Feature selection for semi-supervised data analysis in decisional information systems

Mohammed Hindawi

2013ISAL0015

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES - LYON

ÉCOLE DOCTORALE INFOMATHS

INFORMATIQUE ET MATHEMATIQUES

T H È S E

présentée pour obtenir le grade de

DOCTEUR DE L'INSA DE LYON

Spécialité : INFORMATIQUE

par

Mohammed HINDAWI

Sélection de Variables pour l'Analyse Semi-Supervisée des Données dans les Systèmes d'Information Décisionnels

soutenue publiquement le 21, Février, 2013

devant le jury :

Président :	Mohamed NADIF	-	Pr. Université Paris 5
Rapporteurs :	Younès BENNANI	-	Pr. Université Paris 13
	Yann GUERMEUR	-	DR. CNRS (LORIA - Nancy)
Examinateur :	Yves LECHEVALLIER	-	DR. INRIA (Rocquencourt)
Directeurs :	Khalid BENABDESLEM	-	MCF. Université Lyon 1
	Alexandre AUSSEM	-	Pr. Université Lyon 1
	Jean-François BOULICAUT	-	Pr. INSA de Lyon

NATIONAL INSTITUTE OF APPLIED SCIENCES - LYON

DOCTORAL SCHOOL INFOMATHS

INFORMATIQUE ET MATHEMATIQUES

PHD THESIS

submitted in partial fulfillment for the degree of

Doctor of Philosophy

in the National Institute of Applied Sciences - Lyon

Specialty : COMPUTER SCIENCE

by

Mohammed HINDAWI

Feature Selection for Semi-Supervised Data Analysis in Decisional Information Systems

defended on February 21, 2013

before the committee :

President :	Mohamed NADIF	-	Prof. University of Paris 5
Reviewers :	Younès BENNANI	-	Prof. University of Paris 13
	Yann GUERMEUR	-	DR. CNRS (LORIA - Nancy)
Examiner :	Yves LECHEVALLIER	-	DR. INRIA (Rocquencourt)
Advisors :	Khalid BENABDESLEM	-	Assoc. Prof. University of Lyon 1
	Alexandre AUSSEM	-	Prof. University of Lyon 1
	Jean-François BOULICAUT	-	Prof. INSA of Lyon.

المعهد الوطني للعلوم التطبيقية - ليون

مدرسة الدكتوراة إنفومات
المعلوماتية والرياضيات

أطروحة دكتوراة

قدمت للحصول على درجة

دكتور في الهندسة المعلوماتية

تقديم

محمد هنداوي

اختيار المواصفات بشبه إشراف بهدف تحليل البيانات في أنظمة معلومات اتخاذ القرار

تم الدفاع عنها علناً بتاريخ ٢١ شباط ٢٠١٣ أمام اللجنة المؤلفة من :

السيد محمد نظيف - أستاذ جامعي (جامعة باريس ٥) - رئيس اللجنة
السيد يونس بنّاني - أستاذ جامعي (جامعة باريس ١٣) - مراجع
السيد يان غورمور - مدير أبحاث (المركز الوطني للبحوث العلمية - لوريا) - مراجع
السيد إيف لوشوفالييه - مدير أبحاث (المعهد الوطني لبحوث المعلوماتية والأتمتة - روكونكور) - فاحص
السيد خالد بن عبد السلام - أستاذ جامعي مساعد (جامعة ليون ١) - الأستاذ المشرف
السيد ألكسندر أوسم - أستاذ جامعي (جامعة ليون ١) - المشرف المساعد
السيد جان فرانسوا بوليكو - أستاذ جامعي (المعهد الوطني للعلوم التطبيقية - ليون) - المشرف المساعد

To the honest rebels who are elminating the "noise" that had always affected the performance of syrian people...

— Mohammed

To the “values” that were always “relevant”
And never “redundant” in my life...
My parents, my sisters, my wife and my children

Acknowledgements

First of all, I am grateful to **The Almighty God** for establishing me to complete this work.

I wish to express my sincere thanks to the panel of expert examiners before whom I defended this work for their questions and remarks that enriched the work. In detail, I wish to thank Mr. Younès Bennani, Professor at University of Paris 13 and Mr. Yann Guermeur, Research Director at CNRS-LORIA for their reports that highlighted the ideas and contribution of this work. I also thank Mr. Mohamed Nadif, Professor at University of Paris 5 for accepting to preside the defense committee. Besides, I would like to thank Mr. Yves Lechevallier, Research Director at INRIA-Rocquencourt for accepting to examine my work.

I would like to express my sincere gratitude to my advisors Mr. Alexandre Aussem, Professor at University of Lyon 1, and Mr. Jean-François Boulicaut, Professor at INSA de Lyon for their help especially at the first and last phases of my PhD. Besides, I am deeply indebted to my supervisor, Mr. Khalid Benabdeslem, Associate professor at University of Lyon 1 who has helped me shape my research, and who has always been supportive and patient throughout the whole period of my study until the very last day before defense. I would like also to express my gratitude to Mr. Haytham Elghazel, associate professor at University of Lyon 1 for his endless support and encouragement since the very first day of my study. I would also like to thank all my colleagues at LIRIS laboratory for creating such an enjoyable working environment.

I owe a debt of gratitude to Mr. Mazen Said, Professor at University of Aleppo, for believing in me, for supporting me during the whole period of my studies,

and for cheering me up whenever I felt uneasy.

Finally, I am particularly indebted to my family, without whom I would never have done anything. I would like to thank my parents and my sisters who helped me fulfilling my dreams. Besides, I would like to thank my wife and my children who helped my establishing new dreams

Villeurbanne, February 25, 2013 Mohammed Hindawi

Résumé

La sélection de variables est une tâche primordiale en fouille de données et apprentissage automatique. Il s'agit d'une problématique très bien connue par les deux communautés dans les contextes, supervisé et non-supervisé. Le contexte semi-supervisé est relativement récent et les travaux sont embryonnaires. Récemment, l'apprentissage automatique a bien été développé à partir des données partiellement labélisées. La sélection de variables est donc devenue plus importante dans le contexte semi-supervisé et plus adaptée aux applications réelles, où l'étiquetage des données est devenu plus couteux et difficile à obtenir.

Dans cette thèse, nous présentons une étude centrée sur l'état de l'art du domaine de la sélection de variable en s'appuyant sur les méthodes qui opèrent en mode semi-supervisé par rapport à celles des deux contextes, supervisé et non-supervisé. Il s'agit de montrer le bon compromis entre la structure géométrique de la partie non labélisée des données et l'information supervisée de leur partie labélisée. Nous nous sommes particulièrement intéressés au «small labeled-sample problem» où l'écart est très important entre les deux parties qui constituent les données.

Pour la sélection de variables dans ce contexte semi-supervisé, nous proposons deux familles d'approches en deux grandes parties. La première famille est de type «Filtre» avec une série d'algorithmes qui évaluent la pertinence d'une variable par une fonction de score. Dans notre cas, cette fonction est basée sur la théorie spectrale de graphe et l'intégration de contraintes qui peuvent être extraites à partir des données en question. La deuxième famille d'approches est de type «Embedded» où la sélection de variable est intrinsèquement liée à un modèle d'apprentissage. Pour ce faire, nous proposons des algorithmes à base de pondération de variables dans un paradigme de classification automa-

tique sous contraintes. Deux visions sont développées à cet effet, (1) une vision globale en se basant sur la satisfaction relaxée des contraintes intégrées directement dans la fonction objective du modèle proposé ; et (2) une deuxième vision, qui est locale et basée sur le contrôle stricte de violation de ces dites contraintes. Les deux approches évaluent la pertinence des variables par des poids appris en cours de la construction du modèle de classification.
En outre de cette tâche principale de sélection de variables, nous nous intéressons au traitement de la redondance. Pour traiter ce problème, nous proposons une méthode originale combinant l'information mutuelle et un algorithme de recherche d'arbre couvrant construit à partir de variables pertinentes en vue de l'optimisation de leur nombre au final.
Finalement, toutes les approches développées dans le cadre de cette thèse sont étudiées en termes de leur complexité algorithmique d'une part et sont validés sur des données de très grande dimension face et des méthodes connues dans la littérature d'autre part.

Mots clés : Sélection de variables, données semi-supervisées, contraintes, redondance, réduction de dimension.

Abstract

Feature selection is an important task in data mining and machine learning processes. This task is well known in both supervised and unsupervised contexts. The semi-supervised feature selection is still under development and far from being mature. In general, machine learning has been well developed in order to deal with partially-labeled data. Thus, feature selection has obtained special importance in the semi-supervised context. It became more adapted with the real world applications where labeling process is costly to obtain.

In this thesis, we present a literature review on semi-supervised feature selection, with regard to supervised and unsupervised contexts. The goal is to show the importance of compromising between the structure from unlabeled part of data, and the background information from their labeled part. In particular, we are interested in the so-called «small labeled-sample problem» where the difference between both data parts is very important.

In order to deal with the problem of semi-supervised feature selection, we propose two groups of approaches. The first group is of «Filter» type, in which, we propose some algorithms which evaluate the relevance of features by a scoring function. In our case, this function is based on spectral-graph theory and the integration of pairwise constraints which can be extracted from the data in hand. The second group of methods is of «Embedded» type, where feature selection becomes an internal function integrated in the learning process. In order to realize embedded feature selection, we propose algorithms based on feature weighting. The proposed methods rely on constrained clustering. In this sense, we propose two visions, (1) a global vision, based on relaxed satisfaction of pairwise constraints. This is done by integrating the constraints in the objective function of the proposed clustering model; and (2) a second vision, which is local and based on strict control of constraint violation. Both

approaches evaluate the relevance of features by weights which are learned during the construction of the clustering model.
In addition to the main task which is feature selection, we are interested in redundancy elimination. In order to tackle this problem, we propose a novel algorithm based on combining the mutual information with maximum spanning tree-based algorithm. We construct this tree from the relevant features in order to optimize the number of these selected features at the end.
Finally, all proposed methods in this thesis are analyzed and their complexities are studied. Furthermore, they are validated on high-dimensional data versus other representative methods in the literature.

Keywords: Feature selection, semi-supervised data, pairwise constraints, redundancy, dimensionality reduction.

ملخص

إن اختيار المواصفات هي عملية أساسية في تطبيقات التعلم الآلي والتنقيب في البيانات. كما أن إشكالية هذا العلم معروفة بشكل جيد في مجالي التعليم الآلي الرئيسيين : التعليم بإشراف و بدون إشراف. بينما يبقى مجال التعليم بشبه إشراف مازال في مراحله البدائية ومازالت الأعمال فيه متواضعة. كلمات مفتاحية : اختيار المواصفات، بيانات نصف معلمة، قيود ثنائية، تكرار المواصفات، تخفيض الأبعاد.

Contents

List of Figures

List of Tables

List of Algorithms

1 General Introduction

1.1 Context and Motivations

In the various fields of engineering and nowadays applications, data acquisition tools have extensively proliferated, and decisional information systems are then requiring more complex analysis of large amount of data (signals, images, documents, etc.). However, while this accumulation of data is sure to have useful information, the abundance of such data poses problems of structuring and knowledge extraction. Indeed, databases are usually defined by two-dimensional arrays corresponding to data instances and attributes characterizing these data. The instances and/or attributes can be of very high dimensionality, which can be a problem during storage, exploration and analysis of such data in several application domains. In addition, it is important to develop specific tools for data processing that are efficient in extracting the underlying knowledge. Knowledge extraction is carried out according to two directions, 1) the categorization of data (Cluster analysis), and/or 2) dimensionality reduction of the representation space which can help in improving the performance of learning algorithms. Moreover, while clustering aims to discover the intrinsic structure of a dataset by forming groups that share similar characteristics, dimensionality reduction is considered as a crucial step in the process of data pre-processing (filtering, cleaning, removal of outliers, etc..). Indeed, for data belonging to a high-dimensional space, some attributes do not provide any information or express noise and others might be redundant

or irrelevant. In general such useless dimensionality makes the algorithms complex, inefficient, less general and difficult to interpret. The methods of dimensionality reduction may be roughly divided into feature extraction and feature selection approaches. Feature Extraction methods transform the problem into a lower dimensional space by proposing new features extracted from the original ones, while feature selection measures the relevance of individual features (or subsets of features). Feature selection depends largely on explicit and/or implicit background knowledge about data.

With the plenty of acquired data, the labeling procedure performed by a human expert can be tedious, costly in time and labor. This is why, for many real world applications, it is usual that databases are composed of large amount of unlabeled data, and few number of labeled instances. This learning context is called "semi-supervised" because the analyst is supposed to use both labeled and unlabeled data in the learning process.

The general problem of feature selection is well addressed in the literature by data mining and machine learning communities. The goal of this task is to remove both irrelevant and redundant features in order to decrease the complexity and improve the interpretability and the performance of learning algorithms [Guan et al., 2011]. Feature selection is well studied in both supervised and unsupervised contexts in several works [Guyon and Elisseeff, 2003, Dy and Brodley., 2004]. In the context of supervised feature selection, the relevance of a feature is evaluated by its correlation with the class label. The unsupervised feature selection is considered as a much harder problem, because of the absence of class labels that could guide the search for relevant information.

Recently, learning from both labeled and unlabeled data has been gaining a considerable interest. Thus, the semi-supervised feature selection became more important and more adapted to real-world applications whereas labeled data are hardly and costly obtained. In addition, the task is more challenging with the so called “small labeled-sample” problem in which the amount of data that is unlabeled can be much larger than the amount of labeled data [Zhao

and Liu, 2007a]. In order to deal with the aforementioned specification of semi-supervised data, novel approaches were proposed instead of using the methods from the neighboring paradigms (supervised and unsupervised). On the one hand, supervised feature selection algorithms require a large amount of labeled training data. As a result, such algorithms provide insufficient information about the structure of the target concept, and thus could fail to identify the relevant features that are discriminative to different classes. On the other hand, unsupervised feature selection algorithms ignore label information, thus may lead to performance deterioration. Therefore, semi-supervised feature selection has now special interest as being a relatively recent domain, where few of works exist in the literature.

Semi-supervised feature selection algorithms can be categorized as filter, wrapper and embedded methods. Filter model techniques examine intrinsic properties of the data to evaluate the features prior to the learning task, while Wrapper approaches evaluate the features using the learning algorithm that will ultimately be employed. They "wrap" the selection process around the learning algorithm. Finally, embedded methods are locally specific to a model during its construction.They aim to learn the feature relevance with the associated learning algorithm. In other terms, they incorporate feature selection and learning algorithm in the same objective function.

1.2 Contributions

The main motivation of this thesis is the semi-supervised feature selection from high-dimensional data, we try to deal with this problem from different viewpoints. Feature selection is known to be the process by which the irrelevant and redundant features are identified. Therefore, we first tackle the problem of relevant feature selection, and then the redundancy elimination.

In order to identify relevant features, we firstly propose a specific semi-supervised feature selection score that we call, Constrained Laplacian Score (**CLS**). In this score, we assess the exploiting of the two parts of semi-supervised dataset, i.e. labeled and unlabeled parts, with efficient and low computational-

complexity cost function. CLS uses information in the labeled part of data after transforming it into pairwise constraints. The reason lying behind using these constraints is their efficiency in improving the learning performance, and because they are more general than class labels. In fact, these constraints can be generated from class labels but not the opposite. In addition, these constraints are easier to be identified *a priori* than the class labels (e.g. similarity may generate a constraint but not labels). The use of constraints in our score raises some challenges. First of all, constraints are rather few in semi-supervised data, this makes their quality a critical issue. In addition, it is practically proven that constraints might have noise which can deteriorate performance and mislead the learning process. Therefore, the paucity of constraints and the probable noise in them were the main problem which we tried to handle in new approaches.

In order to cope with these problems we propose the employment of a constraint selection process based on a utility measure. In this sense, we propose a Constraint Selection-based Feature Selection (**CSFS**) framework, by which we improve the feature relevance function in order to weigh certain situations where there are some conflicts between the data structure and the labels (e.g. if two data points are relatively near to each other but have different labels).

Furthermore, in order to treat the redundancy in the selected relevant features, we propose a graph-based approach in order to eliminate the redundant features. The extended method, called Constraint Selection-based Feature Selection with Redundancy Elimination (**CSFSR**), has proven -as expected- to improve the quality of features (hence the underlying learning process) after redundancy elimination.

In the other part, we propose two embedded approaches for feature selection that we integrate with the well-known clustering algorithm (k-means). The first approach, called Weighted Constrained k-means (**wCKM**), uses a fuzzy version of k-means with a soft integration of pairwise constraints. This integration is done by modifying the objective function in order to calculate the penalty of constraint violation. In the second approach, called Local-to-Global Feature

selection (**L2GFS**), we present a hard version of k-means with a strict control of constraint violation.

The common point between wCKM and L2GFS is that both methods are based on a weighted metric model. Moreover, both approaches proceed by feature weighting for semi-supervised feature selection based on constrained k-means. However, an essential difference between them is that the former is a direct global approach which selects relevant features over all clusters, while the latter is a local to global approach, which does first, a local feature weighting in order to choose the cluster-relevant features, then it produces a global selection by local weight aggregation.

The results of all approaches are promising and very competitive to several representative methods of feature selection from high-dimensional data.

1.3 Organization of the report

In the remainder of this thesis, we will describe several approaches of dimensionality reduction, especially the semi-supervised feature selection algorithms available in the literature. Then, we will present our proposals with both filter and embedded paradigms, as well as our algorithm for redundancy elimination. This is achieved through the course of the remaining chapters, which are structured as follows:

- In chapter 2, we will describe a variety of representative dimensionality reduction approaches. This includes feature extraction and feature selection techniques in both supervised and unsupervised domains. In addition, we will focus on approaches that are currently available in the literature of semi-supervised feature selection, and we will discuss their limitations. Especially, we will highlight the limitations related to the nature of semi-supervised domain that we placed earlier in this chapter.

