financial Analysts Journal* 67.1, pp. 40–54 (cit. on pp. 88, 136).

Bergmeir, Christoph, Rob J Hyndman, and Bonsoo Koo (2018). "A note on the validity of cross-validation for evaluating autoregressive time series prediction". In: *Computational Statistics & Data Analysis* 120, pp. 70–83 (cit. on pp. 89, 137).

Candelon, Bertrand, Christophe Hurlin, and Sessi Tokpavi (2012). "Sampling error and double shrinkage estimation of minimum variance portfolios". In: *Journal of Empirical Finance* 19.4, pp. 511–527 (cit. on pp. 90, 137).

Celisse, Alain and Stéphane Robin (2008). "Nonparametric density estimation by exact leave-p-out cross-validation". In: *Computational Statistics & Data Analysis* 52.5, pp. 2350–2368 (cit. on p. 92).

Conway, Delores A and Marc R Reinganum (1988). "Stable factors in security returns: Identification using cross-validation". In: *Journal of Business & Economic Statistics* 6.1, pp. 1–15 (cit. on pp. 89, 137).

Deboeck, Guido J (1994). *Trading on the edge: neural, genetic, and fuzzy systems for chaotic financial markets*. Vol. 39. John Wiley & Sons (cit. on pp. 88, 136).

DeMiguel, Victor, Lorenzo Garlappi, Francisco J Nogales, and Raman Uppal (2009). "A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms". In: *Management science* 55.5, pp. 798–812 (cit. on pp. 32, 94, 107, 141, 147).

DeMiguel, Victor, Alberto Martin-Utrera, and Francisco J Nogales (2013). "Size matters: Optimal calibration of shrinkage estimators for portfolio selection". In: *Journal of Banking & Finance* 37.8, pp. 3018–3034 (cit. on pp. 89, 137).

Golosnoy, Vasyl and Yarema Okhrin (2007). "Multivariate shrinkage for optimal portfolio weights". In: *The European Journal of Finance* 13.5, pp. 441–458 (cit. on pp. 89, 137).

— (2009). "Flexible shrinkage in portfolio selection". In: *Journal of Economic Dynamics and Control* 33.2, pp. 317–328 (cit. on pp. 89, 137).

Jensen, Michael C (1968). "The performance of mutual funds in the period 1945–1964". In: *The Journal of finance* 23.2, pp. 389–416 (cit. on pp. 95, 141).

Ledoit, Olivier and Michael Wolf (2003). "Improved estimation of the covariance matrix of stock returns with an application to portfolio selection". In: *Journal of empirical finance* 10.5, pp. 603–621 (cit. on pp. viii, 6, 28–30, 32, 33, 89–91, 103, 105, 137, 138).

— (2004a). "A well-conditioned estimator for large-dimensional covariance matrices". In: *Journal of multivariate analysis* 88.2, pp. 365–411 (cit. on pp. viii, 6, 28, 32, 89, 91, 103, 105, 137).

— (2004b). "Honey, I shrunk the sample covariance matrix". In: *The Journal of Portfolio Management* 30.4, pp. 110–119 (cit. on pp. 6, 34, 89, 91, 103, 106, 137).

— (2010). *Nonlinear Shrinkage Estimation of the Covariance Matrix of Asset Returns*. Tech. rep. Working paper, University of Zurich (cit. on pp. 89, 137).

Li, Yelin, Junjie Wu, and Hui Bu (2016). "When quantitative trading meets machine learning: A pilot survey". In: *2016 13th International Conference on Service Systems and Service Management (ICSSSM)*. IEEE, pp. 1–6 (cit. on pp. 88, 136).

Malkiel, Burton G (2013). "Asset management fees and the growth of finance". In: *Journal of Economic Perspectives* 27.2, pp. 97–108 (cit. on p. 95).

Picard, Richard R and R Dennis Cook (1984). "Cross-validation of regression models". In: *Journal of the American Statistical Association* 79.387, pp. 575–583 (cit. on pp. 89, 137).

Shao, Jun (1993). "Linear model selection by cross-validation". In: *Journal of the American statistical Association* 88.422, pp. 486–494 (cit. on p. 92).

Sharpe, William F (1964). "Capital asset prices: A theory of market equilibrium under conditions of risk". In: *The journal of finance* 19.3, pp. 425–442 (cit. on p. 91).

Upton, David E (1994). "Cross-validation of the economic significance of factors in security returns". In: *Journal of Business Research* 31.1, pp. 33–38 (cit. on pp. 89, 137).

Zhang, Lan (2012). "Implied and realized volatility: empirical model selection". In: *Annals of Finance* 8.2-3, pp. 259–275 (cit. on pp. 89, 137).

Chapter 5

Ensemble Covariance Estimation for the Global Minimum Variance Portfolio

Objectives

A covariance matrix is an important parameter in many computational applications, such as quantitative trading. Recently, a global minimum variance portfolio receive great attention due to its performance after the 2007-2008 Financial Crisis and this portfolio uses only a covariance matrix to calculate weights for assets. However, the calculation process of that portfolio is sensitive to outliers in the covariance matrix, for example, a sample covariance matrix estimation or linear shrinkage covariance matrix estimations. In this chapter, we propose to use an undersampling technique and ensemble learning to stabilize the covariance matrix by reducing the impacts of outliers on an output of covariance estimation. Experimenting on an emerging stock market by three performance metrics shows that our approach improves significantly the sample covariance matrix and also a linear shrinkage to single-index model to a level of two shrinkage estimations, a shrinkage to identity matrix and shrinkage to constant correlation model.

Contents

5.1 Introduction

After a seminal theory of portfolio selection of H. M. Markowitz (1968), the computation of efficient portfolios is an area of interest in the finance profession which aims to estimate optimal weights for assets. That uses the mean and the covariance of stock returns as input parameters, however, an estimation error of the expected returns is larger than the covariance estimation error (Merton, 1980). Recent research studies suggest assuming equal expected returns for all stocks and using only the covariance matrix to estimate the portfolio offers the lowest risk, in other words, a Global Minimum Variance Portfolio (GMVP) (DeMiguel and Nogales, 2009). The standard statistical estimation of the covariance matrix in the GMVP is the sample covariance estimation. While the mathematics of the GMVP and the sample covariance matrix is straightforward, several studies in the theory and industrial practice pointed out the disadvantages of the sample covariance matrix in portfolio trading.

When a number of assets in our universe are greater than a number of historical returns, formally when a number of dimensions are greater than a number of observations, the estimated sample covariance matrix from those data is not invertible and also ill-conditioned. Portfolio optimization in this context contains high estimation error and it turns out that the portfolio needs to adjust its positions more frequently which will increase the risk level of the portfolio and also decrease its profit. One approach to making the covariance matrix invertible in statistics is by imposing the matrix structure, such as a shrinkage technique that takes a weighted combination of two matrices, the sample covariance matrix and another invertible matrix (Stein, 2020). The weight, also known as shrinkage intensity, is optimized from a quadratic loss function on a given dataset without looking at the out-of-sample data. In this case, there is a unique "optimal" shrinkage intensity for one dataset. On the other hand, some recent studies (Tong et al., 2018; T. Tran, N. Nguyen, T. Nguyen, and Mai, 2020) split the dataset and look at validation subsets to evaluate the "optimal" shrinkage intensity by cross-validation method instead of optimization, which resulted in different "optimal" shrinkage intensities based on different pre-defined objectives.

Beyond a linear shrinkage estimation of a sample covariance matrix with another well-structured matrix, Olivier Ledoit et al. (2012) and Olivier Ledoit et al. (2015) proposed a nonlinear shrinkage method by taking into account eigenvalues of the population that are estimated from the eigenvalue distribution of a sample dataset. The original idea is from a Random Matrix Theory which effectively deals with noise in large-dimensional covariance matrices, however, it requires complex mathematics and is difficult to understand for practitioners. Another simpler nonlinear method splits the sample dataset and directly uses eigenvalues on the validation subset as the population eigenvalues (Lam, 2016). Although this method is less accurate, it is simpler and easier to understand than the previous nonlinear approach. Another extension of the linear shrinkage is using multiple target matrices. However, the performances of the original linear shrinkage estimators depend on a chosen target matrix. Therefore multiple different targets could avoid this dependency and make the estimator more flexible (Gray et al., 2018).

Instead of going to be more complex, the benefits of the shrinkage estimators are questioned

by a simple approach of a portfolio of covariance estimators (Jagannathan et al., 2000; Bengtsson et al., 2002) The portfolio of covariance estimators contains the sample covariance matrix and estimators with different assumptions and direction errors. Then it simply takes an average of them to neutralize the estimation error of the sample covariance matrix by specification errors of the others. A general comparison of Disatnik et al. (2007) compares those simple portfolios with the linear shrinkage portfolios, combinations in this study could be equally weighted or even random weighted. They show that out-of-sample performances of the simple portfolios and the complex shrinkage portfolios perform within the same range, at least on a standard deviation of the portfolio returns. Therefore they conclude that there is no statistical improvement and also no benefit from using those complicated shrinkage estimators.

In practice, fund managers are not only looking for the best covariance estimator but also seeking alternatives in order to expand their fund's capacity. Those alternatives are different to the best ones but have comparable portfolio performances. Therefore, the portfolio of estimators is a simple method but important for investment management and they also raise a question about the effectiveness of the shrinkage estimation. However, there is a lack of investigation on sustainability of those covariance matrix estimations. An advanced estimation which has higher performances but unsustainable outcomes is not preferable. For example, if a portfolio changes its positions unreasonably just because of some outliers in the data then it leads to other problems for the fund managers such as transaction costs or liquidity risk. From the risk perspective, that is not the best portfolio and those alternatives are more attractive to fund managers if they are more sustainable.

In this study, we questionize the stability of multiple covariance matrices to understand how they work in portfolio optimization under the data perspective and it helps fund managers to develop their sustainable investments. Firstly, does the strength of the shrinkage estimation come from the combination with other matrices or from an optimal shrinkage intensity calculation? Secondly, could we combine with other sample covariance matrices instead of using different matrices? The first question is important in theoretical mathematics because understanding the characteristics of not only the sample covariance matrix but also the shrinkage estimation is a principal research topic in various domains. The second question is important in practical finance because they prefer a simple, effective, understandable and explainable method to a new, complicated and unproven method.

One problem with an estimated covariance matrix is highly fluctuating and sensitive with outliers (Yuan et al., 2001; Leys et al., 2018; Raymaekers et al., 2021; Ke et al., 2019). In many applications in statistics and machine learning, ensemble learning and undersampling are efficient methods to reduce not only the variance of predictions but also the impact of outliers. Therefore, we propose to use ensemble learning with undersampling technique on a given covariance estimator to reduce the impact of outliers and stabilize our estimated covariance matrix. We test our approach with the sample covariance matrix and three linear shrinkage estimations on the Vietnam stock market from 2013 to the end of 2019. Backtesting on the historical data, out-of-sample portfolio performances show that applying our approach to the sample covariance matrix achieves comparable results to the portfolios of shrinkage techniques, or even better in

some cases. Comparing to the sample covariance matrix, our portfolio significantly outperforms on all performance metrics.

The rest of this chapter is organized as follows. Section 5.2 describes backgrounds and their formulations. Section 5.3 presents our methodology. Section 5.4 summarizes our dataset and evaluation metrics. Section 5.5 shows our experimental results. Section 5.6 makes our conclusions.

5.2 Background

5.2.1 Linear shrinkage estimation

Given a dataset of N random variables and T observations representing T returns of N assets on a universe, the GMVP is formulated as follows:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^\intercal \mathbf{\Sigma} \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^\intercal \mathbf{1} = 1 \end{aligned} \tag{5.1}$$

where the $\mathbf{\Sigma}^{N \times N}$ is the covariance matrix and the $\mathbf{w} = \{w_1, \cdots, w_N\}$ is a weight vector of N assets respectively. The well-known solution of Equation 5.1 is:

$$\mathbf{w} = \frac{\mathbf{\Sigma}^{-1}\mathbf{1}}{\mathbf{1}^\intercal \mathbf{\Sigma}^{-1}\mathbf{1}}. \tag{5.2}$$

The most common estimator, the sample covariance matrix estimator, not always be invertible and also ill-conditioned. Particularly in the financial industry, when the number of assets N is greater than the number of observations T ($N >> T$), the inverse of the sample covariance matrix amplifies estimation error.

For the high dimensional covariance matrices, Olivier Ledoit et al. (2004a), Olivier Ledoit et al. (2003), and Olivier Ledoit et al. (2004b) proposed to impose the structure of the covariance matrix by using the shrinkage technique. They combined linearly the sample covariance matrix with another well-conditioned structured matrix. Given a target matrix $\mathbf{F}$, the linear shrinkage estimator is as follows:

$$\widehat{\Sigma} = \delta \mathbf{F} + (1 - \delta)\mathbf{S} \tag{5.3}$$

in which, $\mathbf{S}$ is the sample covariance matrix and $\delta \in (0, 1)$ is a shrinkage intensity. The optimal δ is obtained by minimize the quadratic loss of $|\mathbf{F} - \mathbf{S}|$. The target matrix $\mathbf{F}$ is a domain-based matrix and particularly in finance, Ledoit and Wolf proposed three different targets and showed that they are empirically significant better than the sample covariance matrix, including Shrinkage towards identity matrix (Olivier Ledoit et al., 2004a), Shrinkage towards single-index model (also known as Shrinkage to market) (Olivier Ledoit et al., 2003) and Shrinkage towards constant correlation model (Olivier Ledoit et al., 2004b).

5.2.2 Ensemble learning and undersampling

Besides of dimension curse, the covariance matrix is also sensitive with outliers (Yuan et al., 2001; Leys et al., 2018; Raymaekers et al., 2021; Ke et al., 2019), especially in the high dimension matrix estimated from the limited input data. This leads to an unstable covariance matrix. The outliers also affect several problems in Machine Learning and various approaches have been proposed to handle this issue. For example, Breiman (2001) propose to use undersampling and ensemble learning to build one robust estimator from several weak estimators which are low-bias but high variance functions. They use undersampling to build a large number of decorrelated estimators and then combine them to reduce the variance of the final prediction. The combination rule could be a simple average for regression problems or major voting for classification problems. This approach is not only simple to use but also efficient in many problems. One of the advantages is complementary: i) combining the strength of each estimator and ii) any outlier in the input data or even in the prediction of a single estimator does not affect the final estimator.

Dealing with a small partial of the outliers in the large covariance matrix, outlier detection in this context could face with another problem, a problem of imbalance between outliers and non-outliers. T. Tran, L. Tran, et al. (2019) show that the undersampling technique combines with ensemble learning is efficient not only on a massive dataset but also on extremely imbalance cases. They propose to split the original data into k segments with the same imbalance ratio then build k estimators and finally combine them into a final predictor. Those segments are different but have similar data semantics, this characteristic is similar to the stock data for covariance estimation.

5.3 Methodology

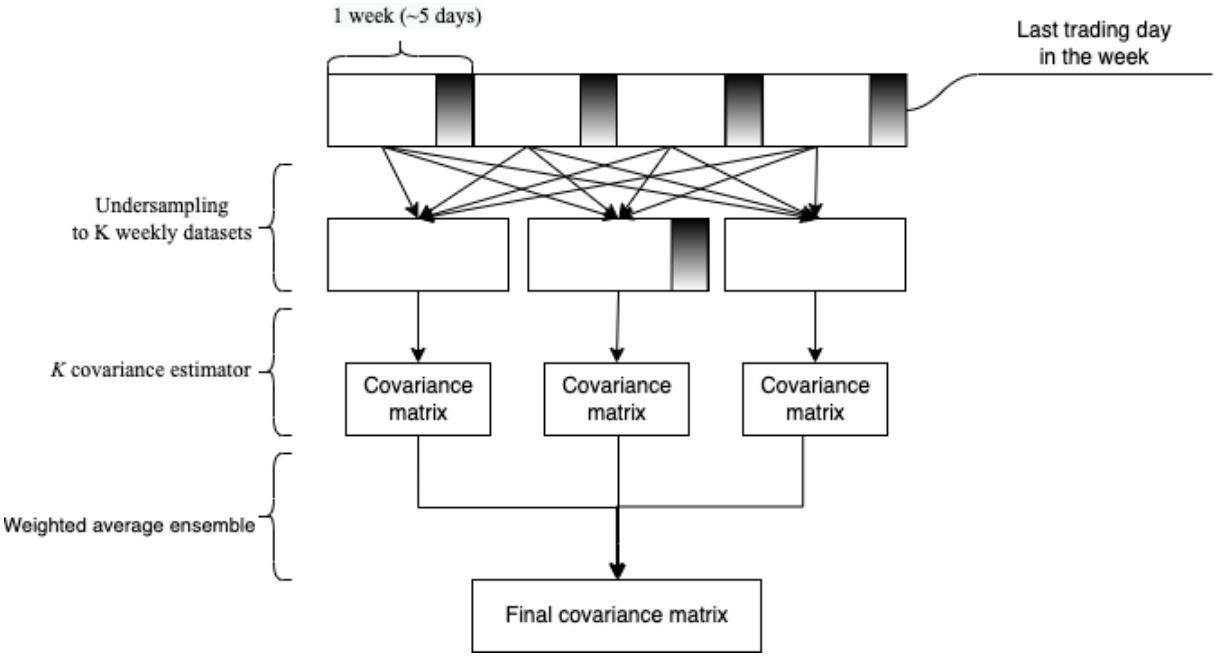
The goal of the GMVP is to minimize the variance of the portfolio and it requires a stable covariance matrix over time. Typically, a dataset for covariance estimation is a weekly basis which is extracted from the daily dataset simply by getting the last trading day in each week[1]. However, abnormal data exist in those last days and that leads to outliers in the covariance matrix. To reduce the impact of the outliers in the covariance matrix, we propose to use the undersampling method and then apply ensemble learning for covariance estimations at the data level. Our sampling procedure is as follows: a daily historical return dataset is randomly sampled with replacement into k weekly subsets with the same size, in other words, we pick a random day in each week instead of the last day. With a given covariance estimator, each of them is used to estimate a covariance matrix as a covariance predictor.

The above sampling process aims to take a nearby data point in each week that has similar information to the last day. The covariance predictors build on those subsets have similar semantics and also contain different outliers. To combine those covariance predictors, we apply the simple average to neutralize the outliers in one predictor by the non-outliers in other predictors.

[1]Many research studies use monthly data but there are not enough monthly data for estimations in younger markets.

This ensemble also reduces the changes in the covariance matrix, in other words, the covariance matrix is more stable. This procedure is illustrated in Figure E.2.

Figure 5.1: Visualization in details of our K-covariance approach. In finance, the traditional weekly dataset to estimate the covariance matrix take only those last trading day in each week. In K-covariance, we undersampling randomly a trading day in each week. From those weekly datasets, with any given covariance estimator, we estimate K covariance matrices. Then using weighted average ensemble to combine those matrices into one final covariance matrix. This matrix is used in the portfolio optimization of the GMVP as normal.



We conduct an empirical experiment to compare with the linear shrinkage covariance estimators and also the portfolios of covariance estimators. While the combinations in the portfolios of covariance estimators could be seen as a method-level ensemble, our approach basically is data-level. Totally, there are eleven portfolios in our experiment with different covariance estimators, including:

- The sample covariance matrix (denoted as sample),
- The Shrinkage to the identity matrix (denoted as STIM): this is the linear shrinkage estimator which combines the sample covariance matrix and the identity matrix by an optimal shrinkage intensity (Olivier Ledoit et al., 2004a),
- The Shrinkage to single-index model (denoted as SSIM): this is the linear shrinkage estimator which uses the single-index model of Sharpe (1963) as the target matrix and then combines with the sample covariance by an optimal shrinkage intensity (Olivier Ledoit et al., 2003),
- The Shrinkage to constant correlation matrix (denoted as SCCM): this is the linear shrinkage estimator, a target matrix assumes the correlations of each pair of stocks are the same

to estimate a constant-correlation covariance matrix and then combines with the sample covariance matrix by an optimal shrinkage intensity (Olivier Ledoit et al., 2004b),

- The combination of the sample matrix, the diagonal matrix and the single-index model (denoted as sTS): this is a portfolio with covariance matrix as a combination of basic elements, including the sample matrix, the diagonal matrix and the single-index model, then simply take an equally average instead of optimal weighting (Jagannathan et al., 2000),

- The combination of the sample matrix, the single-index model and the constant correlation matrix (denoted as sSC): this is a portfolio with covariance matrix as a combination of basic elements, including the sample matrix, the single-index model and the constant correlation matrix, then simply take an equally average instead of optimal weighting (Disatnik et al., 2007),

- The combination of the sample matrix, the diagonal matrix and the constant correlation matrix (denoted as sTC): this is a portfolio with covariance matrix as a combination of basic elements, including the sample matrix, the diagonal matrix and the constant correlation matrix, then simply take an equally average instead of optimal weighting,

- Given one of the following estimators: the sample covariance estimator, the linear shrinkage to the identity matrix estimator, the linear shrinkage to the single-index model estimator and the linear shrinkage to constant correlation estimator. We apply our method to a given estimator to build our GMVPs, they are denoted as k-sample, k-STIM, k-SSIM and k-SCCM respectively.

A backtesting process for the portfolios above is as follows: given one of the above covariance estimators, at the end of each week a GMVP is built on the latest data then at the first trading day of next week, it will execute and match by the closing prices of that day. We hold it until the weekend then re-compute and rebalance the portfolio, in other words, the out-of-sample period is one week. Particularly, the training data for the covariance estimator is two-year weekly data, in other words, the in-sample period is two years. Considering the data quality, at each date, we use only the assets which have at least a half of non-null data, in other words, one year after their IPOs.

In our backtesting process, to evaluate a portfolio precisely, we follow real regulations and restrictions of the market. Firstly, instead of assuming no impact of commission, we apply a real commission fee on every transaction. Secondly, we use the long-term interest rate of government bonds in the portfolio performance calculations. Thirdly, we limit the maximum number of shares that could be traded at each date but to simplify the process, we assume that we could long/short up to the real traded volume in the dataset and also there is no slippage in our transactions. Last but not least, to avoid a lookahead bias, at the end of each date the latest data we can use is the historical data up to that date and then the orders will fit with the next date's prices.

5.4 Dataset and Evaluation Metrics

5.4.1 Dataset

In this study, we test our approach and other portfolios on the Vietnam stock market, one of the emerging markets in Asia. Particularly, we use historical trading data of all stocks traded on the Ho Chi Minh City Stock Exchange (HOSE) which are available on their website. The period of our experiment is over seven years, from 2013 to 2019, with 1744 trading days in total. According to their trading regulation, short-sell is not allowed and we only focus on long-only portfolios. Statistical summary of data is described in Table 5.1, unit for average volume is 10^5 shares and for others is a basis point.

Table 5.1: Statistical summary of the historical data on the HOSE exchange from 2013 to 2019.

	2013	2014	2015	2016	2017	2018	2019
Minimum number of stocks	304	302	303	308	324	350	380
Maximum number of stocks	316	310	313	324	350	380	387
Average daily return	16	13	6.23	5.26	10.44	-1.79	2.42
Standard deviation	815.88	276.2	354.13	283.13	255.54	275.92	249.2
Average volume	4.54	9.03	8.12	8.88	11.84	11.19	9.66

5.4.2 Evaluation Metrics

To evaluate the portfolios, we use three portfolio performance metrics that are commonly used in finance, including Annual Volatility, Sharpe ratio and Portfolio Turnover. The Annual Volatility (σ) is standard deviation of portfolio returns which indicates a risk of that portfolio. The volatility in the context of the GMVP, lower is better. The Sharpe ratio is developed by Sharpe (1994) to describe the profit of an investment compared to its risk, it is defined as follows:

$$\text{Sharpe ratio} = \frac{R - R_f}{\sigma}, \tag{5.4}$$

where R is the return of a portfolio and R_f is a risk-free rate. For R_f, we use Vietnam's 10-year government bond in the same period. A portfolio with a higher Sharpe ratio is a better portfolio.

The Portfolio Turnover measures a level of stability of the portfolio weights over the investment horizon (DeMiguel, Garlappi, et al., 2009). In other words, it measures a fluctuation of portfolio weights. In the scope of the GMVP, we prefer a portfolio with a lower portfolio turnover. Formula of the portfolio turnover is as follows:

$$\text{Portfolio Turnover} = \frac{1}{T-1} \sum_{t=1}^{T-1} \sum_{i=1}^{N} \left(|\mathbf{w}_{t+1,i} - \mathbf{w}_{t,i}| \right) \tag{5.5}$$

where $\mathbf{w}_{t,i}$ is a weight of asset i at date t. A maximum value for the Portfolio Turnover from Equation 5.5 is two, in other words, 200% changes from a portfolio weights to another completely different weights, and a minimum value is 0%.

Only out-of-sample results are reported for those evaluation metrics. We also report p-value for each pair of covariance matrix estimators for all performance metrics. Particularly for the Sharpe ratio, we use a bootstrapping method which is suggested by Oliver Ledoit et al. (2008) to do a robust performance hypothesis testing.

5.5 Results and discussions

In Table 5.2, we report out-of-sample portfolio performances of eleven portfolios on three evaluation metrics. Hypothesis testing results for each pair of those portfolios, including: Table 5.4 is p-values of the annual volatility, Table 5.5 is p-values of the Sharpe ratio and Table 5.6 is p-values of the portfolio turnover. Similarly in Table 5.3, we report out-of-sample portfolio performances of eleven portfolios but only on one hundred largest assets by market capitalization ($N = 100$). Table 5.7, Table 5.8 and Table 5.9 in are p-values of the annual volatility, the Sharpe ratio and the portfolio turnover respectively for these portfolios.

A first noteworthy observation is that the most basic method in our experiment, the sample portfolio, is significantly worse than all other portfolios on all of three metrics, this shows the instability of the sample covariance matrix. Three more complicated portfolios, using the linear shrinkage estimators, have no difference both on the volatility and the Sharpe ratio . But on the portfolio turnover, the SSIM portfolio is significantly higher than the STIM and SCCM portfolios and these two portfolios have the same turnover .

Table 5.2: Out-of-sample portfolio performances of eleven Global Minimum Variance Portfolios with different covariance matrix estimations. All available stocks on the HOSE exchange are considered.

Estimator	Annual Volatility (%)	Sharpe ratio	Portfolio Turnover (%)
sample	8.30	1.39	7.28
STIM	7.24	1.94	3.49
SSIM	7.37	2.00	4.41
SCCM	7.71	1.94	3.43
k-sample	7.80	1.77	5.62
k-STIM	7.13	2.04	3.34
k-SSIM	7.17	2.07	3.86
k-SCCM	7.52	1.91	3.23
sTS	6.79	2.24	2.88
sTC	7.46	2.01	2.89
sSC	7.47	2.01	2.90

Table 5.3: Out-of-sample portfolio performance of eleven Global Minimum Variance Portfolios with different covariance matrix estimations. Only the top one hundred assets by market capitalization are considered ($N = 100$).

Estimator	Annual Volatility (%)	Sharpe ratio	Portfolio Turnover (%)
sample	9.81	0.82	4.30
STIM	9.34	0.99	3.17
SSIM	9.54	0.92	3.78
SCCM	9.42	1.09	3.18
k-sample	9.47	0.82	3.67
k-STIM	9.16	0.93	2.91
k-SSIM	9.31	0.86	3.32
k-SCCM	9.29	1.01	2.90
sTS	8.90	1.13	2.47
sTC	9.28	1.10	2.59
sSC	9.28	1.10	2.59

Table 5.4: p-value of the Annual Volatility for each pair of covariance matrix estimators in the Global Minimum Variance Portfolio with all assets in the Vietnam stock market in the k-cov approach.

Estimator	-	1	2	3	4	5	6	7	8	9	10	11
sample	1											
STIM	2	<.001										
SSIM	3	<.001	.76									
SCCM	4	<.001	.99	.97								
k-sample	5	<.001	.99	.99	.69							
k-STIM	6	<.001	.26	.09	<.001	<.001						
k-SSIM	7	<.001	.35	.13	.001	<.001	.60					
k-SCCM	8	<.001	.94	.81	.15	.06	.99	.98				
sTS	9	<.001	.003	<.001	<.001	<.001	.02	.01	<.001			
sTC	10	<.001	.90	.71	.09	.03	.97	.95	.38	.99		
sSC	11	<.001	.90	.72	.09	.03	.97	.95	.38	.99	.51	

Table 5.5: p-value of the Sharpe ratio for each pair of covariance matrix estimators in the Global Minimum Variance Portfolio with all assets in the Vietnam stock market in the k-cov approach.

Estimator	-	1	2	3	4	5	6	7	8	9	10	11
sample	1											
STIM	2	.002										
SSIM	3	<.001	.56									
SCCM	4	.02	.97	.74								
k-sample	5	.03	.36	.16	.47							
k-STIM	6	<.001	.27	.74	.65	.06						
k-SSIM	7	<.001	.33	.50	.48	.004	.75					
k-SCCM	8	.03	.92	.63	.77	.49	.52	.30				
sTS	9	<.001	.007	.03	.04	.02	.16	.26	.04			
sTC	10	.01	.71	.93	.27	.32	.89	.75	.38	.06		
sSC	11	.01	.71	.93	.28	.32	.89	.75	.38	.06	.44	

Table 5.6: p-value of the Portfolio Turnover for each pair of covariance matrix estimators in the Global Minimum Variance Portfolio with all assets in the Vietnam stock market in the k-cov approach.

Estimator	-	1	2	3	4	5	6	7	8	9	10	11
sample	1											
STIM	2	<.001										
SSIM	3	<.001	<.001									
SCCM	4	<.001	.79	<.001								
k-sample	5	<.001	<.001	<.001	<.001							
k-STIM	6	<.001	.51	<.001	.71	<.001						
k-SSIM	7	<.001	.13	.04	.08	<.001	.03					
k-SCCM	8	<.001	.27	<.001	.41	<.001	.64	.009				
sTS	9	<.001	.008	<.001	.02	<.001	.04	<.001	.12			
sTC	10	<.001	.01	<.001	.02	<.001	.05	<.001	.15	.95		
sSC	11	<.001	.01	<.001	.03	<.001	.06	<.001	.15	.93	.98	

Table 5.7: p-value of the Annual Volatility for each pair of covariance matrix estimators in the Global Minimum Variance Portfolio with only top $N = 100$ market cap assets in the Vietnam stock market in the k-cov approach.

Estimator	-	1	2	3	4	5	6	7	8	9	10	11
sample	1											
STIM	2	.02										
SSIM	3	.12	.81									
SCCM	4	.04	.64	.29								
k-sample	5	.07	.72	.37	.58							
k-STIM	6	.002	.21	.04	.11	.08						
k-SSIM	7	.01	.44	.14	.30	.23	.74					
k-SCCM	8	.01	.41	.12	.27	.20	.71	.46				
sTS	9	<.001	.02	.001	.008	.004	.11	.03	.03			
sTC	11	.009	.38	.11	.25	.19	.69	.44	.47	.95		
sSC	10	.009	.39	.11	.25	.19	.69	.44	.47	.95	.50	

Table 5.8: p-value of the Sharpe ratio for each pair of covariance matrix estimators in the Global Minimum Variance Portfolio with only top $N = 100$ market cap assets in the Vietnam stock market in the k-cov approach.