- In chapter 3, we will start by presenting our filter approaches for semi-supervised feature selection, where the feature selection in this case is

considered as an independent step of the learning process. We will show different ways to deal with domain requirements such as paucity of labels and inutility in constraints. We will also discuss an original graph-based approach for redundancy elimination, which can be viewed as the other part of feature selection.

- In chapter 4, we will present our embedded approaches for semi-supervised feature selection, which are achieved by integrating feature selection in the k-means algorithm. We will propose two variants which take into account the pairwise constraints generated from labels.

  In chapter 3 and 4, we will present an extensive empirical study for all the proposed methods. The experiments are done on high dimensional benchmarking datasets downloaded from well-known repositories. We will present also a variety of strategies and scenarios during the comparisons, and in different contexts.

- Finally, chapter 5 will conclude this thesis, focusing on the contributions and limitations of the algorithms that we have developed, and will outline future works that can be carried out to extend and enhance the proposed ideas.

2 Semi-Supervised Feature Selection for Dimensionality Reduction

2.1 Introduction

Dimensionality reduction is a significant task when dealing with high-dimensional data. It can be applied to reduce the dimensionality of the original data and improve learning performance. By removing the irrelevant and redundant features, or by effectively combining original features to generate a smaller set of them with more discriminant power, dimensionality reduction techniques bring the immediate effects of speeding up data mining algorithms, improving performance, and enhancing model comprehensibility [Zhao and Liu, 2012]. Dimensionality reduction can be performed by two categories of techniques: Feature extraction or Feature selection.

2.2 Feature Extraction

Feature extraction reduces dimensionality by generating a small set of new features via combining the original ones (Figure 2.1). According to the label information availability, feature extraction methods can be categorized into supervised and unsupervised approaches. Fisher Linear Discriminant (FLD) [Fisher, 1936] is an example of supervised feature extraction, which can extract the optimal discriminant vectors when class labels are available. It is a classification method which projects high-dimensional data onto a line and performs

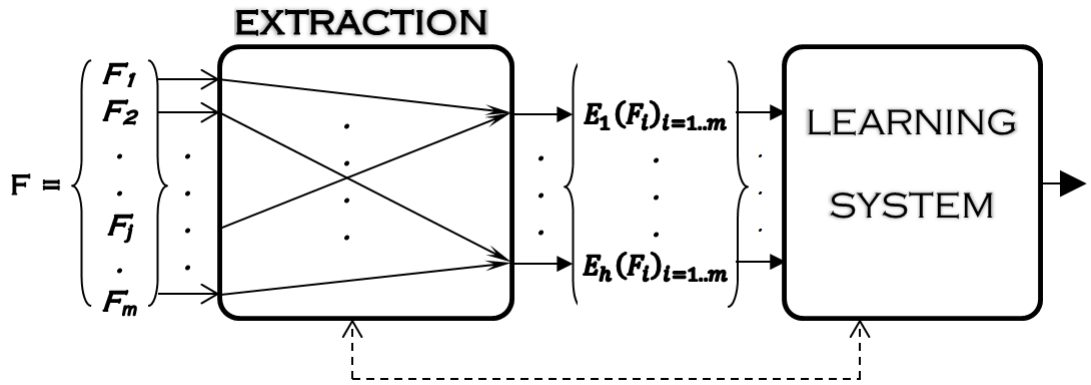


Figure 2.1: General framework of feature extraction

classification in this one-dimensional space. It finds the linear discriminant function between the given classes by minimizing the errors in the least square sense. In [Bar-Hillel et al., 2005], the authors proposed a semi-supervised version of FLD, called (cFLD). The idea behind (cFLD) is the integration of one type of pairwise constraints (positive constraints) in (FLD) for the objective of dimensionality reduction. cFLD was proposed as an interim-step for Relevant Component Analysis (RCA). However, cFLD has the singular problem when constraints are limited.

For unsupervised feature extraction methods, the popular Principal Component Analysis (PCA) [Jolliffe, 2002] tries to preserve the global covariance structure of data when class labels are not available. It is categorized as an eigenvector method designed to model linear variability in high dimensional data. In PCA, the linear projections of greatest variance are computed from the top eigenvectors of the data covariance matrix.

Other methods can be found in the literature dealing with feature extraction, for example (LLE: Locally Linear Embedding) [Roweis and Saul, 2000] is an unsupervised learning algorithm which computes low-dimensional, neighborhood-preserving embeddings of high-dimensional inputs. LLE proposes to learn the global structure of nonlinear manifolds, such as those generated by face images or text documents. Another feature extraction method is (k-PCA: Kernal PCA) [Schölkopf et al., 1998] that generalizes PCA to the case where principal

components in the input space are not the main interest, but the principal components of variables, or features, which are non-linearly related to the input space. The authors in [He and Niyogi, 2004] propose (LPP: Locality preserving Projection) which is a graph-based feature extraction method. It builds a graph incorporating neighborhood information of the dataset. Using the notion of the Laplacian of the graph, it then computes a transformation matrix which maps the data points to a subspace. This linear transformation is attended to preserve local neighborhood information in a certain sense. Furthermore, the authors in [Belkin and Niyogi, 2002] present a geometrically motivated feature extraction algorithm (LE: Laplacian Eigenmap) which has a few local computations and one sparse eigenvalue problem. The method reflects the intrinsic geometric structure of the manifold using the Laplacian operator in providing an optimal embedding.

2.3 Feature Selection

Feature selection attains dimensionality reduction by selecting a small set of the original features (Figure 2.2). To realize this goal, a feature evaluation criterion

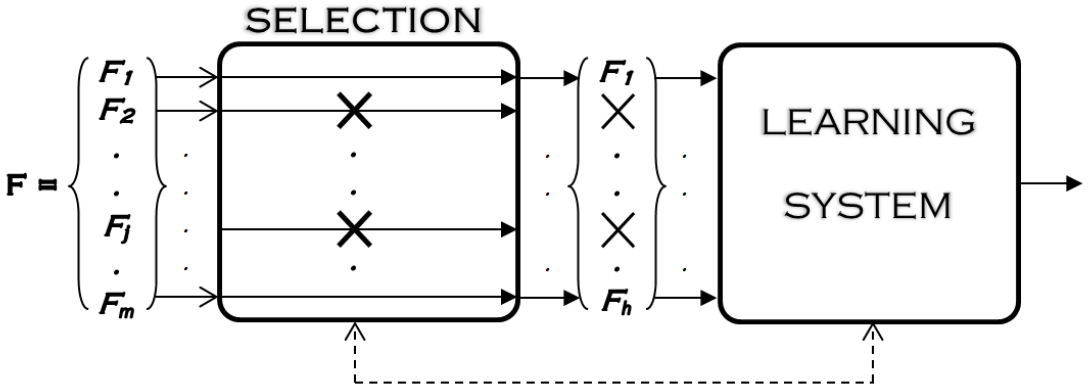


Figure 2.2: General framework of feature selection

is used with a search strategy to identify the relevant features. Actually, feature selection has become an essential task for high-dimensional data analysis in machine learning and data mining tasks. It is one of the effective means to identify relevant features for dimensionality reduction [Jain and Zongker,

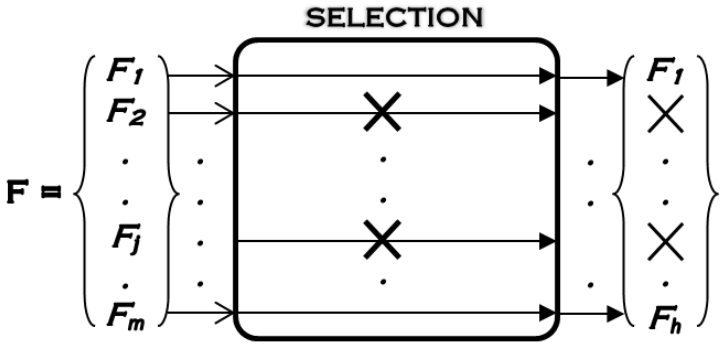


Figure 2.3: Filter feature selection

1997]. This task has led to improved performance for many benchmarking datasets [Frank and Asuncion, 2010, Zhao et al., 2011] as well as for real-world applications over data such as digital images, financial time series and gene expression microarrays [Guyon and Elisseeff, 2003]. Generally, feature selection methods can be classified in three types: filter, wrapper or embedded.

The filter model techniques examine intrinsic properties of the data to evaluate the features prior to the learning tasks [Yu and Liu, 2003] (Figure 2.3). In fact, the independence from learning system makes the filter methods applicable to a large variety of learning algorithm, and more robust against learning overfitting. Moreover, filter approaches have lower computational complexity than the other approaches.

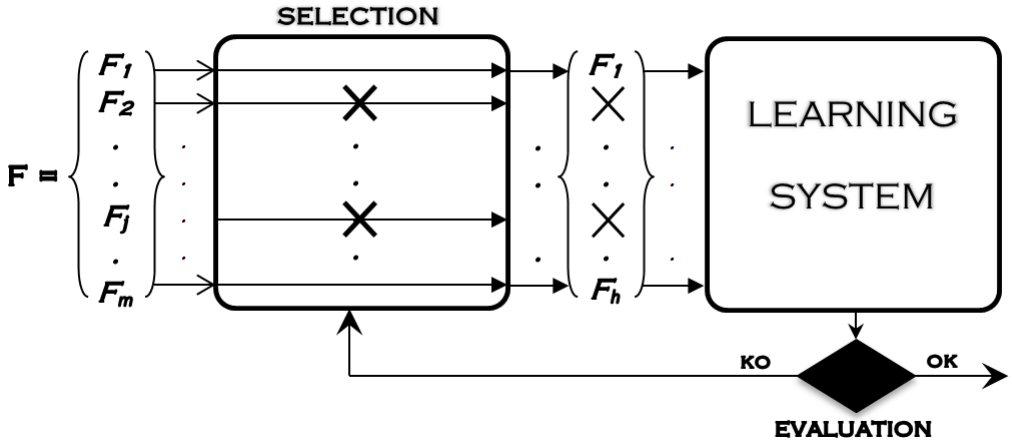


Figure 2.4: Wrapper feature selection

The wrapper based approaches evaluate the features using the learning algorithm that will ultimately be employed [Kohavi and John, 1997]. Thus, they "wrap" the selection process around the learning algorithm (Figure 2.4).

In fact, wrapper methods select the most relevant features using an induction algorithm. However, wrapper approaches are very prone to overfitting and suffer from the high computational complexity.

The embedded methods are locally specific to models during their construction. They aim to assess the feature usefulness with the associated learning algorithm [Roweis and Saul, 2000] (Figure 2.5). In general, embedded feature selection methods are better than wrapper methods when the goal is the relevance of features towards certain algorithm, this is because embedded methods are less computationally expensive and less prone to overfitting than wrapper methods [Saeys et al., 2007].

Feature selection is a well addressed in supervised and unsupervised domains with several works [Guyon and Elisseeff, 2003, Dy and Brodley., 2004]. In the supervised context, the relevance of a feature can be evaluated by its correlation with the class label, Fisher score [Duda et al., 2000], for example, is a supervised method which seeks features with best discriminant ability, it tries to find a subset of features, such that in the data space spanned by the selected

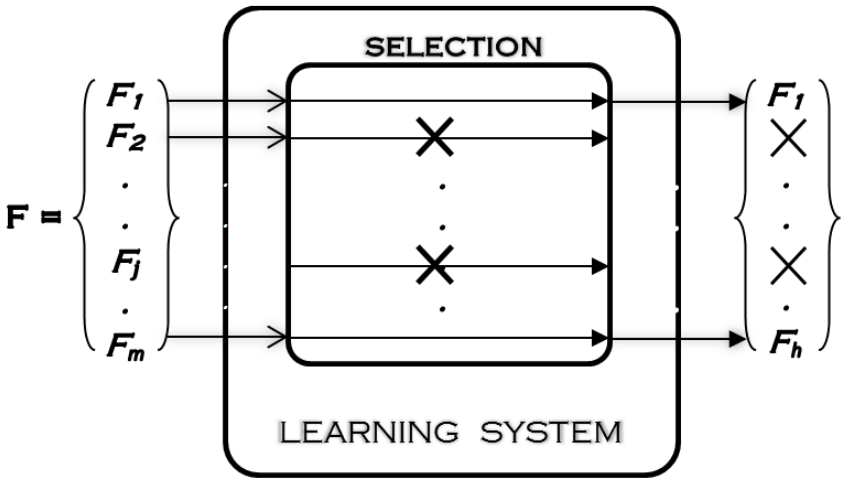


Figure 2.5: Embedded feature selection

features, the distances between data points in different classes are as large as possible, while the distances between data points in the same class are as small as possible [Gu et al., 2011]. In [Robnik-Šikonja and Kononenko, 2003], the authors presented a theoretical and empirical analysis of Relief feature selection algorithms (Relief [Kira and Rendell, 1992], ReliefF [Kononenko, 1994] and RReliefF [Kononenko et al., 1997]). A key idea of these methods is to estimate the quality of features according to how well their values distinguish between instances that are near to each other. The original Relief algorithm was limited to classification problems with two classes, it was extended by ReliefF in order to deal with multiclass problems. ReliefF algorithm is more robust and also able to deal with incomplete and noisy data. Finally, RReliefF was proposed to extend the former algorithm in order to be adapted for regression problems. The authors in [Yu and Liu, 2004] proposed a Fast Correlation-Based Filter method (FCBF) as a novel concept of predominant correlation and analyzing feature redundancy. According to FCBF, a feature is "good" if it is predominant[1] in predicting the class concept. The authors proposed three heuristics that together can identify predominant features and remove redundant ones among them.

The unsupervised feature selection is considered as a much harder problem, due to the absence of class labels that would guide the search of relevant information. For example, Variance score [Bishop, 1995] computes the variance along each feature in order to reflect its representative power. In [Dy and Brodley., 2004], the authors introduced a wrapper framework for performing feature subset selection for unsupervised learning. The method, called (FSSEM) for "Feature Subset Selection and EM[2] Clustering", searches through feature subset space, and exploits EM clustering algorithm [Dempster et al., 1977] on each candidate subset. Then, it evaluates the resulting clusters and feature subset using "scatter separability" or "maximum likelihood" criteria. The whole procedure is repeated until finding the best feature subset with its corresponding clusters based on a given feature evaluation criterion. Another feature selec-

[1] A feature is predominant if it does not have any approximate Markov blanket in its feature set. More details can be found in [Yu and Liu, 2004].

[2] Expectation Maximization.

tion approach is SPEC [Zhao and Liu, 2007b] which was proposed as a general framework of spectral feature selection for both supervised and unsupervised learning. In SPEC framework, the relevance of a feature is determined by its consistency with the structure of the graph induced from the corresponding similarity matrix S. This matrix can be constructed according to the geometric structure of the data (unsupervised case) or the class affiliation (supervised case). Its goal is to represent the relationships between instances. The SPEC authors showed that ReliefF [Kononenko, 1994] and Laplacian score [He et al., 2005] (which will be detailed later in section (2.7.1)) can be derived as special cases from the SPEC framework. They also showed that novel spectral feature selection algorithms can be derived from SPEC conveniently.

2.4 Redundancy Analysis

In feature selection, it has been recognized that the combinations of individually good features do not necessarily lead to good learning performance. In other words, the h best features are not the best h ones. Indeed, redundant features increase dimensionality unnecessarily and worsen learning performance when facing shortage of data [Zhao et al., 2010]. Some researchers have studied indirect or direct means to reduce the redundancy among features. For example, the authors in [Ding and Peng, 2003, Peng et al., 2005] introduced a method for reducing redundancy in feature selection based on pairwise feature correlation which is measured by mutual information. Their method, called (mRMR) for "minimum redundancy – maximum relevance", selects the features such that they are maximally dissimilar regarding their mutuality. Then, it selects the subset which best characterizes the statistical property of a target classification variable. mRMR tries to ensure that the selected features are mutually as dissimilar to each other as possible, but marginally as similar to the classification variable as possible. The authors in [Weston et al., 2003] proposed (AROM-SVM) stands for "Approximation of the zero-norm Minimization". The method relies on an embedded model, which removes redundant features by iteratively reducing the weights of features which are less important for a Support Vector Machine (SVM) classifier [Vapnik, 1995]. In addition, (FCBF)

which we summarized in the previous section performs a redundancy elimination. In fact, it approximates relevance and redundancy analysis by selecting all predominant features and removing the rest ones. Then, it uses both C- and F-correlations (stand for Feature/Class and Feature/Feature correlations respectively) to assess the feature redundancy. Recently, the authors in [Zhao et al., 2012] introduced a framework for Similarity Preserving Feature Selection, named SPFS. The goal of this method is to select a subset of features, upon which, the pairwised sample similarity specified by a predefined similarity matrix is best preserved. The similarity matrix can be constructed either by using the label information in supervised learning or using certain distance metrics in unsupervised learning. By preserving the sample similarity specified in the similarity matrix, SPFS is able to select a subset of features that can maintain or even improve the performance of learning models. In addition, SPFS improves the similarity preservation by handling feature redundancy during feature selection.

2.5 Semi-Supervised Feature Selection

As we pointed out earlier, feature selection can be done in three frameworks according to class label information. The most addressed framework is the supervised one, and the unsupervised feature selection is considered as a much harder problem, due to the absence of labels. The problem becomes more challenging when data contain labeled and unlabeled examples. It is more adapted with real-world applications where labeled data are costly to obtain. In this context, the effectiveness of semi-supervised learning has been demonstrated [Chapelle et al., 2006]. In general, feature selection depends on data structure (unsupervised), or information carried in labels (supervised). Then the semi-supervised feature selection is expected to make profit from both parts. Specifically, the labeled part presents important information about the target concept. In addition, the unlabeled part reflects the data structure which is probable to harmonize with label information (labeled instances which belong to the same class are expected to be close to each other).

In the following, we investigate into the literature of semi-supervised feature selection. We start with some definitions, and then we list the key methods of this domain.