Estimator	-	1	2	3	4	5	6	7	8	9	10	11
sample	1											
STIM	2	.08										
SSIM	3	.02	.43									
SCCM	4	.04	.46	.12								
k-sample	5	.98	.18	.30	.05							
k-STIM	6	.36	.48	.93	.27	.20						
k-SSIM	7	.69	.29	.53	.07	.19	.37					
k-SCCM	8	.20	.87	.49	.34	.07	.51	.10				
sTS	9	.02	.13	.05	.68	.02	.07	.03	.32			
sTC	11	.09	.46	.21	.88	.09	.28	.12	.38	.71		
sSC	10	.09	.46	.21	.88	.09	.28	.12	.39	.71	.50	

Table 5.9: p-value of the Portfolio Turnover for each pair of covariance matrix estimators in the Global Minimum Variance Portfolio with only top $N = 100$ market cap assets in the Vietnam stock market in the k-cov approach.

Estimator	-	1	2	3	4	5	6	7	8	9	10	11
sample	1											
STIM	2	<.001										
SSIM	3	.08	.02									
SCCM	4	<.001	.96	.02								
k-sample	5	.03	.05	.69	.06							
k-STIM	6	<.001	.29	<.001	.27	.002						
k-SSIM	7	<.001	.54	.08	.57	.18	.09					
k-SCCM	8	<.001	.28	<.001	.27	.002	.98	.09				
sTS	9	<.001	.003	<.001	.003	<.001	.05	<.001	.05			
sTC	11	<.001	.01	<.001	.01	<.001	.17	.002	.18	.58		
sSC	10	<.001	.01	<.001	.01	<.001	.17	.002	.18	.57	.99	

The results from our experiment show that the undersampling and ensemble learning applied to the sample covariance matrix estimation could improve significantly the portfolio performances on the annual volatility and the Sharpe ratio. These improvements are approximately the same as the three portfolios of optimal shrinkage covariance estimations. Although on the portfolio turnover, it is better than the sample portfolio but still higher than the linear shrinkage portfolios.

These shrinkage portfolios have similar performances, except that the portfolio turnover of the SSIM portfolio is worse than the portfolio turnover of the STIM and the SCCM portfolios. Applying our approach on the shrinkage to the single-index model, the k-SSIM portfolio, improves significantly the portfolio turnover of the SSIM portfolio to the same level of the STIM and the SCCM portfolios. This suggests that the estimated covariance matrix from the SSIM estimation is less stable than the STIM and SCCM estimations.

Furthermore, combinations of more than two different matrices have the lowest portfolio turnover in our experiment and in many cases, they are better than the others. This suggests that adding more matrices that are well-structured and domain-based could stabilize the portfolio. The exceptional estimator in this study, the sTS portfolio, is the combination of the sample matrix, the diagonal matrix and the single-index model and it has the best results in most cases. The diagonal matrix in this context could be seen as a combination of the sample portfolio with the equally-weighted portfolio and that helps to increase the portfolio diversification.

Comparing the three portfolios of optimal shrinkage estimations with the three portfolios of estimators shows that they have comparable results. This suggests that the strength of the shrinkage technique mostly comes from the combination with other matrices and the "optimal" shrinkage intensity calculation is an additional step to make sure its results are the best. Moreover, we conclude that the combinations do not have to use different matrices, the combination

of multiple sample covariance matrices by using ensemble learning and undersampling technique produces similar performances to the advanced and optimal shrinkage estimations.

5.6 Conclusions

In this chapter, we have presented the k-covariance estimation which is using the undersampling approach and ensemble learning to improve the stability of the covariance matrix in quantitative trading, particularly for the global minimum variance portfolio. Our approach reduces the impact of outliers in the covariance estimation by manipulating the process at the data level. All elements in our combinations are homogeneous covariance matrices, such as combining multiple sample covariance matrices and they are estimated from different samples. The experimental results show that the k-covariance approach improves significantly the sample covariance matrix estimation to the performance level of the linear shrinkage estimations. Although the linear shrinkage estimations are robust, applying our approach shows improvements in some cases, such as on the shrinkage to single-index model. In practice, our results suggest that fund managers should focus on two shrinkage estimations, the Shrinkage to the identity matrix and the Shrinkage to constant correlation matrix, which are more stable and the performances of the portfolios of estimators are worth analyzing by the research community.

The scientific novelty of the research is using Machine Learning techniques to build and combine multiple weak covariance matrix estimations to achieve a sustainable covariance matrix estimation. We conclude that combining multiple sample covariance matrices with our approach and similarly applying them to the weak shrinkage estimation achieve the same level of highest portfolio performances of the shrinkage estimations. In which, the improvement of the shrinkage estimations mostly comes from the combination with other matrices while in our approach, we combine only one kind of covariance matrix estimation multiple times which is weak and noisy.

Furthermore, the portfolios of covariance estimations that combine the sample covariance matrix and more than two other matrices have exceptional results, we propose to investigate these portfolios in future work. There are two limitations in our study, the first is the outdated data which is from 2013 to the end of 2019 and the second is our analysis performed on the single market. Because of the COVID-19 crisis in early 2020, the financial markets around the world performed differently during that period. Therefore we focus on the normal scenario of the markets and the crisis scenario is worth studying in a separate study. Another research direction is to test this approach in other markets to verify our conclusions generally.

References

Bengtsson, Christoffer and Jan Holst (2002). *On portfolio selection: Improved covariance matrix estimation for Swedish asset returns*. Citeseer (cit. on pp. 102, 142).

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32 (cit. on pp. 16, 104, 143).

DeMiguel, Victor, Lorenzo Garlappi, Francisco J Nogales, and Raman Uppal (2009). "A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms". In: *Management science* 55.5, pp. 798–812 (cit. on pp. 32, 94, 107, 141, 147).

DeMiguel, Victor and Francisco J Nogales (2009). "Portfolio selection with robust estimation". In: *Operations Research* 57.3, pp. 560–577 (cit. on pp. 101, 142).

Disatnik, David J and Simon Benninga (2007). "Shrinking the covariance matrix". In: *The Journal of Portfolio Management* 33.4, pp. 55–63 (cit. on pp. 102, 106, 142).

Gray, Harry, Gwenaël GR Leday, Catalina A Vallejos, and Sylvia Richardson (2018). "Shrinkage estimation of large covariance matrices using multiple shrinkage targets". In: *arXiv preprint arXiv:1809.08024* (cit. on pp. 101, 142).

Jagannathan, Ravi and Tongshu Ma (2000). "Three methods for improving the precision in covariance matrix estimation". In: *Unpublished working paper* (cit. on pp. 102, 106, 142).

Ke, Yuan, Stanislav Minsker, Zhao Ren, Qiang Sun, and Wen-Xin Zhou (2019). "User-friendly covariance estimation for heavy-tailed distributions". In: *Statistical Science* 34.3, pp. 454–471 (cit. on pp. 102, 104, 143).

Lam, Clifford (2016). "Nonparametric eigenvalue-regularized precision or covariance matrix estimator". In: *The Annals of Statistics* 44.3, pp. 928–953 (cit. on pp. 101, 142).

Ledoit, Oliver and Michael Wolf (2008). "Robust performance hypothesis testing with the Sharpe ratio". In: *Journal of Empirical Finance* 15.5, pp. 850–859 (cit. on pp. 108, 147).

Ledoit, Olivier and Michael Wolf (2003). "Improved estimation of the covariance matrix of stock returns with an application to portfolio selection". In: *Journal of empirical finance* 10.5, pp. 603–621 (cit. on pp. viii, 6, 28–30, 32, 33, 89–91, 103, 105, 137, 138).

— (2004a). "A well-conditioned estimator for large-dimensional covariance matrices". In: *Journal of multivariate analysis* 88.2, pp. 365–411 (cit. on pp. viii, 6, 28, 32, 89, 91, 103, 105, 137).

— (2004b). "Honey, I shrunk the sample covariance matrix". In: *The Journal of Portfolio Management* 30.4, pp. 110–119 (cit. on pp. 6, 34, 89, 91, 103, 106, 137).

— (2012). "Nonlinear shrinkage estimation of large-dimensional covariance matrices". In: *The Annals of Statistics* 40.2, pp. 1024–1060 (cit. on pp. 3, 101, 142).

— (2015). "Spectrum estimation: A unified framework for covariance matrix estimation and PCA in large dimensions". In: *Journal of Multivariate Analysis* 139, pp. 360–384 (cit. on pp. 3, 101, 142).

Leys, Christophe, Olivier Klein, Yves Dominicy, and Christophe Ley (2018). "Detecting multivariate outliers: Use a robust variant of the Mahalanobis distance". In: *Journal of experimental social psychology* 74, pp. 150–156 (cit. on pp. 102, 104, 143).

Markowitz, Harry M (1968). *Portfolio selection*. Yale university press (cit. on pp. 5, 101, 142).

Merton, Robert C (1980). "On estimating the expected return on the market: An exploratory investigation". In: *Journal of financial economics* 8.4, pp. 323–361 (cit. on pp. 101, 142).

Raymaekers, Jakob and Peter J Rousseeuw (2021). "Fast robust correlation for high-dimensional data". In: *Technometrics* 63.2, pp. 184–198 (cit. on pp. 102, 104, 143).

Sharpe, William F (1963). "A simplified model for portfolio analysis". In: *Management science* 9.2, pp. 277–293 (cit. on pp. 32, 105).

— (1994). "The sharpe ratio". In: *Journal of portfolio management* 21.1, pp. 49–58 (cit. on pp. 36, 107, 146).

Stein, Charles (2020). "Inadmissibility of the usual estimator for the mean of a multivariate normal distribution". In: *Contribution to the Theory of Statistics*. University of California Press, pp. 197–206 (cit. on pp. 101, 142).

Tong, Jun, Rui Hu, Jiangtao Xi, Zhitao Xiao, Qinghua Guo, and Yanguang Yu (2018). "Linear shrinkage estimation of covariance matrices using low-complexity cross-validation". In: *Signal Processing* 148, pp. 223–233 (cit. on pp. 101, 142).

Tran, Tuan, Nhat Nguyen, Trung Nguyen, and An Mai (2020). "Voting shrinkage algorithm for Covariance Matrix Estimation and its application to portfolio selection". In: *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*. IEEE, pp. 1–6 (cit. on pp. 6, 101, 142).

Tran, Tuan, Loc Tran, and An Mai (2019). "K-Segments Under Bagging approach: An experimental Study on Extremely Imbalanced Data Classification". In: *2019 19th International Symposium on Communications and Information Technologies (ISCIT)*. IEEE, pp. 492–495 (cit. on pp. 5, 104, 144).

Yuan, Ke-Hai and Peter M Bentler (2001). "Effect of outliers on estimators and tests in covariance structure analysis". In: *British Journal of Mathematical and Statistical Psychology* 54.1, pp. 161–175 (cit. on pp. 102, 104, 143).

Chapter 6

Conclusions and Future Works

6.1 Conclusion

In this thesis, we have described our works with the ensemble learning and other Machine Learning techniques, such as the undersampling or the cross-validation, on various applications from the Machine Learning domains to the financial industry. Our work could be divided into three research questions in two parts. The first part has applications on different topics in Machine Learning, while the second focuses only on financial applications which expands into two research questions. In this chapter, those three scientific contributions of our works are summarized below.

Firstly, we pointed out the new gap between the extremely imbalanced data and the big data. This problem could be seen in many applications, such as the fraud detection problem. The extremely imbalanced data is a hard scenario in the imbalance problem and usually has a significant volume of data. Therefore, in this case, we not only have to learn patterns accurately of a very small positive sample but also need to process efficiently a very large dataset. Previous works only deal with one of these two problems separately. And even on the extreme imbalance data problem, there are only a few studies that lightly mentioned about this problem. In order to deal with new problem, we proposed to use two Machine Learning techniques, the undersampling and the ensemble learning. The undersampling is efficient in reducing the size of data in the case of the big dataset and also reduces the complexity of the imbalance problem in the case of extremely imbalanced data. In our study, we test our approach on various datasets with different characteristics from several domains. Our experimental results showed that a combination of the undersampling and the ensemble learning is not only efficient in extreme cases (i.e. the extreme imbalance of big data) but also similar to other approaches in normal cases (i.e. the slight imbalance of small data).

Secondly, we discussed the question of model selection in the portfolio selection topic. The global minimum variance portfolio requires an inverse of the covariance matrix and typically the sample covariance matrix is used. There are many covariance matrix estimations have been proposed to improve the sample covariance matrix, which is unstable and ill-conditioned in high-

dimensional space. They have comparable results but no estimator is the best on all performance metrics and on all the time. Therefore, they need a mechanism to select the best estimator for their optimal portfolio. We proposed to use the time-series Leave-One-Out Cross-Validation to estimate the portfolio performances of various covariance matrix estimations and then pick the best one. Considering parameters in those estimators as their second output, the parameters for the final estimator (i.e. the best estimator) are achieved by applying a regression voting ensemble. In our study, we consider three versions of the linear shrinkage covariance matrix estimations, and our data is from the Vietnam stock market. Our experimental results showed that our mechanism is better in most of performance metrics and only worse than others in one metric. It suggests that it could adapt quickly to the market regime by switching its estimation instead of using a fixed one.

Thirdly, we dealt with the outliers in the covariance matrix estimations. The covariance matrix, such as the sample covariance matrix, is sensitive to outliers, while the global minimum variance portfolio requires a stable covariance matrix. Especially in the high-dimensional space, an inverse of the covariance matrix will amplify the impact of outliers in the estimation. To improve the stability of the covariance matrix, we proposed to use under-sampling and ensemble learning to reduce the impact of the outliers on the output of covariance matrix estimation. The undersampling is used to create different samples, and each of them is used to build a covariance matrix by a given covariance matrix estimation. Then we use ensemble learning to combine those covariance matrices into one final covariance matrix. That final covariance matrix is used in the global minimum variance portfolio as usual, tested with the sample covariance matrix and three linear shrinkage covariance matrix estimations above. Our experimental results showed that this approach could improve the sample covariance matrix to the level of the shrinkage approach. An interesting finding is that the shrinkage to single-index model is less stable than the other two, but applying our approach also significantly improves this advanced estimation to the same level as other advanced shrinkage techniques.

6.2 Future Works

For further research, we suggest some research directions base on the limitations of our works above:

- In Chapter 3, our undersampling in the big data context is simply splitting the majority class, and we use all of them to build our models. In a more complex problem, i.e. complex extreme imbalance big data, our base models may not be easy to have high accuracy. Even with Ensemble Learning, our k-segments under-bagging approach may not work in that case. Therefore, we suggest improving the sampling process. For example, by calibrating probability in the undersampling,

- In Chapter 4, our model selection is to pick the best covariance matrix estimator. However, it makes our portfolio switch entirely from a previous "best" state to the new "best" state, and that causes our turnover to be higher than others. We suggest using a non-completely

(i.e. partial) switching mechanism such as a weighted average to switch the estimation slowly in order to reduce the portfolio turnover,

- Both Chapter 4 and Chapter 5 use the stock data from Vietnam stock market. Particularly, the data are from 2011 to the end of 2019 because we only focus on the normal market scenario. Market crises such as the 2008-2009 financial crisis or the COVID-19 pandemic are worth as separate studies. Moreover, our approach should be tested on different stock markets to verify our results.

List of publications

Conférences internationales (premier auteur)

Tuan TRAN, Loc Tran, An Mai. "*K-Segments Under Bagging approach: An experimental Study on Extremely Imbalanced Data Classification*". International Symposium on Communications and Information Technologies, 2019.
DOI: 10.1109/ISCIT.2019.8905145

Tuan TRAN, Nhat Nguyen, Trung Nguyen, An Mai. "*Voting shrinkage algorithm for Covariance Matrix Estimation and its application to portfolio selection*". International Conference on Computing and Communication Technologies, 2020.
DOI: 10.1109/RIVF48685.2020.9140764

Journal articles

Tuan TRAN, Nhat Nguyen, Trung Nguyen. "*k-Covariance: An Approach of Ensemble Covariance Estimation and Undersampling to Stabilize the Covariance Matrix in the Global Minimum Variance Portfolio*". Applied Sciences, 2022.
DOI: 10.3390/app12136403

Other publications outside the scope of this thesis

Nhat Nguyen, Trung Nguyen, **Tuan TRAN**, An Mai. "*Shrinkage Model Selection for Portfolio Optimization on Vietnam Stock Market*". Journal of Asian Finance, Economics and Business, 2020.
DOI: 10.13106/jafeb.2020.vol7.no9.135

Loc Tran, Linh Pham, **Tuan TRAN** and An Mai. "*Text classification problems via BERT embedding method and graph convolutional neural network*". International Conference on Advanced Technologies for Communications, 2021.
DOI: 10.1109/ATC52653.2021.9598337

Loc Tran, Bich Ngo, **Tuan TRAN**, Linh Pham and An Mai. "*On a development of sparse PCA method for face recognition problem*". International Conference on Advanced Technologies for Communications, 2021.
DOI: 10.1109/ATC52653.2021.9598326

Appendices

A Source code of the portfolios above

In this section, we provide a main code file for the portfolios mentioned in Chapter 4 and Chapter 5. This includes four covariance matrix estimations: the sample covariance matrix, the shrinkage to identity matrix, the shrinkage to single-index model, and the shrinkage to constant correlation model.

```
import pandas as pd
from pypfopt import risk_models

from utils.api import (
    get_universe,
    rebalance_portfolio,
    schedule_rebalance,
)

WINDOW = 2
WEEKLY_DATA = True
WEEKLY_REBALANCE = True
METHODS = [
    'sample',
    'single_index',
    'shrinkage_single_index',
]
METHOD = 'sample'


def initialize(context):
    context.frequency = 250
    context.window_length = WINDOW * context.frequency


def handle_data(context, data):
    if WEEKLY_REBALANCE:
        if not schedule_rebalance(context, data, date_rule='week_end'):
            return None

    universe = get_universe(context, data)
    df = data.history(universe, 'close', context.window_length, '1d')

    df = df.dropna(axis=1, thresh=250)
```

```
    universe = df.columns

    if WEEKLY_DATA:
        context.frequency = 52
        df['yearweek'] = df.index.map(lambda x: x.strftime('%Y%W'))
        df = df.groupby('yearweek').tail(1).drop('yearweek', axis=1)

    df = df.loc[:, df.apply(pd.Series.nunique) > 1]
    universe = df.columns

    if METHOD == 'sample':
        cov = risk_models.sample_cov(prices=df, frequency=context.frequency)
    elif METHOD == 'single_index':
        cov = shrinkage_single_index(
            x=df,
            shrink=1,
            frequency=context.frequency,
        )
    elif METHOD == 'shrinkage_single_index':
        cov, shrinkage = shrinkage_single_index(
            x=df,
            frequency=context.frequency,
        )

    max_weights = pd.Series(0.1, index=universe)
    weight_bounds = tuple([(0, max_weight) for max_weight in max_weights])
    weights = min_volatility(cov, weight_bounds)
    weights = pd.Series(weights, index=universe)
    weights = weights[weights > 1e-6]
    context.weights = weights.to_dict()

    rebalance_portfolio(context, data, context.weights)
```

B Source code of the Global Minimum Variance Portfolio

```
import numpy as np
import scipy.optimize as sco
```

```
def volatility(weights, cov_matrix, gamma=0):
    L2_reg = gamma * (weights**2).sum()
    portfolio_volatility = np.dot(weights.T, np.dot(cov_matrix, weights))
    return portfolio_volatility + L2_reg


def min_volatility(cov, weight_bounds):
    initial_guess = [0] * len(cov)
    constraints = [
        {
            "type": "eq",
            "fun": lambda x: np.sum(x) - 1,
        },
    ]
    args = (cov, 0)
    result = sco.minimize(
        volatility,
        x0=initial_guess,
        args=args,
        method="SLSQP",
        bounds=weight_bounds,
        constraints=constraints,
    )
    if not result["success"]:
        raise ValueError
    weights = result["x"]
    return weights
```

C Source code of the Shrinkage to the single-index model

```
import numpy as np
import pandas as pd


def shrinkage_single_index(x, shrink=None, frequency=252):
    """
    This estimator is a weighted average of the sample covariance matrix and a "pr
    Here, the prior is given by a one-factor model.
    The factor is equal to the cross-sectional average   of all the random variabl
```

```
    The notation follows Ledoit and Wolf (2003), version: 04/2014
    NOTE: use (pairwise) covariance on raw returns
    Parameters
    ----------
    x : T x N stock returns
    shrink : given shrinkage intensity factor if none, code calculates
    Returns
    -------
    tuple : np.ndarray which contains the shrunk covariance matrix
          : float shrinkage intensity factor

    """
    x = x.pct_change().dropna(how='all')

    t, n = np.shape(x)
    meanx = x.mean(axis=0)
    x = x - np.tile(meanx, (t, 1))
    xmkt = x.mean(axis=1).reshape(t, 1)

    sample = pd.DataFrame(np.append(x, xmkt, axis=1)).cov() * (t - 1) / t
    sample = sample.as_matrix()
    covmkt = sample[0:n, n].reshape(n, 1)
    varmkt = sample[n, n]
    sample = sample[:n, :n]
    prior = np.dot(covmkt, covmkt.T) * varmkt
    prior[np.eye(n) == 1] = np.diag(sample)

    if shrink == 1:
        return prior

    x = x.as_matrix()
    x = np.nan_to_num(x)

    if shrink is None:
        c = np.linalg.norm(sample - prior, "fro")**2
        y = x**2
        p = 1 / t * np.sum(np.dot(y.T, y)) - np.sum(sample**2)

        rdiag = 1 / t * np.sum(y**2) - sum(np.diag(sample)**2)
        z = x * np.tile(xmkt, (n, ))
        v1 = 1 / t * np.dot(y.T, z) - np.tile(covmkt, (n, )) * sample
```

```
        roff1 = (np.sum(v1 * np.tile(covmkt, (n, )).T) / varmkt -
                 np.sum(np.diag(v1) * covmkt.T) / varmkt)
        v3 = 1 / t * np.dot(z.T, z) - varmkt * sample
        roff3 = (np.sum(v3 * np.dot(covmkt, covmkt.T)) / varmkt**2 -
                 np.sum(np.diag(v3).reshape(-1, 1) * covmkt**2) / varmkt**2)
        roff = 2 * roff1 - roff3
        r = rdiag + roff

        k = (p - r) / c
        shrinkage = max(0, min(1, k / t))
    else:
        shrinkage = shrink

    sigma = shrinkage * prior + (1 - shrinkage) * sample
    sigma = sigma * frequency
    return sigma, shrinkage
```

D Source code of the Shrinkage to the constant correlation model

```
import numpy as np


def shrinkage_constant_correlation(x, shrink=None, frequency=252):
    """
    Shrinks towards constant correlation matrix
    if shrink is specified, then this constant is used for shrinkage

    The notation follows Ledoit and Wolf (2003, 2004) version 04/2014

    NOTE: use (pairwise) covariance on raw returns
    NOTE: shrink as float to return default behavior, as list to return
        different covariance of different shrinkage intensity

    Parameters
    ----------
    x : T x N stock returns
    shrink : given shrinkage intensity factor if none, code calculates

    Returns
    -------
    tuple : np.ndarray which contains the shrunk covariance matrix
          : float shrinkage intensity factor

    """
    x = x.pct_change().dropna(how='all')

    # de-mean returns
    t, n = np.shape(x)
    meanx = x.mean(axis=0)
    x = x - np.tile(meanx, (t, 1))

    # compute sample covariance matrix
    # sample = (1.0 / t) * np.dot(x.T, x)
    sample = x.cov().as_matrix()

    # NOTE: here we have to fillna since we have no assumption
```

```
    x = x.as_matrix()
    x = np.nan_to_num(x)

    # compute prior
    var = np.diag(sample).reshape(-1, 1)
    sqrtvar = np.sqrt(var)
    _var = np.tile(var, (n,))
    _sqrtvar = np.tile(sqrtvar, (n,))
    r_bar = (sum(sum(sample / (_sqrtvar * _sqrtvar.T))) - n) / (n * (n - 1))
    prior = r_bar * (_sqrtvar * _sqrtvar.T)
    prior[np.eye(n) == 1] = var.reshape(-1)

    # compute shrinkage parameters and constant
    if shrink is None:

        # what we call pi-hat
        y = x**2.0
        phi_mat = (
            np.dot(y.T, y) / t - 2 * np.dot(x.T, x) * sample / t + sample**2)
        phi = np.sum(phi_mat)

        # what we call rho-hat
        term1 = np.dot((x**3).T, x) / t
        help_ = np.dot(x.T, x) / t
        help_diag = np.diag(help_)
        term2 = np.tile(help_diag, (n, 1)).T * sample
        term3 = help_ * _var
        term4 = _var * sample
        theta_mat = term1 - term2 - term3 + term4
        theta_mat[np.eye(n) == 1] = np.zeros(n)
        rho = sum(np.diag(phi_mat)) + r_bar * np.sum(
            np.dot((1.0 / sqrtvar), sqrtvar.T) * theta_mat)

        # what we call gamma-hat
        gamma = np.linalg.norm(sample - prior, 'fro')**2

        # compute shrinkage constant
        kappa = (phi - rho) / gamma
        shrinkage = max(0.0, min(1.0, kappa / t))
    else:
        # use specified constant
```

```
    shrinkage = shrink

# compute the estimator
sigma = shrinkage * prior + (1 - shrinkage) * sample
sigma = sigma * frequency
return sigma, shrinkage
```

E Résumé

E.1 Introduction

E.1.1 Contexte

Les techniques d'apprentissage automatique (Machine Learning) ont gagné en importance dans le domaine de la finance en raison de l'augmentation de la puissance de calcul et de la disponibilité des données. Les chercheurs appliquent ces techniques pour obtenir des informations et faire des prédictions sur les marchés financiers, comme la prévision du cours des actions à l'aide de données historiques, d'états financiers ou pour l'analyse de sentiments. Cependant, les marchés financiers sont des systèmes complexes et chaotiques, rendant les prédictions précises difficiles.

Malgré les défis, l'apprentissage automatique offre des avantages comme le traitement de grands volumes de données et l'identification de tendances non apparentes pour les analystes humains. Il peut également améliorer l'efficacité de la prise de décision en automatisant des tâches. Les limites incluent le risque de sur-apprentissage et les problèmes d'interprétabilité. Les modèles d'apprentissage automatique ne peuvent faire des prédictions que sur la base des tendances apprises à partir des données, sans tenir compte des événements inattendus ou des changements de marché.

L'apprentissage automatique pour la finance, également connu sous le nom de science des données financières, vise à extraire de l'ordre des environnements financiers chaotiques en utilisant des approches basées sur les données. Un exemple est le classification de l'industrie basé sur les données, qui regroupe des entreprises similaires à l'aide de l'IA pour extraire des caractéristiques de diverses sources de données, palliant ainsi les limites des systèmes de classification industrielle traditionnels. Par exemple, éconophysique est un domaine interdisciplinaire résolvant des problèmes de finance et d'économie à l'aide de théories et de méthodes de la physique. L'une de ses applications est la théorie des matrices aléatoires, initialement développée en physique, qui a été appliquée pour identifier le bruit dans les matrices de corrélation financières et améliorer les estimations de la matrice de covariance pour l'optimisation de portefeuille.

La gestion de portefeuille, y compris l'allocation d'actifs, la construction de portefeuille, l'optimisation, la gestion des risques et la mesure de la performance, est un domaine crucial de la finance bénéficiant des progrès de l'apprentissage automatique financier. Cette thèse se concentre sur l'application d'approches statistiques et d'apprentissage automatique, comme le sous-échantillonnage et l'apprentissage ensemble, à l'optimisation du portefeuille de variance

minimale globale, démontrant leur efficacité pour améliorer la performance du portefeuille et mettant en évidence le potentiel d'autres progrès grâce à des méthodes avancées.

E.1.2 Défis et contributions scientifiques

Dans cette thèse, nous explorons l'application des techniques d'apprentissage automatique aux problèmes financiers, en nous concentrant sur trois domaines principaux. Les sujets spécifiques de notre recherche comprennent:

Premièrement, nous abordons le problème du déséquilibre extrême des classes dans la détection de fraude. Les ensembles de données de détection de fraude par carte de crédit présentent souvent des ratios de déséquilibre élevés, rendant les méthodes traditionnelles inefficaces. Nous proposons une technique de sous-échantillonnage combinée à l'apprentissage ensemble pour gérer ce déséquilibre extrême. En créant des sous-ensembles avec un déséquilibre plus faible et en utilisant un ensemble de vote, notre approche réduit le temps d'entraînement et maintient la précision, même avec de grands ensembles de données et des ratios de déséquilibre élevés.

Deuxièmement, nous abordons la sélection de modèle dans l'optimisation de portefeuille. Les portefeuilles moyenne-variance traditionnels, basés sur la théorie moderne du portefeuille de Markowitz, ont souvent de mauvaises performances pendant les krachs boursiers. Nous nous concentrons sur le portefeuille de variance minimale globale (GMVP), qui minimise le risque en utilisant uniquement la matrice de covariance. Pour améliorer la stabilité de la matrice de covariance, nous employons des techniques de rétrécissement et proposons une méthode de validation croisée pour sélectionner le meilleur estimateur. Nos expériences sur le marché boursier vietnamien démontrent que cette approche s'adapte bien aux changements de marché et améliore la performance du portefeuille.

Enfin, nous abordons le problème des valeurs aberrantes dans les estimations de la matrice de covariance pour le GMVP. Les valeurs aberrantes peuvent avoir un impact significatif sur la stabilité de la matrice de covariance. Nous utilisons le sous-échantillonnage et l'apprentissage ensemble pour atténuer ce problème. En créant plusieurs ensembles de données hebdomadaires à partir des rendements quotidiens des actions et en estimant la matrice de covariance pour chacun, nous combinons ces matrices pour réduire les fluctuations. Nos résultats montrent que cette méthode améliore la performance de l'estimation de la matrice de covariance de l'échantillon au niveau des estimations de rétrécissement avancées.

E.2 S'adapter à l'Extreme Imbalance: Une combinaison d'Undersampling et d'Ensemble Learning pour la Big Data Classification

E.2.1 Introduction

Les jeux de données déséquilibrés sont un problème répandu dans divers domaines du monde réel, tels que la détection de fraude et la détection du cancer. Le défi consiste à identifier un petit nombre de points de données positifs (classe minoritaire) parmi un grand nombre de points de données négatifs (classe majoritaire). Par exemple, dans une tâche de classification avec un ratio de déséquilibre (IR) de 99, seul 1 échantillon sur 100 est un échantillon positif. Cela pose un défi

important pour les algorithmes de Machine Learning, qui peuvent atteindre une précision élevée en prédisant simplement la classe majoritaire, classant ainsi incorrectement tous les échantillons minoritaires. Plusieurs études ont rapporté une dégradation des performances avec des jeux de données déséquilibrés (C. Chen et al., 2004; X.-w. Chen et al., 2005; J. Wang et al., 2006; Hong et al., 2007).