2.6 Definitions and notations

Definition 1. *(semi-supervised Data)*

In semi-supervised learning, a dataset of n data points $X = \{x_i\}_{i=1..n}$ consists of two subsets depending on the label availability: $X_L = \{x_1, x_2, \ldots, x_l\}_{l\neq 0}$ for which the labels $Y_L = \{y_1, y_2, \ldots, y_l\}$ are provided, and $X_U = \{x_{l+1}, x_{l+2}, \ldots, x_{l+u}\}_{u\neq 0}$ which are unlabeled. A data point (also called instance, example or observation) x_i is a vector with m dimensions (also called features, variables or attributes), while a label $y_i \in \{1, 2, \ \ldots, C\}$ (C is the number of different labels), and $l + u = n$ (n is the total number of instances). When $l = 0$, the whole data points X are unlabeled and we are in the context of unsupervised learning. When $u = 0$, the whole data points X are labeled and we are in the context of supervised learning. In general, $l << u$ in the case of semi-supervised learning, which defines the "small labeled-sample" problem.

Definition 2. *(Pairwise Constraints)*

Pairwise constraints provide guidance about the desired partition and make it possible for many unsupervised learning algorithms to increase their performance [Davidson et al., 2006]. A pairwise constraint concerns two data points and can be of following two types:

- **Must-Link constraint** (***ML***)(called also positive constraint): involving x_i and x_j , specifies that they belong to the same class.
- **Cannot-Link constraint** (***CL***)(called also negative constraint): involving x_i and x_j , specifies that they belong to different classes.

ML and CL constraints are then grouped in two defined subsets Ω_{ML} and Ω_{CL}

respectively. These constraints can be expanded, while taking into account the transitive closure:

- $(x_i, x_j) \in \Omega_{ML} \wedge (x_j, x_k) \in \Omega_{ML} \Longrightarrow (x_i, x_k) \in \Omega_{ML}$.
- $(x_i, x_j) \in \Omega_{ML} \wedge (x_j, x_k) \in \Omega_{CL} \Longrightarrow (x_i, x_k) \in \Omega_{CL}$.

In semi-supervised learning, constraints represent background knowledge, and add a better description of the target concept. They can be added directly to data instances, this is particularly interesting in certain real-world tasks, e.g. image retrieval [Bar-Hillel et al., 2005], because in such cases, the true labels may be unknown *a priori*, while it can be easier for a user to specify whether some pairs of examples belong to the same class or not, i.e. similar or dissimilar. In addition, they can be automatically generated from the labeled part of data as follows: For any pair of observations (x_i, x_j) in X_L there is a constraint of type ML if both observations have the same label, and the constraint type is CL otherwise. Note that in the case of automatic constraint generation, there is no need for transitive closure, since all possible constraints between data points are already generated.
Note that pairwise constraints are not the only type of constraints that may exist over data. There exist other types, like ϵ-constraints, δ-constraints [Davidson and Ravi, 2005], probabilistic constraints [Law et al., 2005], and complex constraints [Law et al., 2004].

Definition 3. *(semi-supervised Feature Selection)*

Let $F_1, F_2, \ldots, F_m$ denote the m features of X and $f_1, f_2, \ldots, f_m$ be the corresponding feature vectors that record the feature value on each instance. semi-supervised feature selection is the use of both X_L and X_U to identify the set of most relevant features $F_{j_1}, F_{j_2}, \ldots, F_{j_h}$ of the target concept, where $h \leq m$ and $j_r \in \{1, 2, \ldots, m\}$ for $r \in \{1, 2, \ldots, h\}$.

The methods that we illustrate in this section are based in large part on spectral graph theory [Chung, 1997]. In the following, we present some definitions of basic concepts from this framework.

The spectral graph theory represents a solid theoretical framework which has been the basis of many powerful existing feature selection methods such as ReliefF [Robnik-Šikonja and Kononenko, 2003], Laplacian Score [He et al., 2005] , sSelect [Zhao and Liu, 2007a], SPEC [Zhao and Liu, 2007b] and Constraint score [Zhang et al., 2008]. All of these methods used the application of graph eigenvalues in the objective of feature selection.

Definition 4. *(Weighted Graph of Data)*

Given a dataset X, let $G(V,E)$ be the complete undirected graph constructed from X, with V is its node set and E is its edge set. The i^{th} node v_i of G corresponds to $x_i \in X$. We associate with the graph G a weight function $w : V \times V \to \mathbb{R}$ satisfying the following constraints

$$\begin{cases} w(v_i,v_j) = w(v_j,v_i) \\ w(v_i,v_j) \geq 0 \end{cases} \tag{2.1}$$

Note that if $\{v_i,v_j\} \notin E(G)$, then $w(v_i,v_j) = 0$. Unweighted graphs are just the special case where all the weights are 0 or 1.

Definition 5. *(Graph of Dissimilarity)*

Given a dataset X, let $G(V,E)$ be its weighted graph of data constructed from X, where each edge's weight is expressed by the Euclidean distance-based Gaussian function $w_{ij} = e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}}$, which represents the dissimilarity between data points x_i and x_j (where λ is a constant to be set, and $\|x_i - x_j\|^2$ denotes the Euclidean distance between x_i and x_j). Then, G is said to be the graph of dissimilarity for data X.

Definition 6. *(Dissimilarity Matrix)*

Given a dataset X, let $G(V,E)$ be its dissimilarity graph with n nodes, a dissimilarity matrix $\boldsymbol{S}$ is an $n \times n$ matrix where

$$\boldsymbol{S}_{ij} = w_{ij} \text{ the dissimilarity between } x_i \text{ and } x_j \tag{2.2}$$

Definition 7. *(Degree Matrix)*

Given a dataset X, and $\boldsymbol{S}$ be its dissimilarity matrix of dimension $n \times n$, the degree matrix $\boldsymbol{D}$ is a diagonal $(n \times n)$ matrix defined by

$$\begin{aligned} \boldsymbol{D}_{ii} &= \sum_{j=1}^{n} \boldsymbol{S}_{ij} \\ &= diag(\boldsymbol{S}\mathbf{1}) \ , \ \mathbf{1} = (1, \ldots, 1)^T \end{aligned} \tag{2.3}$$

Note that $\boldsymbol{D}_{ii}$ represents the density of the node x_i.

Definition 8. *(Laplacian Matrix)*

Given a dataset X with $\boldsymbol{S}$ and $\boldsymbol{D}$ be its dissimilarity and degree matrices respectively. The Laplacian matrix $\boldsymbol{L}$ of X is defined by

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{S} \tag{2.4}$$

In fact, the definitions which we listed above are required for presenting the literature methods later.

2.7 Filter-based approaches

A feature selection approach is called "Filter" if it is independent of the learning algorithm. In general, a filter approach may be viewed as a prior learning step, it removes the irrelevant features which may deteriorate the performance of the later learning process. Thus, the whole feature selection is performed prior to the execution of the learning algorithm. Moreover, the independence of feature selection process from the learning algorithm gives the liberty of choosing different models later to apply. Filter approaches select features according to some structural properties in case of unsupervised learning, and according to correlation with labeling information in case of supervised one. In the case of semi-supervised feature selection, the filter approaches try to make use of both labeled and unlabeled data. In the following sections we will illustrate

in details various known filter score-based approaches which try to solve the "small labeled-sample" problem.

2.7.1 Laplacian Score (LS)

Laplacian Score (LS) [He et al., 2005] belongs to spectral feature selection family. This score was originally used for unsupervised feature selection with the ability to deploy class labels in case of their availability. LS makes a further step over variance score [Bishop, 1995], which uses the variance along certain dimension to reflect its representative power, then the features with the maximum variance are selected. However, LS does not only favor those features with larger variances, which have more representative power, but also tends to select the features with stronger locality preserving ability. This method is also generalized by the SPEC method [Zhao and Liu, 2007b] in the unsupervised context. A key assumption in LS is that instances from the same class are supposed to be close to each other.

Let LS_r denotes the Laplacian Score of the r^{th} feature F_r. Let f_{ri} denotes the i^{th} sample of this feature, where $i = 1, \ldots, n$. The algorithm of Laplacian score can be stated as follows:

1. Given $G(V, E)$ the dissimilarity graph of data X, construct $G_{kn}(V, E_{kn})$ which is a k-nearest neighborhood subgraph from G as follows :

   - The nodes V in G_{kn} remains the same as in G (as they represent data points)
   - E_{kn} in the graph G_{kn} form a subset of the edges set E in the graph G. The choice of an edge subset from E to be kept in E_{kn}, is based on k-nearest neighborhood. This means that an edge $\{e_{i,j}\}$ is kept in E_{kn} if x_i is one of the k-nearest neighbors of x_j (and vice-versa), or if x_i and x_j share the same class labels (when they are available), thus LS can take into consideration the case where labels are given.

2. The Dissimilarity matrix $\boldsymbol{S}$ is then defined as :

$$\boldsymbol{S}_{ij} = \begin{cases} e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}} & \text{if there is an edge between } x_i \text{ and } x_j \text{ i.e. } x_i \text{ and } x_j \\ & \text{are neighbors or } (x_i, x_j) \in \Omega_{ML} \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

3. Then, the following definitions are given:

   - For each feature F_r, its vector $f_r = (f_{r1}, ..., f_{rn})^T$
   - The diagonal matrix $\boldsymbol{D}$ according to eq.(2.3)
   - The Laplacian matrix $\boldsymbol{L}$ according to eq.(2.4)

4. Laplacian Score of the r^{th} feature is then computed as follows :

$$LS_r = \frac{\tilde{f}_r^T \boldsymbol{L} \tilde{f}_r}{\tilde{f}_r^T \boldsymbol{D} \tilde{f}_r} \quad \text{where} \quad \tilde{f}_r = f_r - \frac{f_r^T \boldsymbol{D}\mathbf{1}}{\mathbf{1}^T \boldsymbol{D}\mathbf{1}}\mathbf{1} \tag{2.6}$$

The Authors in [He et al., 2005] proved That the above score is equivalent to the minimization of the objective function:

$$LS_r = \frac{\sum_{i,j}(f_{ri} - f_{rj})^2 \boldsymbol{S}_{ij}}{\sum_i (f_{ri} - \mu_r)^2 \boldsymbol{D}_{ii}} \tag{2.7}$$

where λ is a constant to be set and $\mu_r = \frac{1}{n}\sum_i f_{ri}$ is the mean of feature vector F_r. In addition, they provided a theoretical analysis of the connection between LS algorithm and the canonical Fisher score [Duda et al., 2000]. The algorithm of LS is detailed in Algorithm 1.

LS presents interesting results in the case of unsupervised learning, this is because it investigates the variance of data in order to assess the locality preserving ability of features. Then, a "good" feature for this score is the one where two neighboring examples record close values. In addition, in semi-supervised context, this score can process the labeled part of data which carry important background information. Such information is provided to guide the learning process and proved to have considerable effect on learning process. However,

Algorithm 1 LS

Input: Dataset X, pairwise constraints set Ω_{ML}, degree of neighborhood k and the constant λ
Output: the ranked features list
1: Build G the dissimilarity graph of data X
2: Calculate the dissimilarity matrix S, the diagonal matrix $\boldsymbol{D}$ and the Laplacian matrix $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{S}$
for $j = 1$ **to** m **do**
  3: Calculate LS_r, the score of F_j using eq.(2.6)
end for
4: Rank the features according to their scores in ascending order.

LS does not profit from the background information (the CL constraints in particular), which are provided to guide the learning process. In addition, the (k)-neighborhood parameter has significant effects on the results as it was discussed by the authors, and its choice is not clearly defined.

2.7.2 Spectral Graph-based Semi-Supervised Feature Selection score (sSelect)

This method [Zhao and Liu, 2007a] introduced the first semi-supervised feature selection algorithm based on spectral analysis. The algorithm exploits both labeled and unlabeled data through a regularization framework, which provides an effective way to address the "small labeled-sample" problem.

The idea of sSelect method is to transform a feature vector f_r into a cluster indicator g_r, so each element f_{ri} where $(i = 1, 2, \ldots, n)$ of f_r indicates the affiliation of the corresponding instance x_i . In order to calculate the cluster indicator, the authors defined a "$F - C$ transformation" as follows:
Let $f_r \in \mathbb{R}^n$ and $\mathbf{1} = (1, \ldots, 1)^T$, the $F - C$ transformation θ is defined as:

$$g_r = \theta(f_r) = f_r - \frac{f_r^T \boldsymbol{D}\mathbf{1}}{\mathbf{1}^T \boldsymbol{D}\mathbf{1}}.\mathbf{1}; \tag{2.8}$$

where $\boldsymbol{D}$ is the degree matrix of data. The fitness of a cluster indicator g_r is then evaluated by two factors: (1) separability - whether the cluster structures

formed are well separable; and (2) consistency - whether they are consistent with the given label information.

The $F-C$ transformation proceeds as follows: Given a cluster indicator $\boldsymbol{g}_r$, labeled data X_L and unlabeled data X_U, the fitness should be evaluated by: 1) whether the clusters formed by the indicator are well separable (renders a small cut value), and 2) whether it is consistent with the label information. To do so, the authors designed a regularization framework, which evaluates the fitness of the cluster indicator using both labeled and unlabeled data. They defined it as follows:

Let $\boldsymbol{g}_r$ be the cluster indicator generated from a feature vector f_r and $\hat{\boldsymbol{g}}_r = sign(\boldsymbol{g}_r)$, the regularization framework is defined as:

$$sSelect_r = \eta \frac{\boldsymbol{g}_r^T \boldsymbol{L} \boldsymbol{g}_r}{\boldsymbol{g}_r^T \boldsymbol{D} \boldsymbol{g}_r} + (1-\eta)(1 - NMI(\hat{\boldsymbol{g}}, Y_L)) \tag{2.9}$$

where Y_L are the available labels, $\boldsymbol{L}$ is the Laplacian matrix, $\boldsymbol{D}$ is the diagonal matrix, η is a constant to be set, and $NMI(\hat{\boldsymbol{g}}, Y_L)$ is the normalized mutual information between $\hat{\boldsymbol{g}}$ and Y_L [Press, 2007], which is used to measures the consistency between the discretized cluster indicators and the labeled data, and is defined as:

$$NMI(\hat{\boldsymbol{g}}, Y_L) = \frac{I(\hat{\boldsymbol{g}}, Y_L)}{\sqrt{H(\hat{\boldsymbol{g}}) H(Y_L)}} \tag{2.10}$$

where $I(\cdot)$ is the mutual information metric, and $H(\cdot)$ is the entropy metric.

Algorithm 2 sSelect

Input: Dataset X, η, k
Output: the ranked features list
1: Construct the k-nearest neighbors graph G from X
2: Build the dissimilarity matrix $\boldsymbol{S}$, the degree matrix $\boldsymbol{D}$ and the Laplacian matrix $\boldsymbol{L}$ from G
for $r = 1$ **to** m **do**
  3: Construct the cluster indicators $\boldsymbol{g}_r$ from F_r using eq.(2.8)
  4: Calculate $sSelect_r$, the score of the feature F_r using eq.(2.9)
end for
5: Rank the features according to their scores in descending order.

The first term of eq.(2.9) calculates the cut-value of using g_r as the cluster indicator for data X. The second term estimates the corresponding classification loss of $\hat{g}_r$ according to the labeled data. The ideal case is that all labeled data in each cluster come from the same class. The algorithm of sSelect is summarized in Algorithm 2. Note that sSelect also relies on a good choice of the k-nearest neighborhood parameter.

Later, the authors exploited intrinsic properties underlying supervised and unsupervised feature selection algorithms, and proposed a unified framework for feature selection based on spectral graph theory [Chung, 1997].

2.7.3 Semi-Supervised Dimensionality Reduction (SSDR)

The authors in [Zhang et al., 2007] proposed semi-supervised dimensionality reduction algorithm (SSDR), which can preserve the structure of the labeled and unlabeled data in the projected low-dimensional space. The labeled data is expressed by the must-link and the cannot-link constraints. The SSDR algorithm was proposed with different variants: SSDR-M, SSDR-CM and SSDR-CMU, where M stands for Must-Link constraints, C for Cannot-Link constraints and U stands for unlabeled data. Authors formulated their method as follows:

Given a set of data instances $X = \{x_1, x_2, \ldots, x_n\}$ together with some pairwise must-link constraints Ω_{ML} and cannot-link constraints Ω_{CL}, the idea is to find a set of projective vectors $g = [g_1, g_2, \ldots, g_d]$ where d represents the dimension of vectors (to be set), such that the transformed low-dimensional representations (denoted by $\mathsf{Y} = \{\mathsf{Y}_1, \ldots, \mathsf{Y}_d\}$ where $\mathsf{Y}_i = g^T x_i$) can preserve the structure of the original dataset as well as the pairwise constraints Ω_{ML} and Ω_{CL}. To do that, the authors define the objective function as maximizing $J(g)$ w.r.t. $g^T g = 1$, where

$$J(g) = \frac{1}{2n^2}\sum_{i,j}(\mathsf{Y}_i - \mathsf{Y}_j)^2 \quad + \quad \frac{\alpha}{2\,|\Omega_{CL}|}\sum_{(y_i,y_j)\in\Omega_{CL}}(\mathsf{Y}_i - \mathsf{Y}_j)^2 \quad - \quad \frac{\beta}{2\,|\Omega_{ML}|}\sum_{(x_i,x_j)\in\Omega_{ML}}(\mathsf{Y}_i - \mathsf{Y}_j)^2$$

$$= \frac{1}{2n^2}\sum_{i,j}(g^T x_i - g^T x_j)^2 \quad + \quad \frac{\alpha}{2\,|\Omega_{CL}|}\sum_{(x_i,x_j)\in\Omega_{CL}}(g^T x_i - g^T x_j)^2 \\ - \quad \frac{\beta}{2\,|\Omega_{ML}|}\sum_{(x_i,x_j)\in\Omega_{ML}}(g^T x_i - g^T x_j)^2 \qquad (2.11)$$

where α and β are scaling parameters to balance the contribution of the corresponding terms, since the distance between instances in the same class is typically smaller than that in different classes. The idea behind the proposed objective function is to let the average distance in the transformed low-dimensional space between instances involved by the cannot-link set Ω_{CL} as large as possible, while distances between instances involved by the must-link set Ω_{ML} as small as possible. Then, in order to propose the variant version of the score, the authors proposed a concise from eq.(2.11):

$$\begin{aligned} J(g) &= \frac{1}{2}\sum_{i,j}(\mathsf{Y}_i - \mathsf{Y}_j)^2 \boldsymbol{S}_{ij} \\ &= \frac{1}{2}\sum_{i,j}(g^T x_i - g^T x_j)^2 \boldsymbol{S}_{ij} \end{aligned} \qquad (2.12)$$

where

$$\boldsymbol{S}_{ij} = \begin{cases} \frac{1}{n^2} + \frac{\alpha}{|\Omega_{CL}|} & if\ (x_i, x_j) \in \Omega_{CL} \\ \frac{1}{n^2} - \frac{\beta}{|\Omega_{ML}|} & if\ (x_i, x_j) \in \Omega_{ML} \\ \frac{1}{n^2} & otherwise \end{cases} \qquad (2.13)$$

Based on spectral graph theory, the Authors proved that the equation eq.(2.12) can be rewritten as maximizing $J(g)$ w.r.t $g^T g = 1$, where:

$$J(g) = g^T X \boldsymbol{L} X^T g \qquad (2.14)$$

where $\boldsymbol{L}$ is the Laplacian matrix, and the problem expressed by eq.(2.14) is an eigen-problem, which can be solved by computing the eigenvectors of $\mathcal{L} = X\boldsymbol{L}X^T$ corresponding to the largest eigenvalues.