Pour relever ce défi, plusieurs méthodes ont été proposées, qui peuvent être largement catégorisées en niveaux algorithmiques et de données. Au niveau algorithmique, de nouveaux classificateurs sont conçus ou des algorithmes existants sont modifiés pour gérer les données déséquilibrées (Bradford et al., 1998; Cieslak et al., 2008). Au niveau des données, des techniques de prétraitement sont appliquées aux données déséquilibrées d'origine avant d'utiliser des algorithmes de classification standard. Les trois techniques les plus courantes sont l'under-sampling, l'over-sampling (Drummond et al., 2003), et la Synthetic Minority Over-sampling Technique (SMOTE) (N. V. Chawla, Bowyer, et al., 2002). Des études ont montré que l'utilisation d'un ensemble d'entraînement équilibré avec des algorithmes standard peut améliorer les performances (Weiss and F. Provost, 2001; Laurikkala, 2001; Estabrooks et al., 2004). Une autre approche est l'approche sensible aux coûts, qui vise à minimiser les coûts de classification erronée, en particulier pour la classe minoritaire, mais nécessite des informations supplémentaires sur les coûts (Ling, Q. Yang, et al., 2004).

L'avancement rapide des technologies informatiques a conduit à une augmentation exponentielle du volume de données, comme en biologie génomique ou dans les systèmes bancaires. Les Big Data, composées souvent de millions ou de milliards d'enregistrements, posent un défi important pour les solutions traditionnelles, en particulier dans le contexte de la classification de données déséquilibrées (Del Rio et al., 2014; Triguero, Galar, Vluymans, et al., 2015; Fernandez et al., 2017). Les méthodes traditionnelles se sont révélées inefficaces pour traiter de tels grands ensembles de données en raison de contraintes de ressources ou de mauvaises performances. Récemment, le problème des données déséquilibrées dans le contexte des Big Data a reçu plus d'attention (Triguero, Rio, et al., 2015; Krawczyk, 2016).

Dans de nombreuses applications pratiques, les jeux de données sont non seulement volumineux mais aussi fortement déséquilibrés, comme dans la détection de fraude où le ratio de déséquilibre est souvent supérieur à 1000 (Juszczak et al., 2008; Shuhao Wang et al., 2017; W. Yang et al., 2019; Mekterović et al., 2021; X. Zhang et al., 2021). Ce scénario de classification de Big Data extrêmement déséquilibrées nécessite une solution efficace qui aborde simultanément ces deux problèmes. Dans cette étude, nous démontrons qu'une combinaison de la technique d'under-sampling et d'Ensemble Learning peut efficacement gérer ce problème. Notre méthode proposée offre une solution prometteuse pour les grands jeux de données déséquilibrés, fournissant des performances améliorées par rapport aux approches traditionnelles.

E.2.2 Problème de déséquilibre dans les données traditionnelles

L'un des principaux défis en data mining et en machine learning est le problème des données déséquilibrées, courant dans des applications comme la détection de fraude dans les appels téléphoniques (Fawcett et al., 1997) et les transactions par carte de crédit (Chan et al., 1999).

Dans ces cas, la classe minoritaire (par exemple, les transactions frauduleuses) est largement surpassée en nombre par la classe majoritaire (par exemple, les transactions légitimes). Par exemple, dans un jeu de données où seulement 1% des données appartient à la classe minoritaire, un classificateur qui prédit toujours la classe majoritaire atteindrait une précision de 99% mais échouerait à détecter toute instance de la classe minoritaire. Cela souligne la nécessité de méthodes efficaces pour aborder la classification des données déséquilibrées.

Les méthodes pour traiter les données déséquilibrées peuvent être catégorisées en approches au niveau des données et au niveau des algorithmes (N. V. Chawla, Japkowicz, et al., 2004). Les méthodes au niveau des données incluent l'échantillonnage, l'ensemble, les techniques basées sur les coûts, basées sur la distance, et hybrides. Les méthodes d'échantillonnage équilibrent le jeu de données en supprimant ou en répliquant des points de données. Les méthodes d'ensemble combinent plusieurs classificateurs pour améliorer les performances. Les méthodes basées sur les coûts ajustent la taille de l'échantillon de chaque classe en fonction des coûts associés. Les méthodes basées sur la distance réduisent la distance entre les classes minoritaires et majoritaires. Les méthodes hybrides combinent différentes techniques pour équilibrer le jeu de données.

Les approches au niveau des algorithmes incluent des classificateurs spécifiquement conçus pour les données déséquilibrées et des classificateurs sensibles aux coûts qui attribuent différents coûts aux erreurs de classification. Les méthodes wrapper peuvent convertir des classificateurs insensibles aux coûts en classificateurs sensibles aux coûts en ajustant les seuils de décision.

Les méthodes sensibles aux coûts au niveau des données et des algorithmes abordent les données déséquilibrées en attribuant différents coûts aux classes minoritaires et majoritaires (Zadrozny et al., 2003). Au niveau des données, les techniques basées sur les coûts ajustent les tailles d'échantillons pour équilibrer les coûts. Au niveau des algorithmes, des fonctions de perte spécifiques minimisent les coûts. Les méthodes wrapper ajustent les seuils de décision pour convertir les classificateurs insensibles aux coûts en classificateurs sensibles aux coûts (Domingos, 1999). Ces méthodes se sont révélées efficaces dans diverses applications.

E.2.3 Problème de déséquilibre dans les big data

Les ensembles de données déséquilibrés sont un problème courant dans divers domaines du monde réel, tels que la détection de fraude et la détection du cancer. Le défi consiste à identifier un petit nombre de points de données positifs (classe minoritaire) parmi un grand nombre de points de données négatifs (classe majoritaire). Par exemple, dans une tâche de classification avec un ratio de déséquilibre (IR) de 99, seul 1 échantillon sur 100 est un échantillon positif. Cela pose un défi important pour les algorithmes de Machine Learning, qui peuvent atteindre une précision élevée en prédisant simplement la classe majoritaire, classant ainsi incorrectement tous les échantillons minoritaires. Plusieurs études rapportent une dégradation des performances avec des ensembles de données déséquilibrés (C. Chen et al., 2004; X.-w. Chen et al., 2005; J. Wang et al., 2006; Hong et al., 2007).

Pour relever ce défi, plusieurs méthodes sont proposées, qui peuvent être largement catégorisées en niveaux algorithmiques et de données. Au niveau algorithmique, de nouveaux classifieurs sont conçus ou des algorithmes existants sont modifiés pour gérer les données déséquili-

brées Bradford et al., 1998; Cieslak et al., 2008. Au niveau des données, des techniques de prétraitement sont appliquées aux données déséquilibrées d'origine avant d'utiliser des algorithmes de classification standard. Les trois techniques les plus courantes sont l'under-sampling, l'over-sampling (Drummond et al., 2003), et la Synthetic Minority Over-sampling Technique (SMOTE) (N. V. Chawla, Bowyer, et al., 2002). Des études montrent que l'utilisation d'un ensemble d'entraînement équilibré avec des algorithmes standard peut améliorer les performances (Weiss and F. Provost, 2001; Laurikkala, 2001; Estabrooks et al., 2004). Une autre approche est l'approche cost-sensitive, qui vise à minimiser les coûts de classification erronée, en particulier pour la classe minoritaire, mais nécessite des informations supplémentaires sur les coûts (Ling, Q. Yang, et al., 2004).

L'avancement rapide des technologies informatiques a conduit à une augmentation exponentielle du volume de données, comme en biologie génomique ou dans les systèmes bancaires. Les big data, composées souvent de millions ou de milliards d'enregistrements, posent un défi important pour les solutions traditionnelles, en particulier dans le contexte de la classification des données déséquilibrées (Del Rio et al., 2014; Triguero, Galar, Vluymans, et al., 2015; Fernandez et al., 2017). Les méthodes traditionnelles se révèlent inefficaces pour traiter de tels grands ensembles de données en raison de contraintes de ressources ou de mauvaises performances. Récemment, le problème des données déséquilibrées dans le contexte des big data a reçu plus d'attention (Triguero, Rio, et al., 2015; Krawczyk, 2016).

Dans de nombreuses applications pratiques, les ensembles de données sont non seulement volumineux mais aussi très déséquilibrés, comme dans la détection de fraude où le ratio de déséquilibre est souvent supérieur à 1000 (Juszczak et al., 2008; Shuhao Wang et al., 2017; W. Yang et al., 2019; Mekterović et al., 2021; X. Zhang et al., 2021). Ce scénario de classification de big data extrêmement déséquilibrées nécessite une solution efficace qui aborde simultanément ces deux problèmes. Dans cette étude, nous démontrons qu'une combinaison de la technique d'under-sampling et d'ensemble learning peut efficacement gérer ce problème. Notre méthode proposée offre une solution prometteuse pour les grands ensembles de données déséquilibrés, fournissant des performances améliorées par rapport aux approches traditionnelles.

E.2.4 Travaux récents sur le déséquilibre extrême dans la classification des big data

Les chercheurs font face à des défis importants lorsqu'ils traitent des ensembles de données extrêmement déséquilibrés, en particulier lorsque le ratio de déséquilibre dépasse 100, ce qui signifie que la classe minoritaire constitue moins de 0,9901% de l'ensemble de données (Tang et al., 2009; Triguero, Rio, et al., 2015). Ce problème est exacerbé dans les contextes de big data en raison de la taille même des ensembles de données. Peu d'études ont abordé la classification des big data extrêmement déséquilibrées, soulevant des questions sur l'efficacité des méthodes traditionnelles dans de tels scénarios. Nous définissons un ensemble de données comme des big data extrêmement déséquilibrées s'il a un ratio de déséquilibre supérieur à 100 et contient plus de 100 000 instances. Les ensembles de données avec un ratio de déséquilibre supérieur à 50 sont considérés comme hautement déséquilibrés.

Dans les grands ensembles de données avec un déséquilibre de classe important, il est crucial de considérer la distribution des classes positives et négatives. Chai et al. (2013) ont exploré le déséquilibre de classe dans la classification de textes de dossiers médicaux en utilisant le Random Under-Sampling pour équilibrer les classes. Leur ensemble de données comprenait 516 000 échantillons et 85 650 attributs, avec seulement 0,3% des échantillons dans la classe positive (ratio de déséquilibre de 333). Ils ont utilisé la Regularized Logistic Regression en raison de sa capacité à gérer de nombreuses caractéristiques sans surapprentissage. L'under-sampling a augmenté le Recall mais diminué la Precision, le score F1 restant relativement inchangé. L'étude n'a pas expliqué pourquoi des ratios de classe égaux avec l'under-sampling étaient préférés ou pourquoi l'under-sampling a été choisi plutôt que d'autres méthodes, ni examiné l'impact de différents ratios de classe sur les performances. Les résultats étaient spécifiques aux données médicales et non généralisables à d'autres domaines.

Del Rio et al. (2014) ont évalué diverses méthodes pour traiter le déséquilibre de classe en utilisant le framework MapReduce avec Apache Hadoop et Apache Mahout, en employant des random forests comme classifieur de base. Les techniques évaluées comprenaient le Random Over-Sampling (ROS), le Random Under-Sampling (RUS), SMOTE, et une variante cost-sensitive de Random Forests. Les ensembles de données allaient de 435 000 à 5 700 000 points de données avec des ratios de déséquilibre allant jusqu'à 77 670. Aucune méthode n'a systématiquement surpassé les autres, et la meilleure méthode dépendait du nombre de mappers utilisés, contredisant d'autres études. L'applicabilité de l'étude pourrait être améliorée en utilisant des frameworks big data plus adaptables comme Apache Spark.

Galpert et al. (2015) ont étudié le déséquilibre de classe dans l'identification d'orthologues parmi les espèces de levure. Ils ont comparé plusieurs approches big data, notamment un algorithme Random Forest cost-sensitive avec MapReduce (RF-BDCS), ROS avec Random Forest utilisant MapReduce (ROS+RF-BD), et SVM dans Apache Spark combiné avec ROS (ROS+SVM-BD). Les ensembles de données allaient de 8 millions à 29,9 millions d'instances avec des ratios de déséquilibre de 1 630 à 10 520. Les techniques supervisées ont surpassé les techniques non supervisées, ROS+SVM-BD obtenant les meilleures performances (AUC de 0,885 et GM de 0,879). L'inclusion de l'algorithme SMOTE dans la comparaison aurait fourni une vue plus complète.

Les occurrences de défaillances peu fréquentes dans les processus de fabrication étendus créent également un environnement de classe déséquilibré. Hebert (2016) ont comparé les performances de Random Forest (RF) et XGBoost avec la logistic regression sur un ensemble de données d'environ 1 180 000 instances et 4 264 caractéristiques, avec un ratio de déséquilibre de 170. Les classifieurs basés sur les arbres ont surpassé la logistic regression mais présentent des inconvénients, tels que la difficulté à comprendre les interactions entre paramètres. D'autres classifieurs, comme les k plus proches voisins et les réseaux de neurones, pourraient également être comparés pour évaluer les performances des systèmes linéaires et non linéaires.

E.2.5 Méthodologie

E.2.5.1 K-Segments Under Bagging (K-SUB)

Dans cette section, nous proposons une approche pour traiter le déséquilibre extrême dans les big data en combinant l'undersampling et l'ensemble learning. Notre méthode garantit que chaque point de données de la classe majoritaire n'est utilisé qu'une seule fois. Initialement, nous appliquons une technique d'undersampling qui divise aléatoirement tous les points de données négatifs en plusieurs sous-ensembles disjoints de taille égale. Chaque sous-ensemble est ensuite fusionné avec tous les points de données positifs pour entraîner un classifieur. Enfin, ces classifieurs sont combinés à l'aide d'un ensemble de vote pour produire la prédiction finale. Cette approche est illustrée dans la Figure 3.2. Contrairement aux méthodes existantes qui utilisent des schémas d'échantillonnage répété ou de pondération, notre approche assure une contribution égale de chaque point de données de la classe majoritaire, réduisant ainsi le risque de surapprentissage.

Considérons un ensemble de données D avec N points de données et un ratio de déséquilibre IR. Nous désignons le nombre de sous-ensembles par K. Soit D^{major} et D^{minor} représentant respectivement les classes majoritaire et minoritaire. Pour garantir que chaque point de données de la classe majoritaire n'est utilisé qu'une seule fois, chaque sous-ensemble est échantillonné pour avoir une taille de $\left\lfloor \frac{\mid D^{major} \mid}{K} \right\rfloor$. Nous utilisons principalement l'algorithme Random Forest en raison de sa robustesse et de ses performances élevées. Le processus détaillé est résumé dans l'Algorithme 3.

E.2.5.2 Mesure d'évaluation

Dans les problèmes de déséquilibre, de nombreuses mesures d'évaluation ne sont pas significatives lorsqu'elles sont utilisées indépendamment car elles sont influencées par la classe majoritaire. Une mesure appropriée qui équilibre les classes est le score F_1, dérivé de la matrice de confusion (voir Figure 2.6). Le score F_1, moyenne harmonique de la Precision et du Recall, est calculé par l'Équation 3.8. Par conséquent, nous utilisons le score F_1 comme principale mesure d'évaluation dans cette étude.

E.2.5.3 Ensembles de données

Pour démontrer l'efficacité de notre algorithme proposé, nous l'appliquons à 11 ensembles de données différents, variant en taille et en ratios de déséquilibre. Dans chaque expérience, nous ajustons le paramètre K pour obtenir la meilleure précision. Nous utilisons une validation croisée stratifiée à 5 plis, calculant le score F_1 comme la moyenne des cinq plis. Le temps d'exécution est rapporté avec une précision de trois décimales, montrant que notre algorithme est efficace en termes de temps. Ce test est effectué sur une seule machine avec 24 cœurs et 128 gigaoctets de mémoire. Les résultats finaux sont résumés dans le Tableau 3.5.

E.2.6 Résultats et discussions

Les résultats en gras indiquent les meilleurs scores F_1 obtenus en utilisant Under Bagging (UB), RUSBoost, et notre méthode K-Segments Under Bagging (K-SUB) avec des valeurs de K ajustées. Pour les ensembles de données avec un faible ratio de déséquilibre (IR < 50), UB et RUSBoost performent de manière adéquate, tandis que K-SUB montre des résultats légèrement meilleurs. Dans les cas de déséquilibre élevé à extrême (IR ≥ 50 et IR ≥ 100), K-SUB surpasse significativement à la fois UB et RUSBoost. De plus, K-SUB s'exécute beaucoup plus rapidement, ne prenant que quelques minutes sur la même machine. Dans les plus grands ensembles de données avec le plus haut IR, les scores F_1 d'UB tombent à zéro, mais K-SUB maintient de bons scores F_1.

E.2.7 Conclusion

Ce chapitre présente l'approche K-Segments Under Bagging (K-SUB) pour la classification de données extrêmement déséquilibrées. Les résultats expérimentaux montrent que K-SUB non seulement surpasse les méthodes précédentes dans les cas de déséquilibre extrême, mais s'exécute également rapidement. Pour les données à faible déséquilibre, K-SUB reste compétitif avec les méthodes de pointe lorsque K est convenablement ajusté. Cela démontre qu'une combinaison simplifiée d'undersampling et d'ensemble learning peut donner de meilleurs résultats que des méthodes plus complexes. De plus, K peut toujours être ajusté pour une précision acceptable, ce qui est crucial pour les applications industrielles. Les travaux futurs devraient explorer davantage cette approche et aborder les problèmes connexes dans les données extrêmement déséquilibrées et de haute dimension.

E.3 Ensemble de vote pour les estimations de matrices de covariance à rétrécissement linéaire dans l'optimisation de portefeuille

E.3.1 Introduction

En finance quantitative, la sélection de portefeuille est une méthode permettant de choisir le meilleur portefeuille parmi toutes les options possibles en fonction d'un objectif spécifique qui peut être modélisé mathématiquement. Un objectif courant est de minimiser la variance, ce qui signifie que nous essayons de réduire le risque de notre portefeuille. Ce type de portefeuille est idéal pour les investisseurs qui recherchent le risque le plus faible pour un rendement donné. Sur le long terme, il génère souvent d'excellents rendements (Baker et al., 2011). Avec la puissance croissante des ordinateurs, de nombreuses études en finance quantitative utilisent maintenant une combinaison de techniques mathématiques et de science des données, rendant cette approche populaire dans la sélection de portefeuille (Deboeck, 1994; Y. Li et al., 2016).

L'objectif principal de la recherche sur les portefeuilles à variance minimale est d'estimer avec précision la matrice de covariance. Les méthodes traditionnelles comme la matrice de covariance de l'échantillon (SCM) présentent de nombreux problèmes, en particulier dans la sélection de portefeuille de haute dimension. La haute dimensionnalité peut entraîner des erreurs et des données insuffisantes pour estimer la véritable matrice de covariance. Cela conduit souvent

à une matrice de covariance mal conditionnée ou singulière, ce qui est un problème courant en calcul matriciel. En conséquence, les portefeuilles basés sur la matrice de covariance de l'échantillon ont souvent de mauvaises performances et ne parviennent pas à générer des profits.

Pour résoudre ce problème, des estimateurs de rétrécissement pour l'estimation de la matrice de covariance ont été développés. Cette approche équilibre le biais et la variance de l'estimation de la matrice de covariance en utilisant une matrice cible prédéfinie et bien conditionnée $\mathbf{F}$ et une intensité de rétrécissement δ qui peut être calculée à partir de la matrice de covariance de l'échantillon et $\mathbf{F}$. Cependant, la plupart des recherches sur l'amélioration des méthodes de rétrécissement se concentrent sur des formulations mathématiques, avec peu d'études adoptant une approche basée sur les données. Ce chapitre propose une méthode adaptative pour améliorer la sélection de l'intensité de rétrécissement en combinant le cadre de rétrécissement original avec la technique de validation croisée leave-one-out. Nous concevons et menons également des expériences pour démontrer l'efficacité de notre algorithme proposé dans la sélection de portefeuille en utilisant des données du marché boursier vietnamien.

E.3.2 Travaux connexes

En apprentissage automatique, la Cross-Validation (CV) est une technique courante pour évaluer les modèles sur des données hors échantillon, souvent utilisée pour la sélection de variables et le réglage des paramètres. En finance et en économie, la CV a été appliquée pour stabiliser les rendements des actions et évaluer les modèles prédictifs. Par exemple, Conway et al. (1988) ont utilisé la CV pour identifier des facteurs stables dans les rendements des actions, tandis que Upton (1994) a amélioré cette méthode pour l'analyse statistique et économique. Picard et al. (1984) ont utilisé la CV pour évaluer la capacité prédictive des modèles de régression, et L. Zhang (2012) ont appliqué la CV pour étudier la relation entre les volatilités implicites et réalisées. Arlot et al. (2010) ont examiné les performances de la CV en matière de sélection de modèles, suggérant des orientations pour les recherches futures. Bergmeir et al. (2018) ont démontré l'utilité de la CV dans la prévision des séries temporelles, montrant qu'elle peut contrôler le surajustement.

En finance quantitative, l'estimation de la matrice de covariance est cruciale pour des applications telles que la sélection de portefeuille. Les méthodes traditionnelles rencontrent des problèmes lorsque le nombre d'actifs est égal au nombre de points de données. Des estimateurs de rétrécissement ont été proposés pour réduire les erreurs d'estimation. Golosnoy and Okhrin (2007) ont introduit une approche de rétrécissement multivarié pour les portefeuilles basés sur Markowitz, et Golosnoy and Okhrin (2009) ont proposé un estimateur de rétrécissement flexible. Olivier Ledoit et al. (2003), Olivier Ledoit et al. (2004b), and Olivier Ledoit et al. (2004a) ont développé une méthode de rétrécissement linéaire pour les matrices de covariance de grande dimension, qui est devenue une référence dans la sélection de portefeuille. Ils ont étendu ce travail en utilisant des transformations non linéaires des valeurs propres (Olivier Ledoit et al., 2010). DeMiguel, Martin-Utrera, et al. (2013) ont passé en revue les méthodes de rétrécissement et introduit de nouvelles techniques pour les moyennes, les matrices de covariance et les poids de portefeuille. Candelon et al. (2012) ont proposé une méthode de double rétrécissement pour

améliorer la stabilité de l'estimation avec de petits échantillons en utilisant la régression ridge. Malgré ces avancées, il existe encore un besoin de recherche exploitant la CV pour améliorer l'analyse de la sélection de portefeuille.

E.3.3 Notre approche proposée

E.3.3.1 Intensité de rétrécissement dans l'estimation de la covariance

Dans la théorie moderne du portefeuille, la sélection d'un bon portefeuille d'actifs implique souvent de considérer un portefeuille à variance minimale globale avec N actions, où les poids $\mathbf{w} = (w_1, w_2, ..., w_N)$ somment à un ($\Sigma_{i=1}^{N} w_i = 1$) et $w_i > 0$ (pas de vente à découvert). Le problème de sélection de portefeuille est défini comme suit:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^T \mathbf{\Sigma} \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^T \mathbf{1} = 1 \\ & w_i > 0 \quad \forall i = \overline{1, N} \end{aligned} \tag{1}$$

où $\mathbf{1}$ est un vecteur de uns, et $\mathbf{\Sigma}$ est la matrice de covariance des prix des actions. La solution analytique est:

$$\mathbf{w}_* = \mathbf{\Sigma}^{-1} \mathbf{1} \left(\mathbf{1}^T \mathbf{\Sigma}^{-1} \mathbf{1} \right)^{-1}. \tag{2}$$

Comme la vraie matrice de covariance $\mathbf{\Sigma}$ est inconnue, nous utilisons la matrice de covariance de l'échantillon $\mathbf{S}$, qui peut ne pas être bien conditionnée ou inversible. Pour résoudre ce problème, nous utilisons des estimateurs de shrinkage comme proposé par Olivier Ledoit et al. (2003):

$$\hat{\mathbf{\Sigma}} = \delta \mathbf{F} + (1 - \delta) \mathbf{S} \tag{3}$$

où $\delta \in [0, 1]$ est l'intensité de shrinkage, et $\mathbf{F}$ est une matrice cible. Différentes matrices cibles $\mathbf{F}$ conduisent à différentes estimations de shrinkage, telles que Shrinkage towards market (**SSIM**), Shrinkage towards constant correlation (**SCCM**), et Shrinkage towards identity matrix (**STIM**). Nous proposons une approche basée sur les données pour adapter δ aux fluctuations des données, appelée estimateur de voting shrinkage.

E.3.3.2 Principale mesure d'évaluation

Nous utilisons le Sharpe ratio comme principale mesure d'évaluation, défini comme:

$$SR = \frac{R - R_f}{\sigma} \tag{4}$$

où R est le rendement et σ est la volatilité d'un portefeuille:

$$R = ((\mathbf{R} + 1) \cdot \mathbf{1})^{\frac{252}{|\mathbf{R}|}} - 1, \tag{5}$$

$$\sigma = \sqrt{252 \mathrm{Var}[\mathbf{R}]}, \tag{6}$$

avec $\mathbf{R}$ comme vecteur de rendement quotidien du portefeuille et R_f comme taux sans risque (supposé être zéro). Un Sharpe ratio plus élevé indique une meilleure compensation pour le risque.

E.3.3.3 Algorithme de voting pour la sélection de l'intensité de Shrinkage

Nous proposons une estimation de l'intensité de voting shrinkage basée sur la Cross-Validation leave-one-out. Étant donné les données de prix $[D(t)]_{(V+W+1)\times N}$ pour $t \in [t_0 - V - W, t_0]$, avec une fenêtre glissante W et une fenêtre de validation V, nous sélectionnons la meilleure intensité de shrinkage δ qui maximise le Sharpe ratio sur V points de test. L'algorithme renvoie la moyenne de ces intensités comme intensité de voting pour la construction du portefeuille. Le processus est résumé dans l'Algorithme 0.

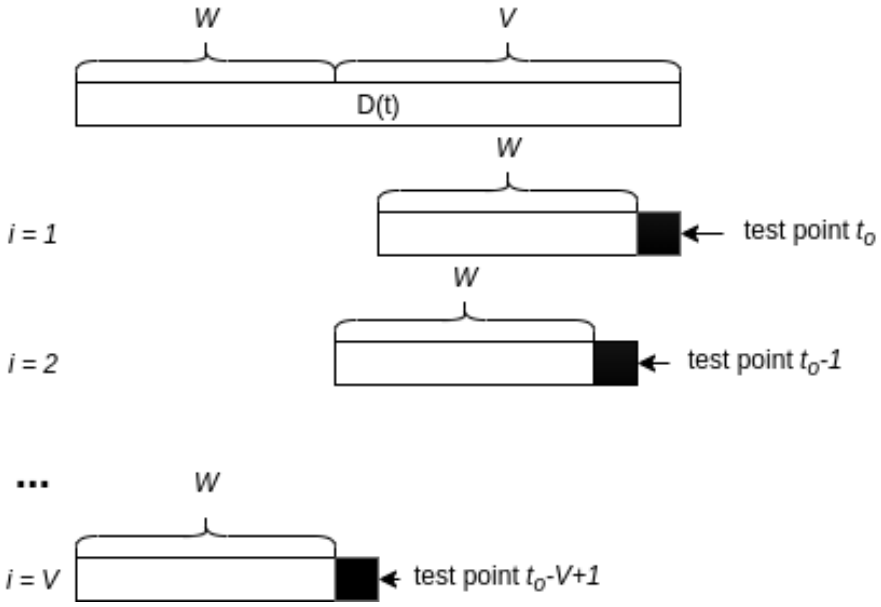


Figure E.1: Une visualisation intuitive du processus de Cross-Validation en série temporelle dans notre approche. À partir d'une série temporelle de données boursières $D(t)$, nous utilisons W points de données avant un point de test $t_0 - i + 1$ pour construire un portefeuille puis évaluer sa performance sur le point de test. Il y a V plis pour la validation, c'est-à-dire V points de test.

Algorithm 5 Intensité de rétrécissement par vote basée sur le cadre de Ledoit-Wolf (Voting-LW)

Entrée: $D(t)$ pour $t \in [t_0 - V - W, t_0]$, t_0, W, V, $M_{\mathbf{F}}$.
Sortie: Intensité de rétrécissement Voting-LW $\hat{\delta}^*$

```
Δ ← liste vide d'intensités de rétrécissement
for i = 1: V do
    Sélectionner un sous-ensemble de D(t) de t_0 − i − W à t_0 − i: D_i ← D(t), |_[t_0−i−W, t_0−i]
    r = D(t_0 − i + 1) diag^{−1}(D(t_0 − i)) − 1
    for j = 1: M_F do
        δ_i^j ← LW_j(D_i)
        Ri^j = r · ŵ ∗ (δ_i^j, D_i)
    end for
end for
for j = 1: M_F do
    SR^j = SR(R_1^j, ..., R_V^j)
end for
Voting: j_* = arg max_{j=1:M_F} (SR^j)
Retourner δ̂* = (1/V) Σ_{i=1}^{V} δ_i^{j_*}.
```

E.3.4 Résultats expérimentaux

E.3.4.1 Données

Pour nos expériences, nous utilisons les prix quotidiens des actions vietnamiennes de la Ho Chi Minh City Stock Exchange (HOSE) de janvier 2011 à octobre 2019. Le jeu de données comprend $T = 2275$ observations, avec une fenêtre glissante $W = 250$ et des périodes de validation $V = 250$. Nous avons rencontré des erreurs d'ingestion de données, telles que des prix/volumes manquants et des jours consécutifs avec des volumes nuls. Après avoir mis à jour les données quotidiennement, nous avons fait correspondre les prix et volumes des actions avec d'autres sources et utilisé des techniques d'imputation. Nous n'avons considéré que les actifs avec au moins 250 points de données quotidiens pour garantir la qualité de l'analyse.

E.3.4.2 Mesures de performance du portefeuille

Pour mettre en évidence les avantages de notre algorithme, nous le comparons aux méthodes de Ledoit-Wolf en utilisant cinq indicateurs: Annual Return, Annual Volatility, Sharpe Ratio, Portfolio Turnover, et Alpha.

E.3.4.3 Annual Return & Volatility, Sharpe Ratio Le rendement annuel (R) et la volatilité (σ) sont calculés à partir des rendements du portefeuille en utilisant les Équations 4.6 et 4.7. Le ratio de Sharpe est le ratio du rendement excédentaire à la volatilité, comme décrit dans la Section 4.3.2.