This formulation with the weight matrix S allowed to have three variants of

Algorithm 3 SSDR-CMU

Input: Dataset X, pairwise constraints sets Ω_{ML} and Ω_{CL}, dimension of projective vectors d
Output: the dimensionality-reduced data matrix
1: Build G the dissimilarity graph of data X
2: Calculate the dissimilarity matrix $\boldsymbol{S}$ using eq.(2.13)
3: Calculate $\mathcal{L} = X\boldsymbol{L}X^T$ in order to solve eq.(2.13)
4: Calculate the eigenvectors and eigenvalues of $\mathcal{L}$
5: Sort the eigenvalues with the corresponding eigenvectors in descendant order
6: Construct the g matrix corresponding the top d sorted eigenvectors
7: Calculate the new dimensionality-reduced data matrix $\mathsf{Y} = g^T x$

SSDR score, they are denoted as:

- SSDR-M: Using only the must-link constraints, with

$$\boldsymbol{S}_{ij} = \begin{cases} -\frac{\beta}{|\Omega_{ML}|} & if\ (x_i, x_j) \in \Omega_{ML} \\ 0 & otherwise \end{cases} \tag{2.15}$$

- SSDR-CM: Using both the cannot-link and must-link constraints, with

$$\boldsymbol{S}_{ij} = \begin{cases} \frac{\alpha}{|\Omega_{CL}|} & if\ (x_i, x_j) \in \Omega_{CL} \\ -\frac{\beta}{|\Omega_{ML}|} & if\ (x_i, x_j) \in \Omega_{ML} \\ 0 & otherwise \end{cases} \tag{2.16}$$

- SSDR-CMU: Using both the cannot-link and must-link constraints together with unlabeled data, with the weights $\boldsymbol{S}$ defined in eq.(2.13).

The algorithm of SSDR-CMU is detailed in Algorithm 3.

2.7.4 Constraint Score (CS)

The SSDR authors proposed a constraint score-based method (CS) which evaluates the relevance of features according to constraints only [Zhang et al., 2008].

Algorithm 4 CS

Input: Dataset X, pairwise constraints sets Ω_{ML}, Ω_{CL} and ν (for **Constraint Score-2** only)
Output: The ranked features list
for $r = 1$ **to** m **do**
1: Calculate CS_r, the score of F_r using eq.(2.17) for **Constraint Score-1** or eq.(2.18) for **Constraint Score-2**
end for
2: Rank the features according to their scores in ascending order.

They showed that using few labels of data, this score records better results than Fisher score [Duda et al., 2000], which employs all labels in feature selection process. They defined two different Constraint scores for evaluating the relevance of the r^{th} feature F_r, which should be minimized, as follows :

$$CS_r^1 = \frac{\sum_{(x_i,x_j)\in\Omega_{ML}}(f_{ri} - f_{rj})^2}{\sum_{(x_i,x_j)\in\Omega_{CL}}(f_{ri} - f_{rj})^2} \tag{2.17}$$

$$CS_r^2 = \sum_{(x_i,x_j)\in\Omega_{ML}} (f_{ri} - f_{rj})^2 - \nu \sum_{(x_i,x_j)\in\Omega_{CL}} (f_{ri} - f_{rj})^2 \tag{2.18}$$

where ν is a regularization coefficient whose function is to balance the contributions of the two terms in eq.(2.18). The Algorithm of CS is summarized in Algorithm 4.

The authors presented a spectral graph formulation of their scores, to do this, they construct two graphs G^M and G^C and both with n nodes, using the pairwise constraints in Ω_{ML} and Ω_{CL} respectively. In both graphs, the i^{th} node corresponds to the i^{th} instance. The edges in both graphs represent the pairwise constraints, i.e. an edge exist between node i and j in G^M (or in G^C) graph if there is a must-link constraint (or a cannot-link constraint) between instances, then they define their weight matrices, denoted by $\boldsymbol{S}^M$ and $\boldsymbol{S}^C$, respectively, as:

$$\boldsymbol{S}_{ij}^M = \begin{cases} 1 & if (x_i, x_j) \in \Omega_{ML} or (x_j, x_i) \in \Omega_{ML} \\ 0 & otherwise \end{cases} \tag{2.19}$$

$$\boldsymbol{S}_{ij}^{C} = \begin{cases} 1 & if(x_i, x_j) \in \Omega_{CL} or (x_j, x_i) \in \Omega_{CL} \\ 0 & otherwise \end{cases} \tag{2.20}$$

After calculation of Diagonal and Laplacian matrices, they get:

$$CS_r^1 = \frac{f_r^T \boldsymbol{L}^M f_r}{f_r^T \boldsymbol{L}^C f_r} \tag{2.21}$$

and

$$CS_r^2 = f_r^T \boldsymbol{L}^M f_r - \nu f_r^T \boldsymbol{L}^C f_r \tag{2.22}$$

The method carries out with little supervision information in labeled data ignoring the unlabeled data part even if it is very large.

2.7.5 Bagging Constraint Score (BCS)

The major drawback of the Constraint Score is that its performance is dependent on a good choice of the composition and cardinality of constraint set, which is very challenging in practice. Later, the same authors addressed the problem by importing Bagging into Constraint Score and proposed a Bagging Constraint Score (BCS) method [Sun and Zhang, 2010].

Instead of seeking one appropriate constraint set for single Constraint Score. The authors of BCS performed multiple Constraint Scores (CS), each of which uses a bootstrapped subset of original given constraint set. Diversity analysis on instances of ensemble showed that resampling pairwise constraints is helpful for simultaneously improving accuracy and diversity of instances.

The authors tackled the problem of feature selection with pairwise constraints from the ensemble perspective with the goal of improving classification accuracy. Their method is based on bootstrapping and aggregating concepts.

The algorithm, called Bagging Constraints Score (BCS), constructs individual components using different constraint subsets generated by resampling

Algorithm 5 BCS

Input: training Data $X_L = \{x_i\}_{i=1}^{l}$,
Base learning algorithm Learner[3],
Must-link constraint set Ω_{ML},
Cannot-link constraint set Ω_{CL},
ν (for **Constraint Score-2** only), number of selected features N_f, ensemble size EL,
Class labels $Y_L = \{y_i\}_{i=1}^{l}$ corresponding to X_L
Output: The final hypothesis
for $b = 1$ **to** EL **do**
1: Take a bootstrapped sample M^b of the must-link constraints set Ω_{ML} and a sample C^b of the cannot-link constraints set Ω_{CL};
for $r = 1$ **to** m **do**
2: Calculate CS_r, the score of F_r using eq.(2.17) for **Bagging Constraint Score-1** or eq.(2.18) for **Bagging Constraint Score-2**
end for
3: Rank the features according to their scores in ascending order
4: Get the training dataset $XT = \{xt_i\}_{i=1}^{t}$ (where t is the size of the training dataset) projected to subspace by selecting the n_f highest-scoring features only;
5: Call Learner, providing it with the training dataset XT;
6: Get a hypothesis $h_b : XT \rightarrow Y_L$;
end for
7: The final hypothesis by combining the outputs of EL learners [4] as follows: $h_f(X) = \arg\max_{y \in Y, xt \in XT} \sum_{b:h_b(xt)=y} 1$

pairwise constraints in the given constraint set (Algorithm 5).

However, the method is still depending entirely on the labeled part of data only, which is generally small in the semi-supervised feature selection applications. In addition, ignoring the unlabeled part of data which is usually huge in semi-supervised learning may hide important information about the target concept, and then misleading the learning process.

[3] In [Sun and Zhang, 2010], the authors chose the Nearest Neighborhood (1-NN) and Support Vector Machine (SVM) classifiers as learners.
[4] The authors adopted majority vote.

2.7.6 Semi-Supervised Selection with Constraint score (SC4)

The authors in [Kalakech et al., 2011] proposed to solve the problem of semi-supervised feature selection by a simple combination of scores computed on labeled data and unlabeled data respectively. The method (called SC4) tries to find a consensus between an unsupervised score (Laplacian Score) and a supervised one (Constraint Score) by multiplying both scores. The proposed score to be minimized is defined as:

$$SC4_r = LS_r.CS_r \tag{2.23}$$

The algorithm of SC4 is presented in Algorithm 6. The combination is simple,

Algorithm 6 SC4

Input: Dataset X, pairwise constraints sets Ω_{ML}, Ω_{CL} and λ (for Laplacian score)
Output: The ranked features
for $r = 1$ **to** m **do**
1: Calculate LS_r, the Laplacian score of F_r using Algorithm 1
2: Calculate CS_r, the Constraint score of F_r using Algorithm 4
3: Calculate $SC4_r$, the score of F_r using eq.(2.23)
end for
4: Rank the features according to their scores in ascending order.

but can dramatically bias the selection for the features having best scores for labeled part of data and bad scores for the unlabeled part and vice-versa.

2.8 Wrapper approaches

The Wrapper methods perform a search in the space of feature subsets, guided by the outcome of the learning model. Typically, a criterion is firstly defined for evaluating the quality of a candidate feature subset and wrapper approaches aim to identify the best subset such that the learning algorithm can achieve the optimal value of the predefined criterion.

Algorithm 7 FW-SemiFS

Input: l, u, $sizeFS$, $samplingRate$, $samplingTimes$, $maxIterations$, $startfn$, $fnstep$
Output: $resultfs$
1: Perform feature selection on l using $SFFS$, select $startfn$ features form the current feature subset $currentfs$;
2: $ReducedL \leftarrow l * currentfs$;
3: $ReducedU \leftarrow u * currentfs$;
for $iteration = 1$ **to** $maxIterations$ **do**
  4: $Predicted \leftarrow$ classifier($ReducedL$, $ReducedU$);
  for $rand = 1$ **to** $samplingTimes$ **do**
    5: Randomly select samplingRate% of instances from Predicted, and add it into l to form a new dataset $NewDataset$;
    6: Perform feature selection on $NewDataset$ using $SFFS$, select $fnstep$ features to form feature subset $fs\,[rand]$;
  end for
  7: Count the frequency of every feature in fs, add the most frequent and not in $currentfs$ feature into $currentfs$;
  8: $ReducedL \leftarrow l * currentfs$;
  9: $ReducedU \leftarrow u * currentfs$;
  10: if $SIZE(currentfs) == sizeFS$ then break;
end for
11: $resultfs \leftarrow currentfs$;

2.8.1 Forward Semi-Supervised Feature Selection (FW-SemiFS)

The authors in [Ren et al., 2008] introduced a "wrapper-type" forward semi-supervised feature selection framework (FW-SemiFS). They extended the Supervised Sequential Forward Feature Selection (SFFS) [Pudil et al., 1994]. This algorithm is an iterative process starting with an empty feature subset. In each iteration, one feature is chosen among the remaining features. To determine which feature to add, it tests the accuracy of a model built on the incremented feature subset. Then, the feature that results in the highest accuracy is selected. The process terminates when no additional features could result in an improvement in accuracy or the feature subset already reaches a predefined size. This method is supervised, i.e. it concerns labeled examples only, so the authors proposed (FW-SemiFS) in order to extend it to take unlabeled data into account, which makes it suitable to be used with semi-supervised data.

(FW-SemiFS) uses SFFS as a wrapper model to select initial features $startfn$ to be used to train a given classifier [5]. This learner is then used to predict the labels of the unlabeled data. Then, a randomly selected unlabeled data $samplingRate\%$ with predicted labels, is combined with labeled data to form a new training set. Afterwards, the new obtained training dataset is used to select a new feature subset $fnstep$ based on SFFS and the learner. The above processes repeat $samplingTimes$ times, and then $samplingTimes$ groups of features are selected. The method counts the frequency of every feature in the $samplingTimes$ groups of features, and the one with the most frequency is added to form a new feature subset. This process is repeating until the size of the feature subset reaches a predefined number.

The detailed algorithm is presented in Algorithm 7, where:

- l and u are the sizes of labeled and unlabeled data respectively.
- $sizeFS$ is the predefined number of selected features.
- $samplingRate$ is the sampling rate according to the unlabeled data with predicted labels.
- $samplingTimes$ is the randomly sampling times.
- $maxIterations$ is the maximal number of iterations.
- $startfn$ is the start feature number.
- $fnstep$ is the number of features selected in every step.
- $resultfs$ is the output feature subset.

In this algorithm, "*" denotes the features reduction operator.

[5] In [Ren et al., 2008], the authors used NaiveBayes, NNge (the nearest neighbor like algorithm using non-nested generalized instances), and *k*-NN classifiers.

2.8.2 Semi-Supervised Feature Importance evaluation (SSFI)

The authors in [Barkia et al., 2011] proposed a semi-supervised feature importance evaluation method (SSFI), that combines ideas from co-training [Blum and Mitchell, 1998] and random forests (RF) [Breiman, 2001] with a new permutation-based out-of-bag feature importance measure. The algorithm ranks features through an ensemble framework, in which a feature's relevance is evaluated by its predictive accuracy using both labeled and unlabeled data. SSFI combines both data resampling (*bagging*) and random subspace strategies for generating an ensemble learner using a co-training style algorithm. The authors claim that a combination of these two main strategies for producing ensemble of classifiers leads to an exploration of distinct views of inter-pattern relationships. Once each ensemble member is obtained, an extension of the RF permutation importance measure [Breiman, 2001], using the labeled and unlabeled data together, is proposed to measure the feature relevance. A ranking of all features is finally obtained with respect to their relevance in all obtained semi-supervised classifiers. Later, the same authors proposed a new method called semi-supervised ensemble learning guided feature ranking method (SEFR)[Bellal et al., 2012], the algorithm ranks features through an ensemble framework, in which a feature relevance is evaluated by its predictive accuracy using both labeled and unlabeled data. The proposed methods presented promising experimental results. However, the computational complexity of such methods is still a critical issue especially when data is high-dimensional.

2.9 Embedded approaches

Embedded feature selection methods are locally specific to a model during its construction.They aim to learn the feature relevance with the associated learning algorithm. In other terms, they incorporate feature selection and learning algorithm in the same objective function. In the following, we will discuss one of the semi-supervised embedded feature selection approaches.

2.9.1 Semi-Supervised Feature Selection via Manifold Regularization (FS-Manifold)

The authors in [Xu et al., 2009a] proposed a discriminative embedded and semi-supervised feature selection method based on the manifold regularization (FS-Manifold). The authors claimed that the regularization in the proposed feature selection method assures that the decision function is smooth on the manifold constructed by the selected features of the unlabeled data. This exploits the underlying structural information of these data. The proposed method selects features through maximizing the classification margin between different classes and simultaneously exploiting the geometry of the probability distribution of both unlabeled and labeled data. Moreover, the authors formulated their semi-supervised feature selection method into a concave-convex problem, where the saddle point corresponds to the optimal solution. Then, they derived an extended level method [Xu et al., 2009b], a fairly recent optimization method, to find the optimal solution of the concave-convex problem.

2.10 Conclusion

In this chapter we reviewed the literature of semi-supervised feature selection as a dimensionality reduction tool. We started by a brief presentation of the general domain of dimensionality reduction. Then, we distinguished between feature extraction which transforms the problem from the original features space into a reduced space with new features representing the original ones. We briefly illustrated some well-known methods for feature extraction. Then, we reviewed the other part of dimensionality reduction which is the feature selection, in which the relevant features are selected, and the others are removed. We illustrated the problem of feature selection in both domains: supervised and unsupervised with citation of the representative methods in each domain. After that, we focused on the semi-supervised feature selection, which is seen as a challenging problem due to the presence of a small sample of labeled instances, with a large amount of unlabeled ones. This domain is rather new, and its “small labeled-sample” problem is still worth studying. From this domain,

we illustrated in details the main approaches that were proposed to deal with both labeled and unlabeled instances (i.e. semi-supervised data). In the next chapters, we present some algorithms which we propose to efficiently solve the problem of semi-supervised feature selection.

3 Constrained Laplacian Scores for Semi-Supervised Feature Selection

3.1 Introduction

One important motivation to have a filter method for feature selection is the specificity of the semi-supervised data. This is because, in this context, data may be used in the service of both unsupervised and supervised learning. On the one hand, semi-supervised data could be used in the goal of data clustering, then using the labels to generate constraints which could, in turn, improve the clustering task. In this sense, ”good” features are those which better describe the geometric structure of data. On the other hand, semi-supervised data could be used for supervised learning, i.e. classification or prediction of the unlabeled examples, using a classifier constructed from the labeled ones. In this context, "good" features are those which are better correlated with the labels. Subsequently, the use of a filter method makes the feature selection process independent from the further learning algorithm whether it is supervised or unsupervised. This is important to eliminate the bias of feature selection in both cases, i.e. ”good” features in this case would be those which compromise between better description of data structure and better correlation with desired labels.