E.3.4.4 Portfolio Turnover Selon DeMiguel, Garlappi, et al. (2009), le turnover du portefeuille mesure la stabilité des poids du portefeuille dans le temps. Un turnover plus faible est préférable car il réduit les risques et les coûts de transaction. Il est défini comme suit:

$$PT = \frac{1}{L-1} \sum_{t=1}^{L-1} \sum_{i=1}^{N} \left(|w_{t+1,i} - w_{t,i}| \right) \tag{7}$$

où $w_{t,i}$ est le poids de l'actif i à l'instant t.

E.3.4.5 Alpha L'alpha, ou alpha de Jensen (Jensen, 1968), mesure la performance du portefeuille par rapport à un benchmark. Dans notre cas, le VNIndex est le benchmark. Un $\alpha = 1\%$ indique que le portefeuille surperforme le marché de 1%. Il est calculé comme suit:

$$\alpha = 252\mathrm{E}[(\mathbf{R} - \mathbf{R}_f) - \beta(\mathbf{R}_m - \mathbf{R}_f)], \tag{8}$$

avec β donné par:

$$\beta = \frac{\mathrm{Cov}[\mathbf{R} - \mathbf{R}_f, \mathbf{R}_m]}{\mathrm{Var}[\mathbf{R} - \mathbf{R}_f]} \tag{9}$$

où $\mathbf{R}_f$ et $\mathbf{R}_m$ sont les vecteurs de taux sans risque et de rendement de référence, respectivement.

E.3.4.6 Analyse des résultats

Nous comparons notre algorithme Voting-LW avec trois méthodes de Ledoit-Wolf (SSIM, SCCM, STIM) en utilisant cinq mesures de performance. Un portefeuille idéal a des rendements plus élevés, une volatilité plus faible, un ratio de Sharpe plus élevé, un turnover plus faible et un alpha plus élevé. Nos résultats, présentés dans les Tableaux 4.1, 4.2, et 4.3, indiquent que Voting-LW surpasse les benchmarks dans au moins quatre indicateurs à travers différents univers d'actions ($N = [50, 100, 200]$). Voting-LW montre des rendements améliorés, une volatilité compétitive ou plus faible, et des ratios de Sharpe plus élevés, conduisant à l'alpha positif le plus élevé. Cependant, son turnover est moins stable, potentiellement en raison de la période de rééquilibrage fixe de notre système de backtesting. Ajuster les points de rééquilibrage pourrait réduire le turnover.

E.3.5 Conclusions

Dans ce chapitre, nous présentons un algorithme de vote qui s'adapte aux fluctuations des données réelles pour améliorer la sélection de l'intensité de rétrécissement dans l'estimation de la matrice de covariance. Notre approche utilise la technique de validation croisée leave-one-out pour sélectionner indirectement la matrice cible de l'estimation de rétrécissement avec le ratio Sharpe le plus élevé. Les résultats expérimentaux montrent que notre méthode de vote obtient des résultats remarquables par rapport à trois méthodes d'estimation de rétrécissement de pointe, basées sur cinq indicateurs financiers courants. Pour les recherches futures, nous

prévoyons d'étendre notre travail sur la sélection de cibles en utilisant des ensembles de validation aléatoires pour soutenir notre système de backtesting dans la sélection de périodes appropriées pour les transactions boursières.

E.4 Estimation de covariance d'ensemble pour le Global Minimum Variance Portfolio

E.4.1 Introduction

Après la théorie fondatrice de sélection de portefeuille de H. M. Markowitz (1968), le calcul de portefeuilles efficaces vise à estimer les poids optimaux des actifs en utilisant la moyenne et la covariance des rendements des actions. Cependant, l'erreur d'estimation des rendements attendus est plus importante que celle de la covariance (Merton, 1980). Des études récentes suggèrent de supposer des rendements attendus égaux pour toutes les actions et d'utiliser uniquement la matrice de covariance pour estimer le portefeuille, connu sous le nom de Global Minimum Variance Portfolio (GMVP) (DeMiguel and Nogales, 2009). L'estimation statistique standard de la matrice de covariance dans le GMVP est l'estimation de la covariance échantillonnale, qui présente plusieurs inconvénients dans le trading de portefeuille.

Lorsque le nombre d'actifs dépasse le nombre de rendements historiques, la matrice de covariance échantillonnale devient non inversible et mal conditionnée, entraînant des erreurs d'estimation élevées et des ajustements fréquents du portefeuille, augmentant le risque et réduisant le profit. Une approche pour rendre la matrice de covariance inversible est la technique de shrinkage, qui combine la matrice de covariance échantillonnale avec une autre matrice inversible (Stein, 2020). Des études récentes (Tong et al., 2018; T. Tran, N. Nguyen, T. Nguyen, and Mai, 2020) utilisent la Cross-Validation pour évaluer l'intensité de shrinkage "optimale", résultant en différentes intensités basées sur des objectifs prédéfinis.

Au-delà du shrinkage linéaire, Olivier Ledoit et al. (2012) and Olivier Ledoit et al. (2015) ont proposé une méthode de shrinkage non linéaire utilisant les valeurs propres de la population estimées à partir de la distribution des valeurs propres de l'échantillon. Bien qu'efficace, cette méthode est complexe. Une méthode non linéaire plus simple utilise les valeurs propres d'un sous-ensemble de validation comme valeurs propres de la population (Lam, 2016). Une autre extension implique l'utilisation de plusieurs matrices cibles pour éviter la dépendance à une seule matrice cible (Gray et al., 2018).

Une approche plus simple remet en question les avantages des estimateurs de shrinkage en utilisant un portefeuille d'estimateurs de covariance (Jagannathan et al., 2000; Bengtsson et al., 2002), qui fait la moyenne de différents estimateurs pour neutraliser les erreurs. Disatnik et al. (2007) montrent que des portefeuilles simples performent de manière similaire aux portefeuilles de shrinkage complexes dans les tests hors échantillon, suggérant qu'il n'y a pas d'amélioration statistique avec des estimateurs complexes.

Les gestionnaires de fonds recherchent des alternatives au meilleur estimateur de covariance pour étendre la capacité de leur fonds. Ces alternatives, bien que n'étant pas les meilleures, offrent des performances comparables et soulèvent des questions sur l'efficacité de l'estimation par

shrinkage. Des résultats durables sont cruciaux, car des changements fréquents de portefeuille dus aux valeurs aberrantes peuvent entraîner des coûts de transaction et des risques de liquidité.

Cette étude examine la stabilité de plusieurs matrices de covariance dans l'optimisation de portefeuille pour aider les investissements durables. Nous explorons si la force de l'estimation par shrinkage provient de la combinaison de matrices ou du calcul optimal de l'intensité de shrinkage, et si la combinaison avec d'autres matrices de covariance échantillonnales est faisable. Comprendre ces aspects est vital tant pour les mathématiques théoriques que pour la finance pratique.

Nous proposons d'utiliser l'Ensemble learning avec l'Undersampling pour réduire l'impact des valeurs aberrantes et stabiliser la matrice de covariance estimée. En testant cette approche sur le marché boursier vietnamien de 2013 à 2019, nous constatons qu'elle obtient des résultats comparables ou meilleurs que les techniques de shrinkage. Notre portefeuille surpasse significativement la matrice de covariance échantillonnale sur toutes les mesures de performance.

E.4.2 Contexte

Le contenu donné traite de l'estimation par shrinkage linéaire pour le problème du Global Minimum Variance Portfolio (GMVP). Le GMVP vise à trouver le vecteur de poids $\mathbf{w}$ qui minimise la variance du portefeuille $\mathbf{w}^\intercal\mathbf{\Sigma}\mathbf{w}$, sous la contrainte $\mathbf{w}^\intercal\mathbf{1} = 1$, où $\mathbf{\Sigma}$ est la matrice de covariance.

L'estimateur de la matrice de covariance échantillonnale peut être mal conditionné, surtout lorsque le nombre d'actifs N dépasse le nombre d'observations T. Pour résoudre ce problème, Ledoit et Wolf ont proposé l'estimation par shrinkage linéaire, qui combine la matrice de covariance échantillonnale $\mathbf{S}$ avec une matrice cible bien conditionnée $\mathbf{F}$ en utilisant une intensité de shrinkage δ:

$$\widehat{\Sigma} = \delta\mathbf{F} + (1 - \delta)\mathbf{S} \tag{10}$$

Le δ optimal minimise la perte quadratique entre $\mathbf{F}$ et $\mathbf{S}$. Ledoit et Wolf ont suggéré trois matrices cibles: la matrice identité, le modèle à indice unique et le modèle à corrélation constante, qui surpassent empiriquement la matrice de covariance échantillonnale.

E.4.2.1 Ensemble learning et Undersampling

La matrice de covariance est sensible aux valeurs aberrantes, en particulier dans les matrices de haute dimension estimées à partir de données limitées (Yuan et al., 2001; Leys et al., 2018; Raymaekers et al., 2021; Ke et al., 2019). Cette sensibilité conduit à l'instabilité. Les valeurs aberrantes affectent également divers problèmes de Machine Learning, suscitant plusieurs approches pour résoudre ce problème. Par exemple, Breiman (2001) proposent d'utiliser l'Undersampling et l'Ensemble learning pour créer un estimateur robuste à partir de plusieurs estimateurs faibles, qui sont des fonctions à faible biais mais à forte variance. Ils utilisent l'Undersampling pour générer de nombreux estimateurs décorrélés, puis les combinent pour réduire la variance de la

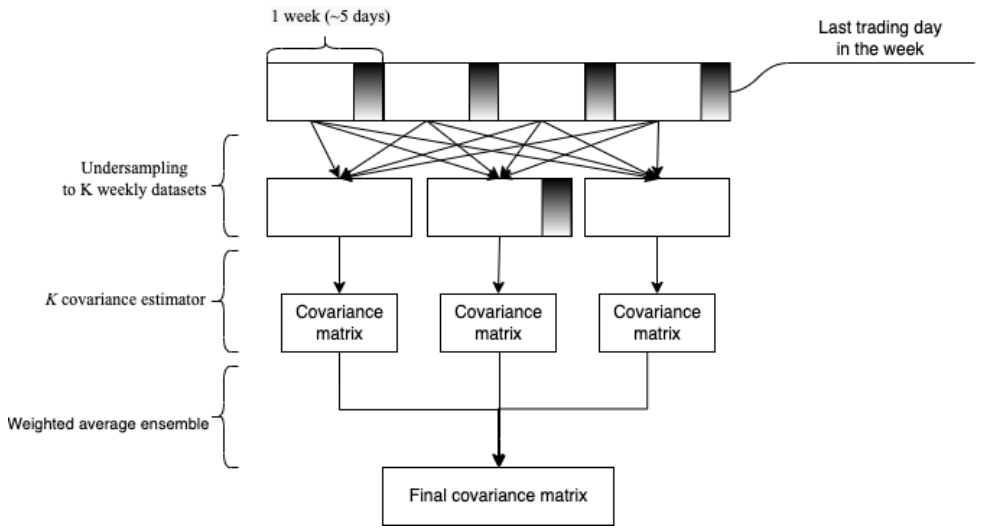
prédiction finale. La règle de combinaison peut être une simple moyenne pour la régression ou un vote majoritaire pour la classification. Cette méthode est simple et efficace pour de nombreux problèmes. Ses avantages comprennent: i) l'exploitation des forces de chaque estimateur, et ii) l'atténuation de l'impact des valeurs aberrantes dans les données d'entrée ou les prédictions individuelles sur l'estimateur final. Lorsqu'on traite un petit nombre de valeurs aberrantes dans une grande matrice de covariance, la détection des valeurs aberrantes fait face au défi du déséquilibre entre les valeurs aberrantes et les valeurs non aberrantes. T. Tran, L. Tran, et al. (2019) démontrent que la combinaison de l'Undersampling avec l'Ensemble learning est efficace non seulement sur de grands ensembles de données, mais aussi dans des cas fortement déséquilibrés. Ils suggèrent de diviser les données originales en k segments avec le même ratio de déséquilibre, de construire k estimateurs, puis de les combiner en un prédicteur final. Ces segments diffèrent mais partagent une sémantique de données similaire, à l'instar des données boursières utilisées pour l'estimation de la covariance.

E.4.3 Méthodologie

L'objectif du GMVP est de minimiser la variance du portefeuille, ce qui nécessite une matrice de covariance stable dans le temps. Typiquement, l'estimation de la covariance utilise des données hebdomadaires dérivées des données quotidiennes en sélectionnant le dernier jour de bourse de chaque semaine. Cependant, des données anormales lors de ces derniers jours peuvent conduire à des valeurs aberrantes dans la matrice de covariance. Pour atténuer ce problème, nous proposons d'utiliser l'undersampling et l'ensemble learning pour l'estimation de la covariance au niveau des données. Notre procédure d'échantillonnage consiste à sélectionner aléatoirement un jour de bourse chaque semaine, plutôt que le dernier jour, pour créer k sous-ensembles hebdomadaires. Chaque sous-ensemble est utilisé pour estimer une matrice de covariance, servant de prédicteur de covariance.

Ce processus d'échantillonnage vise à sélectionner un point de données proche chaque semaine avec des informations similaires au dernier jour. Les prédicteurs de covariance construits sur ces sous-ensembles ont une sémantique similaire mais des valeurs aberrantes différentes. Nous combinons ces prédicteurs en utilisant une moyenne simple pour neutraliser les valeurs aberrantes, ce qui donne une matrice de covariance plus stable. Cette procédure est illustrée dans la Figure E.2.

Figure E.2: Visualisation de notre approche K-covariance. Les ensembles de données hebdomadaires traditionnels utilisent le dernier jour de bourse de chaque semaine pour l'estimation de la covariance. Dans la K-covariance, nous sélectionnons aléatoirement un jour de bourse chaque semaine. À partir de ces ensembles de données hebdomadaires, nous estimons K matrices de covariance en utilisant n'importe quel estimateur de covariance donné. Nous combinons ensuite ces matrices en une matrice de covariance finale en utilisant un ensemble de moyenne pondérée. Cette matrice est utilisée dans l'optimisation du portefeuille GMVP.



Nous menons une expérience empirique comparant notre méthode avec des estimateurs de covariance à shrinkage linéaire et des portefeuilles d'estimateurs de covariance. Alors que les combinaisons dans les portefeuilles d'estimateurs de covariance peuvent être considérées comme un ensemble au niveau de la méthode, notre approche se situe au niveau des données. Notre expérience inclut onze portefeuilles avec différents estimateurs de covariance:

- Matrice de covariance de l'échantillon (sample)
- Shrinkage towards identity matrix (STIM)
- Shrinkage to single-index model (SSIM)
- Shrinkage to constant correlation matrix (SCCM)
- Combinaison de la matrice de l'échantillon, de la matrice diagonale et du modèle à indice unique (sTS)
- Combinaison de la matrice de l'échantillon, du modèle à indice unique et de la matrice de corrélation constante (sSC)
- Combinaison de la matrice de l'échantillon, de la matrice diagonale et de la matrice de corrélation constante (sTC)
- Notre méthode appliquée à l'estimateur de covariance échantillonné (k-sample), STIM (k-STIM), SSIM (k-SSIM) et SCCM (k-SCCM)

Dans notre processus de backtesting, nous construisons un GMVP à la fin de chaque semaine en utilisant les dernières données et l'exécutons le premier jour de bourse de la semaine suivante. Nous conservons le portefeuille jusqu'au week-end, puis le recalculons et le rééquilibrons. La période hors échantillon est d'une semaine, et la période dans l'échantillon est de deux ans de données hebdomadaires. Nous n'utilisons que les actifs ayant au moins un an de données non nulles. Pour évaluer les portefeuilles, nous suivons les réglementations et restrictions réelles du marché. Nous appliquons des frais de commission réels sur chaque transaction, utilisons le taux d'intérêt à long terme des obligations d'État dans les calculs de performance, et limitons le nombre maximum d'actions négociées chaque jour. Nous supposons que nous pouvons acheter/vendre à découvert jusqu'au volume réel négocié dans l'ensemble de données et qu'il n'y a pas de glissement dans les transactions. Pour éviter le biais d'anticipation, nous n'utilisons que les données historiques jusqu'à la fin de chaque date, et les ordres correspondent aux prix de la date suivante.

E.4.4 Ensemble de données et métriques d'évaluation

E.4.4.1 Ensemble de données

Cette étude évalue notre approche et d'autres portefeuilles sur le marché boursier vietnamien, un marché émergent en Asie. Nous utilisons les données historiques de trading de la Ho Chi Minh City Stock Exchange (HOSE), disponibles sur leur site web. L'expérience s'étend sur sept ans, de 2013 à 2019, couvrant 1744 jours de bourse. La vente à découvert n'étant pas autorisée, nous nous concentrons sur les portefeuilles longs uniquement. Le Tableau 5.1 fournit un résumé statistique des données, avec le volume moyen en 10^5 actions et les autres métriques en points de base.

Table E.1: Résumé statistique des données historiques sur la bourse HOSE de 2013 à 2019.

	2013	2014	2015	2016	2017	2018	2019
Nombre minimum d'actions	304	302	303	308	324	350	380
Nombre maximum d'actions	316	310	313	324	350	380	387
Rendement quotidien moyen	16	13	6,23	5,26	10,44	-1,79	2,42
Écart-type	815,88	276,2	354,13	283,13	255,54	275,92	249,2
Volume moyen	4,54	9,03	8,12	8,88	11,84	11,19	9,66

E.4.4.2 Métriques d'évaluation

Nous utilisons trois métriques de performance de portefeuille courantes: Annual Volatility, Sharpe Ratio et Portfolio Turnover. Annual Volatility (σ) est l'écart-type des rendements du portefeuille, indiquant le risque. Une volatilité plus faible est préférable pour le GMVP. Le Sharpe Ratio, développé par Sharpe (1994), mesure le rendement d'un investissement par rapport à son risque:

$$\text{Sharpe ratio} = \frac{R - R_f}{\sigma}, \tag{11}$$

où R est le rendement du portefeuille et R_f est le taux sans risque, utilisant l'obligation d'État vietnamienne à 10 ans. Un Sharpe Ratio plus élevé indique un meilleur portefeuille. Le Portfolio Turnover mesure la stabilité des poids du portefeuille dans le temps (DeMiguel, Garlappi, et al., 2009). Un turnover plus faible est préféré pour le GMVP. La formule est:

$$\text{Portfolio Turnover} = \frac{1}{T-1} \sum_{t=1}^{T-1} \sum_{i=1}^{N} \left(|\mathbf{w}t + 1, i - \mathbf{w}t, i| \right) \tag{12}$$

où $\mathbf{w}_{t,i}$ est le poids de l'actif i à la date t. La valeur maximale est 2 (changement de 200%), et la minimale est 0%. Nous ne rapportons que les résultats hors échantillon pour ces métriques et fournissons des valeurs p pour chaque paire d'estimateurs de matrice de covariance. Pour le Sharpe Ratio, nous utilisons une méthode de bootstrap suggérée par Oliver Ledoit et al. (2008) pour un test d'hypothèse de performance robuste.

E.4.5 Résultats et discussions

Dans le Tableau 5.2, nous présentons la performance hors échantillon de onze portefeuilles en utilisant trois métriques d'évaluation. Les résultats des tests d'hypothèse pour chaque paire de portefeuilles sont montrés dans le Tableau 5.4 pour la volatilité annuelle, le Tableau 5.5 pour le ratio de Sharpe, et le Tableau 5.6 pour le turnover du portefeuille. De même, le Tableau 5.3 rapporte la performance des mêmes portefeuilles mais limitée aux 100 premiers actifs par capitalisation boursière. Les p-valeurs correspondantes pour ces portefeuilles sont fournies dans les Tableaux 5.7, 5.8, et 5.9.

Le portefeuille échantillon performe significativement moins bien que tous les autres portefeuilles sur toutes les métriques, soulignant l'instabilité de la matrice de covariance échantillon. Les portefeuilles utilisant des estimateurs de rétrécissement linéaire ne montrent pas de différences significatives en termes de volatilité et de ratio de Sharpe, mais le portefeuille SSIM a un turnover plus élevé comparé aux portefeuilles STIM et SCCM.

Nos résultats indiquent que l'Undersampling et l'Ensemble learning appliqués à l'estimation de la matrice de covariance échantillon peuvent améliorer significativement la performance du portefeuille en termes de volatilité annuelle et de ratio de Sharpe, comparable aux estimations de covariance par rétrécissement optimal. Cependant, le turnover du portefeuille reste plus élevé que celui des portefeuilles de rétrécissement linéaire.

Les portefeuilles de rétrécissement performent généralement de manière similaire, sauf que le portefeuille SSIM a un turnover plus élevé que les portefeuilles STIM et SCCM. L'application de notre approche de rétrécissement au modèle à indice unique (k-SSIM) réduit significativement le turnover du portefeuille SSIM à des niveaux comparables aux portefeuilles STIM et SCCM, suggérant que la matrice de covariance SSIM est moins stable.

La combinaison de plus de deux matrices différentes résulte en le turnover de portefeuille le plus bas et surpasse souvent les autres méthodes. Le portefeuille sTS, combinant la matrice

échantillon, la matrice diagonale, et le modèle à indice unique, obtient les meilleurs résultats dans la plupart des cas. La matrice diagonale combine efficacement le portefeuille échantillon avec un portefeuille pondéré de manière égale, améliorant la diversification.

La comparaison des portefeuilles de rétrécissement optimal avec d'autres estimateurs montre des résultats similaires, indiquant que la force de la technique de rétrécissement réside dans la combinaison de différentes matrices, avec une intensité de rétrécissement optimale comme raffinement supplémentaire. Les combinaisons de multiples matrices de covariance échantillon utilisant des techniques d'Ensemble learning et d'Undersampling produisent des performances comparables aux estimations de rétrécissement avancées.

Table E.2: Performances hors échantillon de onze portefeuilles de variance minimale globale avec différentes estimations de matrice de covariance. Toutes les actions disponibles sur la bourse HOSE sont considérées.

Estimateur	Annual Volatility (%)	Sharpe ratio	Portfolio Turnover (%)
sample	8.30	1.39	7.28
STIM	7.24	1.94	3.49
SSIM	7.37	2.00	4.41
SCCM	7.71	1.94	3.43
k-sample	7.80	1.77	5.62
k-STIM	7.13	2.04	3.34
k-SSIM	7.17	2.07	3.86
k-SCCM	7.52	1.91	3.23
sTS	6.79	2.24	2.88
sTC	7.46	2.01	2.89
sSC	7.47	2.01	2.90

Table E.3: Performance hors échantillon de onze portefeuilles de variance minimale globale avec différentes estimations de matrice de covariance. Seuls les cent premiers actifs par capitalisation boursière sont considérés ($N = 100$).

Estimateur	Annual Volatility (%)	Sharpe ratio	Portfolio Turnover (%)
sample	9.81	0.82	4.30
STIM	9.34	0.99	3.17
SSIM	9.54	0.92	3.78
SCCM	9.42	1.09	3.18
k-sample	9.47	0.82	3.67
k-STIM	9.16	0.93	2.91
k-SSIM	9.31	0.86	3.32
k-SCCM	9.29	1.01	2.90
sTS	8.90	1.13	2.47
sTC	9.28	1.10	2.59
sSC	9.28	1.10	2.59

E.4.6 Conclusions

Dans ce chapitre, nous présentons l'estimation de la k-covariance en utilisant l'Undersampling et l'Ensemble learning pour améliorer la stabilité de la matrice de covariance dans le trading quantitatif, en particulier pour le portefeuille de variance minimale globale. Notre méthode réduit l'impact des valeurs aberrantes en manipulant le niveau des données. Tous les éléments de nos combinaisons sont des matrices de covariance homogènes, estimées à partir d'échantillons différents. Les résultats expérimentaux montrent que l'approche k-covariance améliore significativement l'estimation de la matrice de covariance échantillon, atteignant le niveau de performance des estimations de rétrécissement linéaire. Bien que les estimations de rétrécissement linéaire soient robustes, notre approche montre des améliorations dans certains cas, comme le rétrécissement vers le modèle à indice unique.

La nouveauté scientifique de cette recherche réside dans l'utilisation de techniques de Machine Learning pour construire et combiner plusieurs estimations faibles de matrice de covariance, obtenant une estimation durable de la matrice de covariance. Nous concluons que la combinaison de plusieurs matrices de covariance échantillon avec notre approche, et de manière similaire en les appliquant à une estimation de rétrécissement faible, atteint les niveaux de performance les plus élevés des estimations de rétrécissement. L'amélioration des estimations de rétrécissement provient principalement de la combinaison avec d'autres matrices, tandis que notre approche combine plusieurs fois un type d'estimation de matrice de covariance, qui est faible et bruyante. De plus, les portefeuilles d'estimations de covariance qui combinent la matrice de covariance échantillon avec plus de deux autres matrices montrent des résultats exceptionnels. Nous proposons d'investiguer ces portefeuilles dans les travaux futurs.

E.5 Conclusions

Dans cette thèse, nous explorons l'ensemble learning et d'autres techniques de Machine Learning, telles que l'undersampling et la cross-validation, appliquées à divers domaines, y compris l'industrie financière. Notre recherche aborde trois questions principales, divisées en deux parties: les applications générales de Machine Learning et les applications financières.

Premièrement, nous identifions un écart entre les données extrêmement déséquilibrées et les big data, courant dans des applications comme la détection de fraude. Ce scénario implique l'apprentissage de modèles à partir d'un petit échantillon positif au sein d'un grand ensemble de données. Les études précédentes ont abordé ces problèmes séparément. Nous proposons de combiner l'undersampling et l'ensemble learning pour traiter simultanément les deux problèmes. L'undersampling réduit la taille et la complexité des données, tandis que l'ensemble learning améliore la précision. Nos expériences sur divers ensembles de données montrent que cette combinaison est efficace à la fois pour les big data extrêmement déséquilibrées et les petites données légèrement déséquilibrées.

Deuxièmement, nous abordons la sélection de modèles dans l'optimisation de portefeuille. Le Global Minimum Variance Portfolio nécessite une matrice de covariance inverse, généralement estimée à l'aide de la matrice de covariance de l'échantillon, qui est instable dans les espaces de haute dimension. Divers estimateurs existent, mais aucun n'est universellement optimal. Nous proposons d'utiliser la Leave-One-Out Cross-Validation en série temporelle pour évaluer la performance du portefeuille à travers différents estimateurs et sélectionner le meilleur. Les paramètres de l'estimateur final sont déterminés à l'aide d'un Voting Ensemble de régression. Les tests sur les données du marché boursier vietnamien montrent que notre méthode surpasse les autres sur la plupart des métriques, s'adaptant rapidement aux changements du marché.

Troisièmement, nous nous attaquons aux valeurs aberrantes dans les estimations de la matrice de covariance. Les valeurs aberrantes peuvent déstabiliser la matrice de covariance de l'échantillon, cruciale pour le Global Minimum Variance Portfolio. Dans les espaces de haute dimension, cette instabilité est amplifiée. Nous proposons d'utiliser l'undersampling pour créer différents échantillons, chacun générant une matrice de covariance, qui sont ensuite combinés en utilisant l'ensemble learning. Cette approche stabilise la matrice de covariance, l'améliorant au niveau des techniques de shrinkage avancées. Nos expériences montrent des améliorations significatives, en particulier pour le modèle à indice unique moins stable.

Dans l'ensemble, nos contributions démontrent l'efficacité de la combinaison de l'undersampling et de l'ensemble learning pour relever les défis du Machine Learning et des applications financières.

RÉSUMÉ

Ensemble Learning est une méthode puissante pour améliorer les performances des modèles d'apprentissage automatique en combinant les prédictions de plusieurs modèles de base. L'idée derrière l'apprentissage d'ensemble est qu'en combinant les points forts de différents modèles de base, l'ensemble dans son ensemble peut obtenir de meilleures performances que n'importe quel modèle de base unique. Des études empiriques ont montré que la méthode d'ensemble est particulièrement efficace lorsque les modèles de base sont diversifiés, un exemple réussi étant les arbres de décision aléatoires. En raison de ses avantages, Ensemble Learning est largement utilisé dans diverses applications, notamment les problèmes de détection de fraude. Plus en détail, les avantages d'Ensemble Learning tiennent à deux points principaux : i) l'ensemble combine les points forts de ses modèles de base, rendant chaque modèle complémentaire l'un de l'autre, et ii) il neutralise le bruit et les valeurs aberrantes parmi tous les modèles de base, réduire leur impact sur les prévisions finales. Dans cette thèse, nous utilisons ces deux idées d'Ensemble Learning pour différentes applications dans l'apprentissage automatique et l'industrie financière. Nos principales contributions dans cette thèse sont triples. Tout d'abord, nous démontrons comment l'apprentissage d'ensemble et les techniques de sous-échantillonnage peuvent être utilisés pour traiter efficacement le scénario difficile du problème de déséquilibre des données dans le domaine de l'apprentissage automatique, en particulier dans le cas de mégadonnées extrêmement déséquilibrées. Deuxièmement, nous proposons de manière appropriée l'utilisation de la validation croisée des séries chronologiques et de l'apprentissage d'ensemble pour résoudre un problème de sélection d'estimateurs de matrice de covariance dans le commerce quantitatif. Enfin, nous montrons comment l'apprentissage d'ensemble peut être utilisé pour réduire l'impact des valeurs aberrantes dans les estimations de la matrice de covariance, augmentant ainsi la stabilité des portefeuilles. Dans l'ensemble, nos recherches mettent en évidence le potentiel de l'apprentissage d'ensemble pour améliorer les performances de diverses applications dans le domaine de l'apprentissage automatique et de la finance.

MOTS CLÉS

apprentissage d'ensemble, sous-échantillonnage, validation croisée, estimation de matrice de covariance

ABSTRACT

Ensemble Learning is a powerful method for improving the performance of machine learning models by combining the predictions of multiple base models. The idea behind ensemble learning is that by combining the strengths of different base models, the ensemble as a whole can achieve better performance than any single base model. Empirical studies have shown that the ensemble method is particularly effective when the base models are diversified, one successful example is random decision trees. Because of its advantages, Ensemble Learning is widely used in various applications, including fraud detection problems. In more detail, the advantages of Ensemble Learning are due to two main points: i) the ensemble combines the strengths of its base models, making each model complementary to one another, and ii) it neutralizes the noise and outliers among all base models, reducing their impact on the final predictions. In this thesis, we use these two ideas of Ensemble Learning for different applications in the machine learning and the finance industry. Our main contributions in this thesis are threefold. Firstly, we demonstrate how ensemble learning and undersampling techniques can be used to efficiently deal with the hard scenario of imbalance data problem in the machine learning field, particularly in the case of extremely imbalance big data. Secondly, we propose appropriately the use of time-series cross-validation and ensemble learning to resolve a covariance matrix estimator selection problem in quantitative trading. Lastly, we show how ensemble learning can be used to reduce the impact of outliers in covariance matrix estimations, thereby increasing the stability of portfolios. Overall, our research highlights the potential of ensemble learning for improving the performance of various applications in the field of machine learning and finance.

KEYWORDS

ensemble learning, undersampling, cross-validation, covariance matrix estimation