3.2 Discussion about Constraint and Laplacian scores

The main advantage of Laplacian score [He et al., 2005] is its locality preserving ability. However, its assumption that data from the same class are close to each other, is not always true. In fact, there are several cases where the classes overlap in some instances. Thus, two close instances can naturally have two different labels and vice-versa. In the semi-supervised context, Laplacian score takes into consideration the class labels if they exist (instances sharing the same label are considered as neighbors). The authors claimed that it would be suitable for the semi-supervised learning. In fact, having the same label (which can generate ML pairwise constraints) adds an important information over unlabeled data. However, other information can be obtained from the instances which have different labels (i.e. which are linked by CL constraints). This may be of high importance if instances from different classes are close to each other (i.e. they are neighbors).

For Constraint score [Zhang et al., 2008], the principle is based entirely on the constraint preserving ability. This method, with few labels, showed an interesting performance comparing with Fisher score [Duda et al., 2000] which use all labels. However, the score imposes that the exploited constraints are well chosen. Hence, the results are biased of results towards the selected constraints. For that reason, the same authors proposed to add more diversity to constraint choice by a bagging constraint score [Sun and Zhang, 2010]. Moreover, Constraint score ignores the unlabeled part of data which carries important information about the structure. The important issue in semi-supervised learning is that the labeled and unlabeled instances are sampled from the same population, so the information included in the structure and the other supplied by the background knowledge (labels) are expected to complementally describe the target concept. Subsequently, we consider that the exploitation of both labeled and unlabeled parts of data is very important for semi-supervised feature selection. From this consideration we inspired our first semi-supervised feature selection score CLS, that we describe in the next section.

3.3 Constrained Laplacian Score (CLS)

The basis idea of CLS is the constraining of Laplacian score by the background information extracted from the labeled data.

The goal of CLS is to assess the ability of features in preserving the local geometric structure offered by unlabeled data, while respecting the constraints offered by labeled data.

For a feature F_r, we define CLS_r, which should be minimized, as follows:

$$CLS_r = \frac{\sum_{i,j}(f_{ri} - f_{rj})^2 \boldsymbol{S}_{ij}}{\sum_i \sum_{j|\exists k,(x_k,x_j)\in\Omega_{CL}} (f_{ri} - \alpha^i_{rj})^2 \boldsymbol{D}_{ii}} \tag{3.1}$$

where :

$$\boldsymbol{S}_{ij} = \begin{cases} e^{-\frac{\|x_i - x_j\|^2}{\lambda}} & \text{if } x_i \text{ and } x_j \text{ are neighbors or } (x_i, x_j) \in \Omega_{ML} \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

and:

$$\alpha^i_{rj} = \begin{cases} f_{rj} & \text{if } (x_i, x_j) \in \Omega_{CL} \\ \mu_r & \text{otherwise} \end{cases} \tag{3.3}$$

Note that if there are no labels ($l = 0$ and $X = X_U$) then $CLS_r = LS_r$ and when ($u = 0$ and $X = X_l$), CLS represents an adjusted CS_r, where the ML and CL information would be weighted by $\boldsymbol{S}_{ij}$ and $\boldsymbol{D}_{ii}$ respectively in the formula.

With CLS, on the one hand, a relevant feature should be the one on which those two instances (neighbors or related by an ML constraint) are close to each other. On the other hand, the relevant feature should be the one with a larger variance or on which those two instances (related by a CL constraint) are well separated.

3.3.1 Spectral graph based formulation of CLS

In this section, we give a spectral graph-based explanation for our proposed score. A reasonable criterion to choose a relevant feature is to minimize the object function represented by CLS. Thus, the problem is to minimize the first term $T_1 = \sum_{i,j}(f_{ri} - f_{rj})^2 \boldsymbol{S}_{ij}$ and maximize the second one $T_2 = \sum_i \sum_{j|\exists k,(x_k,x_j)\in\Omega_{CL}} (f_{ri} - \alpha^i_{rj})^2 \boldsymbol{D}_{ii}$. By resolving these two optimization problems, we prefer those features respecting their pre-defined graphs, respectively. Thus, we construct a k-nearest neighborhood graph G_{kn} from X (dataset) and Ω_{ML} (ML constraint set) and a second graph G_{CL} from Ω_{CL} (CL constraint set).

Given a dataset X, let $G(V,E)$ be the complete undirected graph constructed from X, with V is its node set and E is its edge set. The i^{th} node v_i of G corresponds to $x_i \in X$ and there is an edge between each node pair (v_i, v_j), whose weight $w_{ij} = e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}}$ is the dissimilarity between x_i and x_j.

$G_{kn}(V, E_{kn})$ is a subgraph which could be constructed from G where E_{kn} is the edge set $\{e_{i,j}\}$ from E such that $e_{i,j} \in E_{kn}$ if $(x_i, x_j) \in \Omega_{ML}$ or x_i is one of the k-nearest neighbors of x_j. $G_{CL}(V_{CL}, E_{CL})$ is a subgraph constructed from G with V_{CL}, its node set, and $\{e_{i,j}\}$, its edge set, such that $e_{i,j} \in E_{CL}$ if $(x_i, x_j) \in \Omega_{CL}$.

Once the graphs G_{kn} and G_{CL} are constructed, their weight matrices, denoted by $\boldsymbol{S}^{kn}$ and $\boldsymbol{S}^{CL}$ respectively, can be defined as:

$$\boldsymbol{S}^{kn}_{ij} = \begin{cases} w_{ij} & \text{if } x_i \text{ and } x_j \text{ are neighbors or } (x_i, x_j) \in \Omega_{ML} \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

$$\boldsymbol{S}^{CL}_{ij} = \begin{cases} 1 & if (x_i, x_j) \in \Omega_{CL} \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

Then, we can define :

- For each feature F_r, its vector $f_r = (f_{r1}, ..., f_{rn})^T$
- Diagonal matrices $\boldsymbol{D}^{kn}_{ii} = \sum_j \boldsymbol{S}^{kn}_{ij}$ and $\boldsymbol{D}^{CL}_{ii} = \sum_j \boldsymbol{S}^{CL}_{ij}$

- Laplacian matrices $\boldsymbol{L}^{kn} = \boldsymbol{D}^{kn} - \boldsymbol{S}^{kn}$ and $\boldsymbol{L}^{CL} = \boldsymbol{D}^{CL} - \boldsymbol{S}^{CL}$

Following some simple algebraic steps, we see that:

$$\begin{aligned} T_1 &= \sum_{i,j}(f_{ri} - f_{rj})^2 \boldsymbol{S}_{ij}^{kn} \\ &= \sum_{i,j}(f_{ri}^2 + f_{rj}^2 - 2f_{ri}f_{rj})\boldsymbol{S}_{ij}^{kn} \\ &= \sum_{i,j} f_{ri}^2 \boldsymbol{S}_{ij}^{kn} + \sum_{i,j} f_{rj}^2 \boldsymbol{S}_{ij}^{kn} - 2\sum_{i,j} f_{ri}\boldsymbol{S}_{ij}^{kn} f_{rj} \\ &= 2(\sum_{i,j} f_{ri}^2 \boldsymbol{S}_{ij}^{kn} - \sum_{i,j} f_{ri}\boldsymbol{S}_{ij}^{kn} f_{rj}) \\ &= 2(f_r^T \boldsymbol{D}^{kn} f_r - f_r^T \boldsymbol{S}^{kn} f_r) \\ &= 2 f_r^T \boldsymbol{L}^{kn} f_r \end{aligned} \tag{3.6}$$

Note that satisfying the graph-structures is done according to α_{rj}^i in the eq.(3.3). Indeed, when $\Omega_{CL} = \varnothing$ then $\alpha_{rj}^i = \mu_r$, we should maximize the variance of f_r. Recall that the variance of a random variable x can be written as follows:

$$Var(x) = \int_{\mathcal{M}} (x-\mu)^2 dP(x),\ \mu = \int_{\mathcal{M}} x dP(x) \tag{3.7}$$

where $\mathcal{M}$ is the data manifold, μ is the expected value of x and dP is the probability measure. By spectral graph theory [Chung, 1997], dP can be estimated by the diagonal matrix $\boldsymbol{D}$ on the sample points. Thus, the *weighted* data variance can be estimated as follows:

$$Var(f_r) = \sum_i (f_{ri} - \mu_r)^2 \boldsymbol{D}_{ii}^{kn} \tag{3.8}$$

$$\mu_r = \sum_i \left(f_{ri} \frac{\boldsymbol{D}_{ii}}{\sum_i \boldsymbol{D}_{ii}} \right) = \frac{1}{(\sum_i \boldsymbol{D}_{ii})} \left(\sum_i f_{ri}\boldsymbol{D}_{ii} \right) = \frac{f_r^T \boldsymbol{D}\mathbf{1}}{\mathbf{1}^T \boldsymbol{D}\mathbf{1}} \tag{3.9}$$

To remove the mean from the samples, we define:

$$\tilde{f}_r = f_r - \frac{f_r^T \boldsymbol{D}\mathbf{1}}{\mathbf{1}^T \boldsymbol{D}\mathbf{1}}\mathbf{1} \tag{3.10}$$

Thus,

$$Var(f_r) = \sum_i \tilde{f}_{ri}^2 \boldsymbol{D}_{ii} = \tilde{f}_r^T \boldsymbol{D} \tilde{f}_r \tag{3.11}$$

Also, it is easy to show that $\tilde{f}_r^T \boldsymbol{D} \tilde{f}_r = f_r^T \boldsymbol{D} f_r$ [Kalakech et al., 2011]. In this case, $CLS_r = L_r = \frac{f_r^T \boldsymbol{L}^{kn} f_r}{f_r^T \boldsymbol{D}^{kn} f_r}$.

Otherwise, $\alpha_{rj}^i = f_{rj}$ we develop as above the second term (T_2) as follows:

$$\begin{aligned}
T_2 &= \sum_{i,j} (f_{ri} - f_{rj})^2 \boldsymbol{D}_{ii}^{kn} \\
&= \sum_{i,j} (f_{ri} - f_{rj})^2 \boldsymbol{S}^{CL} \boldsymbol{D}_{ii}^{kn} \\
&= \sum_{i,j} (f_{ri}^2 + f_{rj}^2 - 2 f_{ri} f_{rj}) \boldsymbol{S}^{CL} \boldsymbol{D}_{ii}^{kn} \\
&= \sum_{i,j} f_{ri}^2 \boldsymbol{S}^{CL} \boldsymbol{D}_{ii}^{kn} + \sum_{i,j} f_{rj} \boldsymbol{S}^{CL} \boldsymbol{D}_{ii}^{kn} - 2 \sum_{i,j} f_{ri} \boldsymbol{S}^{CL} \boldsymbol{D}_{ii}^{kn} f_{rj} \\
&= 2(\sum_{i,j} f_{ri}^2 \boldsymbol{S}^{CL} \boldsymbol{D}_{ii}^{kn} - \sum_{i,j} f_{ri} \boldsymbol{S}^{CL} \boldsymbol{D}_{ii}^{kn} f_{rj}) \\
&= 2(f_r^T \boldsymbol{D}^{CL} \boldsymbol{D}^{kn} f_r - f_r^T \boldsymbol{S}^{CL} \boldsymbol{D}^{kn} f_r) \\
&= 2 f_r^T \boldsymbol{L}^{CL} \boldsymbol{D}^{kn} f_r
\end{aligned} \tag{3.12}$$

Subsequently,

$$CLS_r = \frac{f_r^T \boldsymbol{L}^{kn} f_r}{f_r^T \boldsymbol{L}^{CL} \boldsymbol{D}^{kn} f_r} \tag{3.13}$$

In fact, eq.(3.13) seeks those features that respect both G_{kn} and G_{CL}. The whole procedure of the proposed CLS is summarized in Algorithm 8.

Lemma 1. *Algorithm 8 is computed in time* $O(m \times max(n^2, Log\, m))$.

Proof. The first step of the algorithm requires l^2 operations. Steps 2-3 build the graph matrices requiring n^2 operations. Step 4 evaluates the m features

Algorithm 8 CLS

Input: Dataset $X(n \times m)$, the constant λ, the neighborhood degree k
1: Construct the constraint sets (Ω_{ML} and Ω_{CL}) from Y_L
2: Construct the graphs G_{kn} and G_{CL} from (X, Ω_{ML}) and Ω_{CL} respectively.
3: Calculate the weight matrices $\boldsymbol{S}^{kn}$, $\boldsymbol{S}^{CL}$ and their Laplacians $\boldsymbol{L}^{kn}$, $\boldsymbol{L}^{CL}$ respectively.
for $r = 1$ **to** m **do**
  4: Calculate CLS_r according to eq.(3.13).
end for
5: Rank the features F_r according to their scores CLS_r in ascending order.

requiring mn^2 operations and the last step ranks features according to their scores with $m\ Log(m)$ operations. □

Note that the "small labeled-sample" problem becomes an advantage for the complexity of CLS, because it supposes that the number of extracted constraints is smaller since it depends on the number of labels, l. Thus, the cost of the algorithm depends considerably on u, the size of unlabeled data X_U.

To reduce this complexity, we propose to apply a clustering on X_U. The idea aims to substitute this huge part of data by a smaller one $X'_U = (c_1, ..., c_K)$ by preserving the geometric structure of X_U, where K is the number of clusters. We propose to use Self-Organizing Map (SOM) based clustering [Kohonen, 2001] that we briefly present in the next section.

Lemma 2. *By clustering X_U the complexity of Algorithm 8 is reduced to $O(m \times max(u, Log\ m))$.*

Proof. The size of labeled data is very smaller than the one of unlabeled data, $l << u < n$ and the clustering of X_U provides at most $K = \sqrt{u}$ clusters. Therefore, Algorithm 8 is applied over a dataset with size equal to $\sqrt{u} + l \simeq \sqrt{u}$. This allows to decrease the complexity to $O(m \times max(u, Log\ m))$. □

3.3.2 SOM' algorithm

SOM is a very popular tool used for clustering high dimensional data spaces [Kohonen, 2001]. It can be considered as undertaking vector quantization and/or clustering while preserving the spatial ordering of the input data by implementing an ordering of the codebook vectors (also called prototype vectors, cluster centroids or reference vectors) in a one or two dimensional output space. SOM consists of nodes organized on a regular low-dimensional grid, called the map. More formally, the map is described by a graph (V, E). V is a set of K interconnected nodes having a discrete topology defined by E. For each pair of nodes (v, z) on the map, the distance $\Delta(v, z)$ is defined as the shortest path between v and z on the graph (Figure 3.1). This distance imposes a neighborhood relation between nodes.

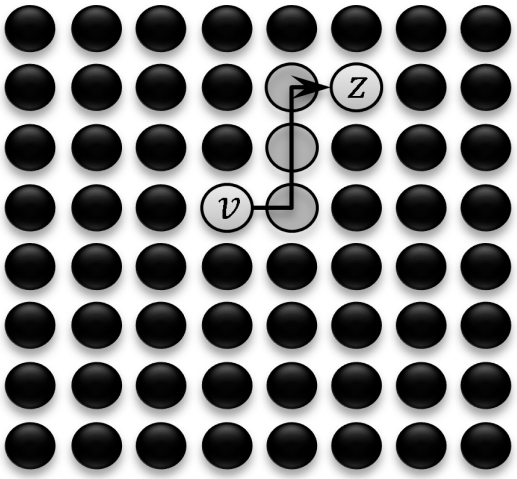


Figure 3.1: SOM architecture - the path between v and z is $\Delta(v, z) = 4$.

Each node v is represented by an m-dimensional reference vector $c_v = c_v^1, ..., c_v^m$ from $\mathcal{M}$ (the set of all map's nodes), where m is equal to the dimension of the input vectors $x_i \in X_U$ (unlabeled dataset). The SOM training algorithm resembles *k*-means. The important distinction is that in addition to the best matching reference vector, its neighbors on the map are updated.

More formally, we define an assignment function γ from $\mathbb{R}^m$ (the input space) to $\mathcal{M}$ (the output space), that associates each element x_i of $\mathbb{R}^m$ to the node whose reference vector is "closest" to x_i. This function induces a partition

$P = P_v; v = 1...K$ of the set of instances where each part P_v is defined by: $P_v = \{x_i \in X_U; \gamma(x_i) = v\}$.

Next, an adaptation step is performed when the algorithm updates the reference vectors by minimizing a cost function, noted $\mathcal{E}(\gamma, \mathcal{M})$. This function has to take into account the inertia of the partition P, while ensuring the topology preserving property. To achieve these two goals, it is necessary to generalize the inertia function of P by introducing the neighborhood notion attached to the map. In the case of instances belonging to $\mathbb{R}^m$, this minimization can be done straightforwardly. Indeed, new reference vectors are calculated as:

$$c_z^{t+1} = \frac{\sum_{i=1}^{c} \tau_{vz}(t) x_i}{\sum_{i=1}^{c} \tau_{vz}(t)} \tag{3.14}$$

where $v = arg\min_z \|x_i - c_z^t\|$, is the index of the best matching unit of the data sample x_i, $\|.\|$ is the distance measure, typically the Euclidean distance, and t denotes the time. $\tau_{vz}(t)$ is the neighborhood function around the winner unit

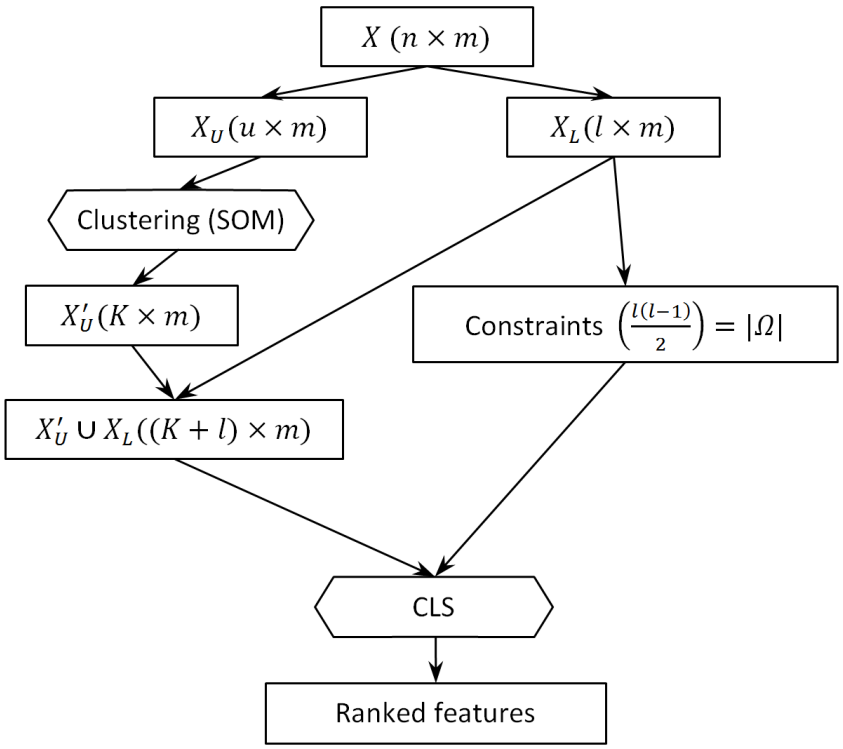


Figure 3.2: Semi-supervised feature selection framework of CLS.

v. In practice, we often use $\tau_{vz} = e^{-\frac{\Delta_{vz}}{2T^2}}$ where T represents the neighborhood radius in the map. It is decreased from an initial value T_{max} to a final value T_{min}.

Subsequently, as explained above, SOM will be applied on the unsupervised part of data (X_U) to obtain X'_U with a size equal to the number of SOM' nodes (K). Therefore, CLS will be performed on the new obtained dataset ($X_L \cup X'_U$). Note that any other clustering method could be applied over X_U, but here, SOM is chosen for its ability to preserve the topological relationship of data well and thus the geometric structure of their distribution. Finally, the feature selection framework is represented in the Figure 3.2.

3.4 Constrained Selection-based Feature Selection (CSFS)

In this section, we present a novel framework for semi-supervised feature selection. In fact, a critical study of CLS concept reveals a number of interesting potential avenues for improving its efficiency. For example, the choice of neighborhood degree (k) might be interesting to be analyzed. Another possible improvement might be to study the constraints utility before integrating them for feature selection. In CLS, we used the maximum number of constraints which can be generated from the labeled data ($\frac{l(l-1)}{2}$). This can have ill effects over accuracy when constraints are incoherent or inconsistent (as we would see later)[Davidson et al., 2006, Allab and Benabdeslem, 2011]. Thus, it would have been more interesting to investigate in constraint selection process. This led us to develop a more efficient semi-supervised feature selection score that we call: CSFS.

Principally, CSFS framework is based on efficient selection of pairwise constraints. The proposal presents also a new developed score that combines the power of the local geometric structure offered by unlabeled data, with the constraint preserving ability offered by labeled data. In the following, we present an illustration about constraint selection, and the measure of constraint utility which we adopt in this approach.

3.4.1 Constraint selection

While it was expected that different constraints sets would contribute more or less to improving accuracy of many constrained algorithms, it was found that some constraint sets actually decrease the performance. It was observed that constraints can have ill effects even when they are generated from the data labels that are used to evaluate accuracy, so this behavior is not caused by noise or errors in the constraints. Instead, it is a result of the interaction between a given set of constraints and the algorithm being used. So it is more important to know why do some constraint sets increase clustering accuracy while others have no effect or even decrease accuracy. For that, the authors in [Davidson et al., 2006] have defined two important measures, informativeness and coherence, that capture relevant properties of constraint sets. These measures provide insight into the effect a given constraint set has on a specific constrained clustering algorithm. The informativeness measure refers to the amount of information in the constraint set that the algorithm cannot determine on its own. In order to calculate this measure, the learning algorithm is run without constraints. Then, the results are checked to measure how much constraints are satisfied. If all the constraints are satisfied (note that they were not used in learning), then the constraint set has no informativeness towards the learning algorithm in hand. However, if many constraints are not satisfied, the constraint set is said to be very informative. Note that this measure is dependent to the learning algorithm. In this framework, we opt using the coherence measure only, since it is independent of any algorithm that could be used for learning, which is specific to our paradigm dealing with a filter approach.

Coherence represents the amount of agreement between the constraints themselves, given a metric d that specifies the distance between points. One view of an ML (or CL) constraint is that it imposes an attractive (or repulsive) force within the feature space along the direction of a line formed by a pair of instances (x_1, x_2), within the vicinity of x_1 and x_2. Two constraints, one an ML constraint (ct_1) and the other a CL constraint (ct_2), are incoherent if they exert contradictory forces in the same vicinity. Two constraints are perfectly coherent if they are orthogonal to each other. To determine the coherence of two

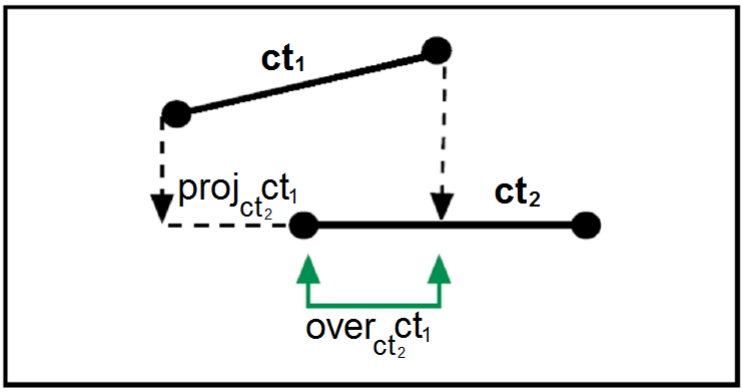


Figure 3.3: Projected overlap between a $ML(ct_1)$ and $CL(ct_2)$ constraints, $(over_{ct_2} ct_1)$ is not null. So, The coherence of the subset $\{ct_1, ct_2\}$ is null.

constraints, ct_1 and ct_2, we compute the projected overlap of each constraint on the other as follows.

The coherence of a given constraint set Ω is defined as a fraction of constraint pairs that have zero projected overlap (Figure 3.3):

$$Coh_d(\Omega) = \frac{\sum_{ct_p \in \Omega_{ML}, ct_q \in \Omega_{CL}} \delta(over_{ct_q} ct_p = 0 \wedge over_{ct_p} ct_q = 0)}{|\Omega_{ML}|\,|\Omega_{CL}|} \tag{3.15}$$

where $over_{ct_q} ct_p$ represents the distance between the two projected points linked by ct_p over ct_q. δ is the number of the overlapped projections.

From the equation (3.15), we can easily define a specific measure for each constraint as follows:

$$Coh_d(ct_p) = \frac{\sum_{ct_q \in \Omega_{CL}} \delta(over_{ct_q} ct_p = 0)}{|\Omega_{CL}|} \tag{3.16}$$

$$Coh_d(ct_q) = \frac{\sum_{ct_p \in \Omega_{ML}} \delta(over_{ct_p} ct_q = 0)}{|\Omega_{ML}|} \tag{3.17}$$

We now show how to select the relevant constraints according to their coherence.

Algorithm 9 Constraint selection

Input: Constraint set $\Omega = \{ct_i | i = 1..(l(l-1)/2)\}$
Onput: Selected constraint set $\Omega_s = \Omega'_{ML} \cup \Omega'_{CL}$

```
Initialize Ω_s = ∅
for i = 1 to |Ω| do
    if Coh_d(ct_i) ≥ Coh_d(Ω) then
        Ω_s = Ω_s ∪ {ct_i}
    end if
end for
```

To be selected, a constraint ct_i must have a coherence $coh_d(ct_i)$ greater than the global coherence of all constraints in (Ω), i.e. it must have a minimum overlap with the other constraints ct_j ($ct_j \in \Omega_{CL}$ if $ct_i \in \Omega_{ML}$ and vice-versa (Algorithm 9)).

From this algorithm we obtain Ω_s, which is a set of coherent constraints of Ω_{ML} and Ω_{CL} in two subsets Ω'_{ML} and Ω'_{CL} respectively. The algorithm tests the coherence of each constraint with all other constraints regardless of the order. Then it supplies the same results (coherent constraints) for the same input (constraint set). The complexity of this algorithm is linear to the number of all a priori constraints in Ω: $O(l(l-1)/2)$.

3.4.2 Feature relevance

We have seen that the main advantage of CLS is its survey of the respect of data structure and locality preserving ability. In addition, it exploits background information which adds a constraint preserving ability. However, In CSFS we propose an improvement of the score function which evaluates the feature relevance. In fact, we propose a more efficient semi-supervised feature selection. To do so, we define a new function score (φ), which should be minimized, as follows:

$$\varphi_r = \frac{\sum_{i,j} (f_{ri} - f_{rj})^2 (\boldsymbol{S}_{ij} + \boldsymbol{N}_{ij})}{\sum_i \left(f_{ri} - \alpha^i_{rj}\right)^2 \boldsymbol{D}_{ii}} \tag{3.18}$$

Where:

$$\boldsymbol{S}_{ij} = \begin{cases} e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}} & \text{if } x_i \text{ and } x_j \text{ are neighbors} \\ 0 & \text{otherwise} \end{cases} \tag{3.19}$$

And:

$$\boldsymbol{N}_{ij} = \begin{cases} -e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}} & \text{if } x_i \text{ and } x_j \text{ are neighbors and } (x_i, x_j) \in \Omega'_{ML} \\ \left(e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}}\right)^2 & \text{if } x_i \text{ and } x_j \text{ are neighbors and } (x_i, x_j) \in \Omega'_{CL} \\ & \text{OR} \\ & \text{if } x_i \text{ and } x_j \text{ are not neighbors and } (x_i, x_j) \in \Omega'_{ML} \\ 0 & \text{otherwise} \end{cases} \tag{3.20}$$

And:

$$\alpha_{rj}^i = \begin{cases} f_{rj} & \text{if } (x_i,\ x_j) \in \Omega'_{CL} \\ \mu_r & \text{otherwise} \end{cases} \tag{3.21}$$

Where λ is a constant to be tuned, and x_i, x_j are neighbors means that x_i is among the k-nearest neighbors of x_j and $\mu_r = \frac{1}{n}\sum_i f_{ri}$ is the mean of the column *r*.

Note that if there are no labels ($l = 0$ and $X = X_U$) then φ_r becomes a Laplacian score and when ($u = 0$ and $X = X_L$), φ_r represents an adjusted constraint score, where the *ML* and *CL* information would be weighted by $(\boldsymbol{S}_{ij} + \boldsymbol{N}_{ij})$ and $\boldsymbol{D}_{ii}$ respectively in the formula.

With CSFS, on the one hand, a relevant feature should be the one on which those two instances (neighbors or related by an *ML* constraint) are close to

each other. On the other hand, the relevant feature should be the one with a larger variance or on which those two instances (related by a *CL* constraint) are well separated.

To assess the previous concept, we use a weight $\boldsymbol{N}_{ij}$. The motivation of adding $\boldsymbol{N}_{ij}$ to our score (over the Laplacian score) is not only the integration of pairwise constraints into the score, but also adding a sensibility dimension to the feature score in the following cases:

When we have two instances related by a *ML* constraint but not neighbors $(\boldsymbol{S}_{ij} + \boldsymbol{N}_{ij}) = \left(e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}}\right)^2$ and when two neighboring instances are related by a *CL* constraint $(\boldsymbol{S}_{ij} + \boldsymbol{N}_{ij}) = \left(e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}}\right)^2 + e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}}$. In both cases, the weight $\left(e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}}\right)^2$ is used in order to more differentiate the features in the both *bad cases*.

3.4.3 Spectral graph analysis

In this section we give a spectral graph-based explanation for the described function score. A reasonable criterion for choosing a relevant feature is to minimize the objective function represented by φ. Thus, the problem is to minimize the first term $T_1 = \sum_{i,j}(f_{ri} - f_{rj})^2(\boldsymbol{S}_{ij} + \boldsymbol{N}_{ij})$ and maximize the second one $T_2 = \sum_{i,j}(f_{ri} - \alpha^i_{rj})^2 \boldsymbol{D}_{ii}$. By resolving these two optimization problems, we prefer those features respecting their pre-defined graphs, respectively. Thus, we construct a k-neighborhood graph G_{kn} from X (data set) and Ω'_{ML} (Selected ML constraint set) and a second graph G_{CL} from Ω'_{CL} (Selected CL constraint set).

Given a data set X, let $G(V, E)$ be the complete undirected graph constructed from X, with V is its node set and E is its edge set. The i^{th} node v_i of G corresponds to $x_i \in X$ and there is an edge between each nodes pair (v_i, v_j), whose weight $w_{ij} = e^{-\frac{\left\|x_i - x_j\right\|^2}{\lambda}}$ is the dissimilarity between x_i and x_j.

$G_{kn}(V, E_{kn})$ is a subgraph which could be constructed from G where E_{kn} is the edge set $\{e_{i,j}\}$ from E such that $e_{i,j} \in E_{kn}$ if $(x_i, x_j) \in \Omega'_{ML}$ or x_i is one of the k-nearest neighbors of x_j. $G_{CL}(V_{CL}, E_{CL})$ is a subgraph constructed from G with V_{CL} its node set and $\{e_{i,j}\}$ its edge set such that $e_{i,j} \in E_{CL}$ if $(x_i, x_j) \in \Omega'_{CL}$.

Once the graphs G_{kn} and G_{CL} are constructed, their weight matrices, denoted by $(\boldsymbol{S}^{kn} + \boldsymbol{N}^{kn})$ and $\boldsymbol{S}^{CL}$ respectively, can be defined as:

$$\boldsymbol{S}_{ij}^{kn} = \begin{cases} w_{ij} & \text{if } x_i \text{ and } x_j \text{ are neighbors} \\ 0 & \text{otherwise} \end{cases} \tag{3.22}$$

and

$$\boldsymbol{N}_{ij}^{kn} = \begin{cases} -w_{ij} & \text{if } x_i \text{ and } x_j \text{ are neighbors and } (x_i, x_j) \in \Omega'_{ML} \\ w_{ij}^2 & \text{if } x_i \text{ and } x_j \text{ are neighbors and } (x_i, x_j) \in \Omega'_{CL} \text{ or} \\ & \text{if } x_i \text{ and } x_j \text{ are not neighbors and } (x_i, x_j) \in \Omega'_{ML} \\ 0 & \text{otherwise} \end{cases} \tag{3.23}$$

and

$$\boldsymbol{S}_{ij}^{CL} = \begin{cases} 1 & if (x_i, x_j) \in \Omega'_{CL} \\ 0 & \text{otherwise} \end{cases} \tag{3.24}$$

Then, we can define:

- For each feature F_r, its vector $f_r = (f_{r1}, ..., f_{rn})^T$.
- Diagonal matrices $\boldsymbol{D}_{ii}^{kn} = \sum_j \boldsymbol{S}_{ij}^{kn}$, $\boldsymbol{DN}_{ii}^{kn} = \sum_j \boldsymbol{N}_{ij}^{kn}$ and $\boldsymbol{D}_{ii}^{CL} = \sum_j \boldsymbol{S}_{ij}^{CL}$.
- Laplacian matrices $\boldsymbol{L}^{kn} = (\boldsymbol{D}^{kn} + \boldsymbol{DN}^{kn}) - (\boldsymbol{S}^{kn} + \boldsymbol{N}^{kn})$ and $\boldsymbol{L}^{CL} = \boldsymbol{D}^{CL} - \boldsymbol{S}^{CL}$.

We can easily develop the first term of φ as follows:

$$T_1 = \sum_{i,j} (f_{ri} - f_{rj})^2 (\boldsymbol{S}_{ij}^{kn} + \boldsymbol{N}_{ij}^{kn}) \tag{3.25}$$

$$
\begin{aligned}
&= \sum_{i,j}(f_{ri}^2 + f_{rj}^2 - 2f_{ri}f_{rj})(\boldsymbol{S}_{ij}^{kn} + \boldsymbol{N}_{ij}^{kn}) \\
&= 2(\sum_{i,j} f_{ri}^2(\boldsymbol{S}_{ij}^{kn} + \boldsymbol{N}_{ij}^{kn}) - \sum_{i,j} f_{ri}(\boldsymbol{S}_{ij}^{kn} + \boldsymbol{N}_{ij}^{kn})f_{rj}) \\
&= 2(f_r^T(\boldsymbol{D}^{kn} + \boldsymbol{DN}^{kn})f_r - f_r^T(\boldsymbol{S}^{kn} + \boldsymbol{N}^{kn})f_r) \\
&= 2f_r^T\boldsymbol{L}^{kn}f_r
\end{aligned} \tag{3.26}
$$

Note that satisfying the graph structures is done according to α_{rj}^i in eq.(3.3). In fact, when $\Omega'_{CL} = \varnothing$, we should maximize the variance of f_r which would be estimated as:

$$Var(f_r) = \sum_i (f_{ri} - \mu_r)^2 \boldsymbol{D}_{ii}^{kn} \tag{3.27}$$

The optimization of eq.(3.27) can be done as in section (3.3.1). In this case, $\varphi_r = LS_r = \frac{f_r^T\boldsymbol{L}^{kn}f_r}{f_r^T\boldsymbol{D}^{kn}f_r}$. Otherwise, we develop as above the second term (T_2) and obtain $2f_r^T\boldsymbol{L}^{CL}\boldsymbol{D}^{kn}f_r$.

Subsequently,

$$\varphi_r = \frac{f_r^T\boldsymbol{L}^{kn}f_r}{f_r^T\boldsymbol{L}^{CL}\boldsymbol{D}^{kn}f_r} \tag{3.28}$$

seeks those features that respect G_{kn} and G_{CL}. The complete algorithm of CSFS is summarized in Algorithm 10.

The step 3 of the Algorithm 10 is computed in time $O(mn^2)$. Note that -as

Algorithm 10 CSFS

Input: Dataset $X(n \times m)$, the constant λ
Output: Ranked features

1: Construct the constraint set (Ω_{ML} and Ω_{CL}) from Y_L
2: Select the coherent set (Ω'_{ML} and Ω'_{CL}) from (Ω_{ML} and Ω_{CL}) using Algorithm.9
3: Construct the graphs G_{kn} and G_{CL} from (X, Ω'_{ML}) and Ω'_{CL} respectively.
4: Calculate the weight matrices $\boldsymbol{S}^{kn}$, $\boldsymbol{N}^{kn}$ and $\boldsymbol{S}^{CL}$ and the Laplacians $\boldsymbol{L}^{kn}$, $\boldsymbol{L}^{CL}$.
5: **for** $r = 1$ **to** m **do**
6: Calculate φ_r according to eq.(3.28)
7: **end for**
8: Rank the features F_r according to their scores φ_r in ascending order.

in CLS- the *small labeled-sample problem* becomes an advantage for CSFS complexity, because it supposes that the number of extracted constraints is smaller since it depends on the number of labels l. Thus, the cost of the algorithm depends considerably on the size of unlabeled data X_U.

To reduce such complexity, we propose to apply a clustering on X_U (with u vectors). We apply the same SOM clustering which we applied in CLS score. Note that SOM algorithm is used in order to group and code the unlabeled data and not to select them. Note also that, by clustering X_U the complexity of step 3 in Algorithm 10 is reduced to (mu).

Subsequently, SOM will be applied on the unsupervised part of data (X_U) for obtaining (X'_U) with a size equal to the number of SOM' nodes (K). Therefore, φ will be performed on the new obtained dataset $(X_L \cup X'_U)$.

3.4.4 Adaptive *k*-neighborhood graph

Among the advantages of φ score is the assessment of locality preserving ability by features. Meanwhile, the principle of fixed k-nearest neighbors for all instances may affect the locality preserving, because there is no certainty that the k-nearest neighbors of an instance are "close" to it (Figure 3.4-a).

In this case, some "far" neighbors would be enrolled in the locality preserving measurement for the example in hand. Hence, we advise using a similarity based clustering approach to all the instances as it reveals their locality

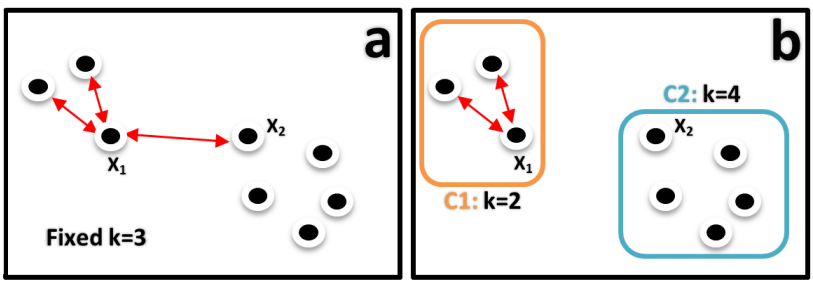


Figure 3.4: (a) Fixed *k*-nearest neighborhood. (b) Adaptive *k*-nearest neighborhood

structures. The k-nearest neighborhood relationship between them will then depend on the membership to the same cluster. Hence, the adaptive k would be related to data structure and could be defined as follows: two instances are neighbors if they belong to the same cluster. Consequently, each cluster has its own k which is the number of its elements (less one).

In Figure 3.4-b, calculating the score of x_1 does not need to look far, but is calculated on the base of the instances belonging to its cluster. Accordingly, the score is less biased and the locality is more preserved. In addition, a main advantage of having such adaptive neighborhood, is reducing the number of parameters of the feature selection algorithm.

To conclude, CSFS has three major advantages:

1. It incorporates the labeled and unlabeled instances in a competent and flexible manner, so it can be utilized regardless of the percentage of the labeled data.

2. It exploits a pairwise constraint selection, which results in a coherent constraint subset extracted from the labeled data.

3. It surveys the structural neighborhood of data examples, which highlights the efficient locality preserving properties of the selected features.

3.5 Redundancy analysis in selected features (CSFSR)

In this section we propose an extension to the original CSFS algorithm in order to eliminate the redundancy in the selected features. We propose CSFSR for semi-supervised feature selection with redundancy elimination.

Feature redundancy is naturally *correlated* to feature correlation. It is widely accepted that two features are redundant to each other if their values are completely correlated [Yu and Liu, 2004]. In this section, we will first introduce

our choice of correlation measure, then we will describe our strategy to reduce the redundancy of relevant features.

3.5.1 Correlation measures

The most known measure that can be used to calculate the relationship between two features F_r and F_c is linear correlation coefficient. It is defined as follows:

$$\rho(F_r, F_c) = \frac{\sum_i (f_{ri} - \overline{f_r})(f_{ci} - \overline{f_c})}{\sqrt{\sum_i (f_{ri} - \overline{f_r})^2}\sqrt{\sum_i (f_{ci} - \overline{f_c})^2}} \tag{3.29}$$

where $\overline{f_r}$ and $\overline{f_c}$ are the means of the feature vectors f_r and f_c respectively.

However, linear correlation is not always adapted to real-world applications. For that, other non-linear measures are better adapted[1]. We choose to define the mutual information between two features F_r and F_c in terms of their probabilistic density functions $p(F_r), p(F_c), p(F_r, F_c)$:

$$I(F_r, F_c) = \int\int p(F_r, F_c) log \frac{p(F_r, F_c)}{p(F_r)p(F_c)} dF_r dF_c \tag{3.30}$$

Mutual information quantifies the dependence between the joint distribution of F_r and F_c and what the joint distribution would be if F_r and F_c were independent. Mutual information is a measure of dependence in the following sense: $I(F_r, F_c) = 0$ if and only if F_r and F_c are independent random variables. This is easy to see in one direction: if F_r and F_c are independent, then $p(F_r, F_c) = p(F_r)p(F_c)$, and therefore $log(\frac{p(F_r,F_c)}{p(F_r)p(F_c)}) = 0$.

Under the hypothesis that the joint distribution of F_r and F_c is multi-variate distribution, the mutual information can be directly related to the correlation coefficient ρ [Kullback, 1959]:

$$I(F_r, F_c) = -\frac{1}{2} log(1 - \rho^2(F_r, F_c)) \tag{3.31}$$

[1] In [Yu and Liu, 2004], the authors listed another correlation measure, the entropy, which is a measure of the uncertainty of a random variable. However, the entropy is adapted to categorical data.

3.5.2 Maximum spanning tree based redundancy elimination

In this section, we show how to automatically detect the subset of features which have a strong multiple correlation in a set of relevant features. We propose a strategy based on maximum spanning tree to eliminate the maximum number of redundant features and keep the strong relevant ones.

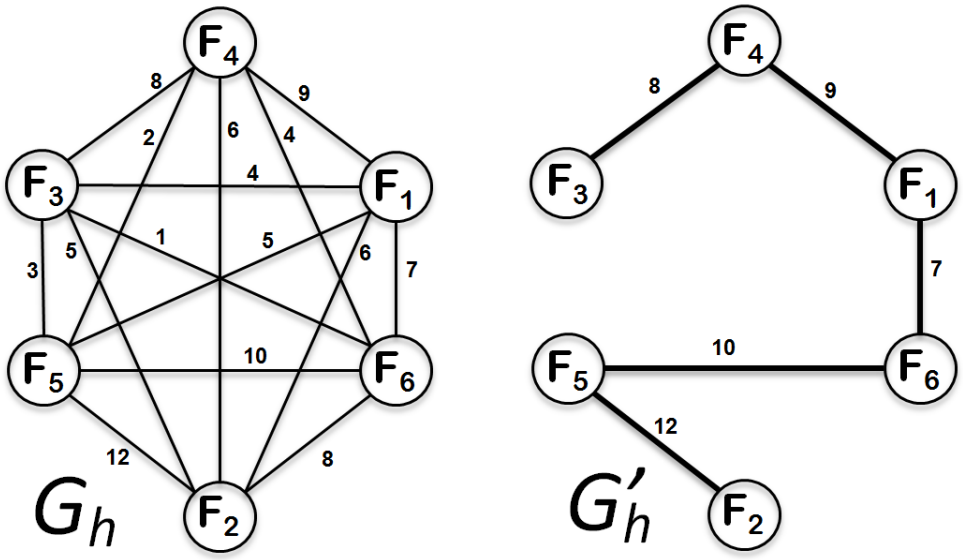


Figure 3.5: G_h: Original Graph; G'_h:Maximum spanning tree. Here $h = 6$ features.

This technique requires a matrix of weights between vertices (features in our case). Thus, we calculate a matrix of correlations based on mutual information according to eq.(3.31). This matrix is of dimension: $h \times h$ such as h is the number of the best first features ranked according to Algorithm 10.

Algorithm 11 Prim

Input: The graph of relevant features: $G_h(V_h, E_h)$
Onput: The maximum spanning tree $G'_h(V'_h, E'_h)$

1: Initialize: $V'_h = \{F^*\}$ where F^* is the most relevant feature in V_h
   $E'_h = \varnothing$
2: Repeat
   Choose an edge (F_i, F_j) with maximum weight such that $F_i \in V'_h$ and $F_j \in \overline{V_h}$
   $V'_h = V'_h \cup \{F_j\}$
   $E'_h = E'_h \cup \{(F_i, F_j)\}$
3: Until $V'_h = V_h$

Algorithm 12 CSFSR

Input: Dataset $X(n \times m)$, the constant λ
Output: Selected features

1: Select the h best relevant features ranked according to Algorithm 10
2: Construct the graph $G_h(V_h, E_h)$
3: Find the maximum spanning tree $G'_h(V'_h, E'_h)$ from G_h using Algorithm 11
4: **repeat**
5: Select a relevant feature F_r from V'_h (in the order of step 1)
6: Remove all features F_j from V'_h such as $(F_r, F_j) \in E'_h$
7: **until** until no more relevant feature can be selected in V'_h

Let $G_h(V_h, E_h)$ be a complete weighted graph, where V_h is the set of the h relevant features (vertices) and E_h is the set of edges weighted according to eq.(3.31).

A maximum spanning tree G'_h is a connected and acyclic sub-graph of G_h, for which the sum of edge weights is maximum (Figure. 3.5).

On the left side of Figure 3.5, G_h represents a complete graph where all the edges are weighted using the mutual information values. On the right, we can see the maximum spanning tree G'_h obtained from G_h where the solid edges represent the tree providing the highest multiple correlation in the considered set of relevant features.

For constructing G'_h we use the optimized algorithm of Prim [Cormen et al., 2001]:

Let be V'_h and E'_h two empty sets. First, we affect to V'_h the most relevant feature

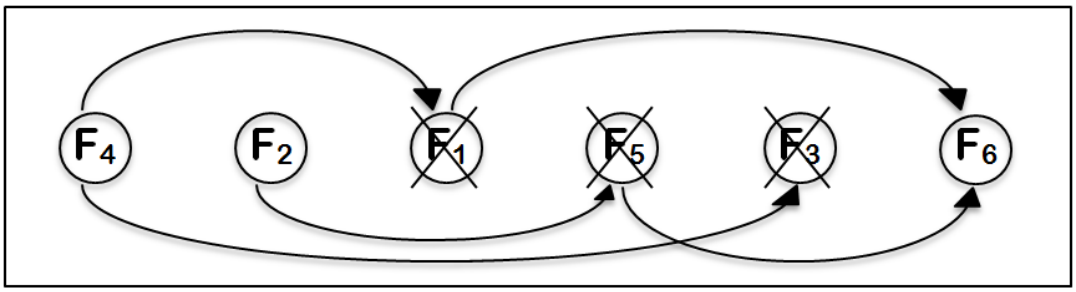


Figure 3.6: Selection of relevant and non redundant features.

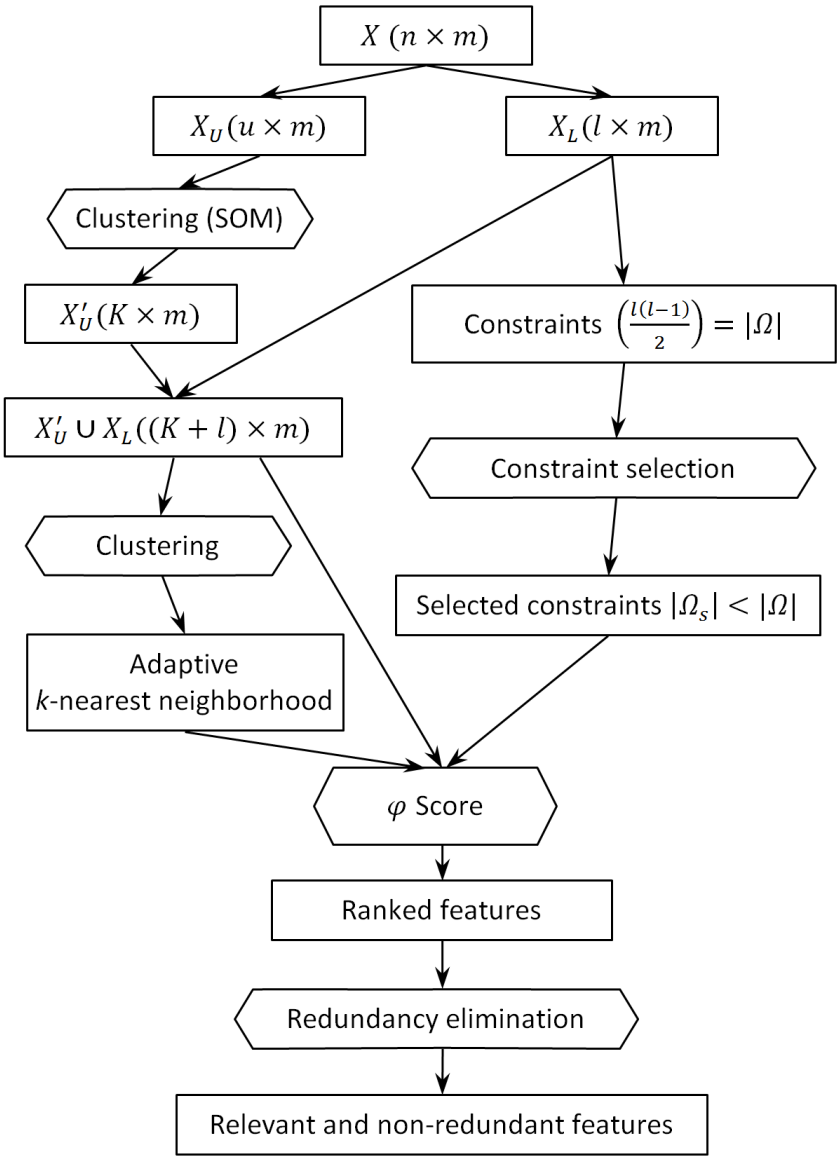


Figure 3.7: Feature selection framework of CSFSR

from V_h (i.e. the feature that has the minimum score). The goal is to find the edge $(F_i, F_j) \in V'_h \times \overline{V_h}$ having the maximum weight ($\overline{V_h} = (V_h - V'_h)$) and to put F_j in V'_h and (F_i, F_j) in E'_h. This procedure is repeated for $(h-1)$ iterations.

A simple implementation using an adjacency matrix graph representation and searching an array of weights to find the minimum weight edge to add requires $O(h^2)$ running time. Using a simple binary heap data structure and an adjacency list representation, Prim's algorithm can be shown to run in time

$O(|E_h|Log|V_h|)$. Using a more sophisticated Fibonacci heap, this can be brought down to $O(|E_h|+|V_h|Log|V_h|) = O(\frac{h(h-1)}{2} + h\,Log\,h)$ [Cormen et al., 2001], which is asymptotically faster when the graph is dense enough (the case of G_h).

Our strategy for redundancy elimination consists of (1) selecting the h first relevant features ranked according to Algorithm.10, (2) constructing a weighted graph between these relevant features using the eq.(3.31), (3) finding the maximum spanning tree according to Algorithm.11, (4) selecting a relevant feature (in the order of (1), (5) removing all features with which it has an edge in (3) , and (6) iterating steps (4) and (5) until no more relevant (and non redundant) feature can be selected (Algorithm 12).

As an example, we present in Figure 3.6 six features selected as relevant ones and ranked according to their φ values. We show how to eliminate redundant relevant features with the help of the maximum spanning tree obtained in Figure 3.5. F_4 is the most relevant feature. In the first round, F_4 is selected and F_1 and F_3 are removed based on F_4. In the second round, F_2 is selected and F_5 is removed based on F_2. In the last round F_6 is selected. Finally, we summarize our feature selection framework in Figure 3.7.

3.6 Experimental results

In this section, we present the empirical results of our proposals, and compare them with a variety of representative methods for dimensionality reduction. Furthermore, we keep the same configurations for each parameter used in the compared methods.

At first, we start by the results of CLS over high-dimensional datasets, downloaded from well-known repositories.

3.6.1 Datasets and methods

We present an empirical study on several datasets. From UCI: "Iris", "Wave", "Ionosphere", "Sonar" and "Soybean"; Microarray datasets: "Leukemia" and

"Colon cancer"; and Face-image datasets: "PIE10P" and "PIX10P". The whole datasets information is detailed in Table 3.1.

These datasets are voluntarily chosen for evaluating the learning performance of our proposal, CLS, and comparing it with other techniques that were experimented over them. The concerned methods are summarized and listed below:

- Variance score, is only based on variance for feature selection [Bishop, 1995].
- Fisher score, is based on variance and all labels for feature selection [Duda et al., 2000].
- ReliefF, estimates the significance of features according to how well their values distinguish between the instances of the same and different labels that are near to each other [Robnik-Šikonja and Kononenko, 2003].
- F2+r[4] and F3+r (SPEC), are spectral feature selection methods [Zhao and Liu, 2007b].
- Laplacian score [He et al., 2005] (described in Section 2.7.1).
- Constraint score [Zhang et al., 2008] (described in Section 2.7.4).
- SC4 [Kalakech et al., 2011] (described in Section 2.7.6).

The experimental results are presented on three folds. First, we test CLS algorithm on datasets whose the relevant features are known. Second, we do some comparisons with known powerful feature selection methods and finally, we apply the algorithm on databases with huge number of features. In most experiments, the λ value is set to 0.1 and $k = 10$ for building the neighborhood graph. These parameters are initialized with the same values as the other competitive methods. For the semi-supervised data, we choose the first labeled examples for all datasets (with different labels). We do no selection neither on the level of examples to be labeled, nor on the generated constraints.

Table 3.1: Datasets

Datasets	n	m	K	Source
Iris	150	4	3	[Frank and Asuncion, 2010]
Wave	5000	40	3	[Frank and Asuncion, 2010]
Ionosphere	351	34	2	[Frank and Asuncion, 2010]
Sonar	208	60	2	[Frank and Asuncion, 2010]
Soybean	47	35	4	[Frank and Asuncion, 2010]
Leukemia	72	7129	2	[Golub et al., 1999]
Colon cancer	62	2000	2	[Alon et al., 1999]
PIE10P	210	2420	10	[Zhao et al., 2011]
PIX10P	100	10000	10	[Zhao et al., 2011]

For the construction of the SOM' maps in the phase of unlabeled data clustering (Algorithm 8), we use the Principal Component Analysis (PCA) based heuristic proposed by Kohonen [Kohonen, 2001] to automatically provide the number of the initial numbers of clusters and the dimensions of the maps[2]. The reference vectors are initialized linearly along the greatest eigenvectors of the associated data X_U.

3.6.2 Validation of feature selection

In this section, we are particularly interested on the two first datasets ("Iris" and "Wave") which are popularly used in machine learning and data mining tasks. In fact, we present the results of our approaches over these two datasets as a starting validation point, this is because we have the a priori information about the noise and the relevant features in both datasets.

In "Iris", one class is linearly separable from the other two which are not linearly separable from each other. Out of the four features it is known that the features F_3 (petal length) and F_4 (petal width) are more important for the underlying clusters than F_1 (sepal length) and F_2 (sepal width) Figure 3.8. The sub-figure (c) shows the data projected on the subspace constructed by F_3 and F_4, whereas the sub-figure (b) shows the data projected on the subspace of F_1 and F_2. In

[2]All experiments are performed on MATLAB. The SOM toolbox is used and can be found at (http://www.cis.hut.fi/somtoolbox/).

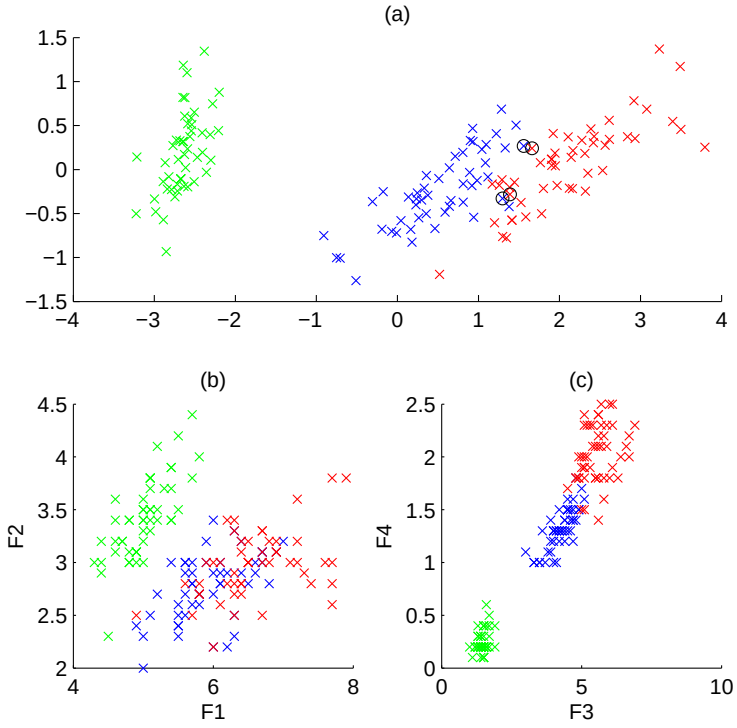


Figure 3.8: 2D-Visualization of "Iris".

[He et al., 2005], it was reported that by using variance score [Bishop, 1995], the four features are sorted as (F_3, F_1, F_4, F_2). With $k \geq 15$, Laplacian score sorts these four features as (F_3, F_4, F_1, F_2). It sorts them as (F_4, F_3, F_1, F_2) when $3 \leq k < 15$. By using CLS, the features are sorted as (F_3, F_4, F_1, F_2) for any value of k (between 1 and 20). For explaining the difference between the two scores, we chose for this dataset, $l = 10$ generating 45 constraints. Two of CL-type constraints are constructed from the pairs $(73^{th}, 150^{th})$ and $(78^{th}, 111^{th})$ according to the labels of the points Figure.3.8(a)[3] (The concerned points are represented by rounds). Since, the data points between brackets are close, with the Laplacian score, the edges $e_{73,150}$ and $e_{78,111}$ are constructed in the associated k-neighborhood graph and affect the feature selection process. With our method, these edges never exist because of the CL constraint property even if k is small. For that, the scores obtained by CLS are smaller than the ones obtained by Laplacian score. We also observed an important gap on scores

[3]Figure.3.8(a) is obtained by PCA (Principal Component Analysis).

between the relevant variables ($CLS_3 = 1.4 \times 10^{-3}$, $CLS_4 = 2.7 \times 10^{-3}$) and the irrelevant ones ($CLS_1 = 1.07 \times 10^{-2}$, $CLS_2 = 1.77 \times 10^{-2}$). In fact, In the region where the points belong to the two non-linearly separable classes, Laplacian score is biased by the dissimilarity which could affect the ranking of features for their selection, while CLS is able to control this problem with the help of constraints.

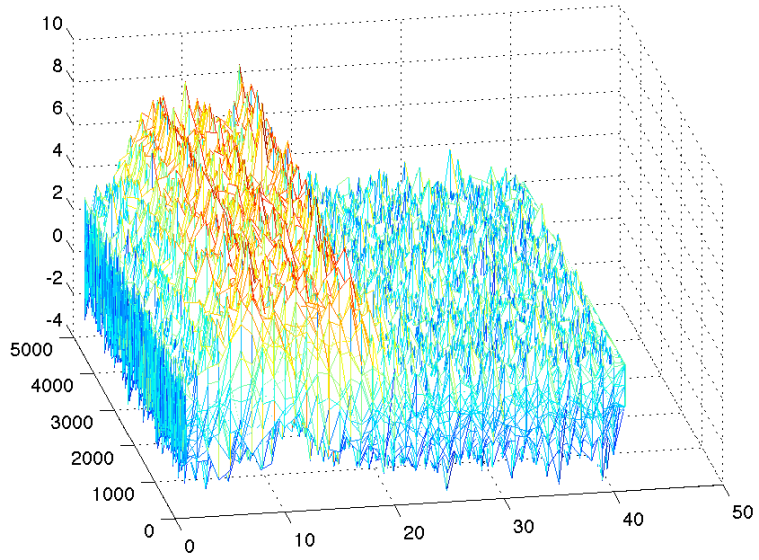


Figure 3.9: "Wave" dataset.

The waveform of Breiman dataset "Wave" consists of 5000 instances divided into 3 classes. This dataset is composed of 21 relevant features (the first ones) and 19 noise features with mean 0 and variance 1. Each class is generated from a combination of 2/3 "base" waves (Figure 3.9).

We tested our feature selection algorithm with $l = 8$ (28 constraints) and the dimension of the map (26×14) for SOM' algorithm. We can see in Figure 3.10 that the features (21 to 40) have high values on CLS. The noise represented by these features is then clearly detected.

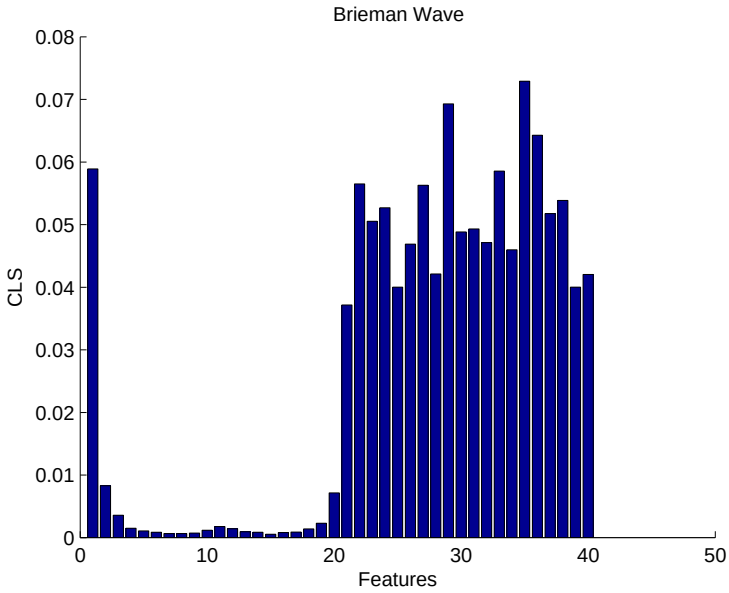


Figure 3.10: Results of CLS on features of "Wave" dataset.

3.6.3 Comparison of the feature selection quality

In order to compare CLS approach with other methods, the nearest neighborhood (1-NN) classifier[4] with Euclidean distance, is employed for classification after feature selection. For each dataset, the classifier is learned in the first half of instances from each class and tested on the remaining data. We tested the accuracy behavior of the ranking feature function represented by CLS for comparing it with those of other methods cited in [Zhang et al., 2008]. These experiments were applied on three datasets [Frank and Asuncion, 2010], the first one is "Ionosphere" which represents radar returns from the Ionosphere. In addition, we use "Sonar" dataset which contains patterns obtained by bouncing sonar signals off a metal cylinder at various angles and under various conditions. The third dataset is "Soybean" which represents Michalski's famous soybean disease database. In order to create a semi-supervised form of these datasets, we keep randomly 5 labeled instances for each one (all classes are represented),so 10 pairwise constraints were generated.

[4] Other classifiers can be exploited like (Decision Tree, SVM, etc.) which will be utilized later in this thesis.

Figure 3.11 indicates that, in most cases, the performance of CLS is comparable to Fisher Score [Duda et al., 2000] and significantly better than that of Variance, Laplacian and Constraint scores. This verifies that merging supervision information of labeled data with geometrical structure of unlabeled data is very useful in learning feature scores. Table 3.2 compares the averaged accuracy under different number of selected features. Here, the values after the symbol $\pm$ denote the standard deviation. From Table 3.2 and Figure 3.11 we can find that, the performance of CLS is almostly better than that of Variance, Laplacian score and Constraint score and is comparable with Fisher Score. More specifically, CLS is superior to Fisher Score on "Soybean" and "Ionosphere" and is inferior on "Sonar". Note that Fisher score uses all labels when CLS score uses just 5 labels for each dataset.

Table 3.2: Averaged accuracy of different algorithms on "Ionosphere", "Sonar" and "Soybean"

Datasets	Variance	Laplacian	Fisher	CS	CLS
Ionosphere	82.2±3.8	82.6±3.6	86.3±2.5	85.1±2.9	**86.73±2.1**
Sonar	79.3±6.3	79.5±7.2	**86.4±6.9**	80.7±7.8	83.3±1.7
Soybean	88.9±12.7	79.4±28.4	94.5±12.1	93.5±11.6	**95.06±1.3**

Then, we compare the performance of CLS with that of Fisher and constraint scores when different levels of supervision are used. Figure 3.12 shows the plots for accuracy under desired number of selected features vs. different numbers of labeled data (for Fisher Score) or pairwise constraints (for CS and CLS) on the three datasets ("Ionosphere", "Sonar" and "Soybean"). Here, the desired number of selected features is chosen as half of the original dimension of instances. For all scores, the results are averaged over 100 runs. As shown in Figure 3.12, except on "Sonar", CLS is much better than the other two algorithms especially when only a few labeled data or constraints are used. On "Sonar", both CS and CLS are inferior to Fisher Score when the number of labeled data (or constraints) is great; CLS is always better when this number is small. A closer study on Figure 3.12 reveals that, generally, the accuracy of CLS increases steadily and fast in the beginning (with few constraints) and slows down at the end (with relatively more constraints). It implies that too many constraints won't help too much to further boost the accuracy, and only a few constraints are

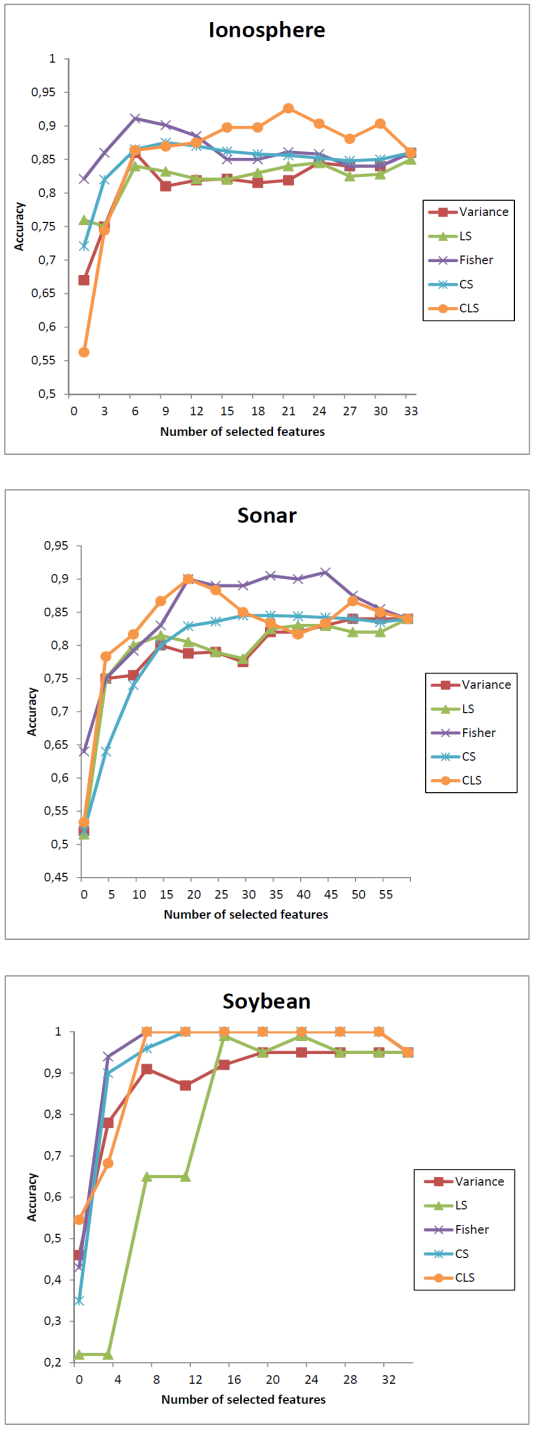
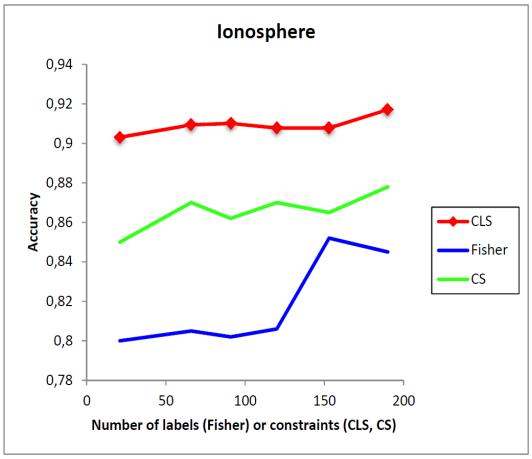
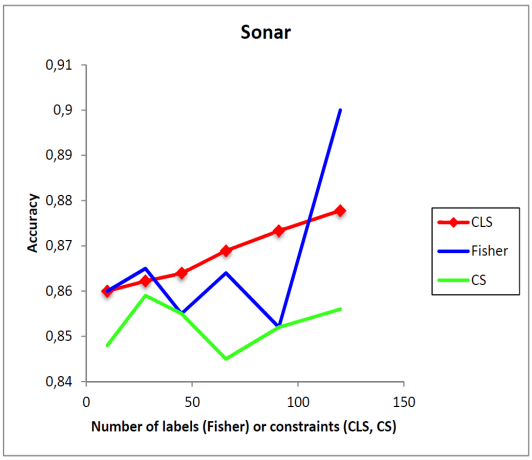


Figure 3.11: Accuracy vs different numbers of selected features.

required in CLS, which corresponds exactly to our initial problem concerning "small labeled-sample" data. While Fisher Score typically requires relatively more labeled data to obtain a satisfying accuracy.





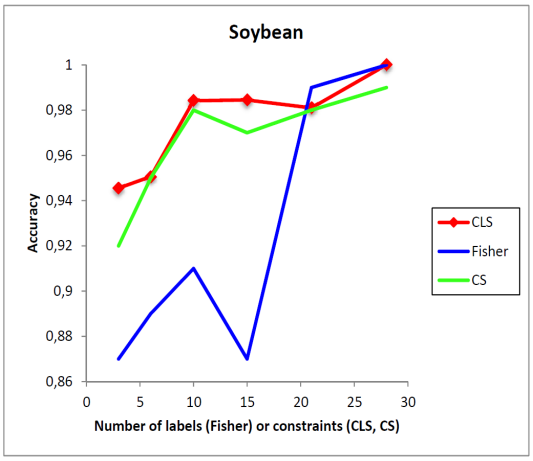


Figure 3.12: Accuracy vs. different numbers of labeled data (for Fisher Score) or pairwise constraints (for CScore and CLS).

3.6.4 Results on gene expression datasets

"Leukemia" and "Colon cancer" are gene expression databases with huge number of features. The microarray Leukemia data is constituted of a set of 72 individuals, corresponding to two types of Leukemia called ALL (Acute Lymphocytic Leukemia) and AML (Acute Myelogenous Leukemia), with 47 ALL and 25 AML. The dataset contains expressions for 7129 genes. While "Colon cancer" is a dataset of 2000 genes measured on 62 tissues (40 tumors and 22 "normal").

We present our results on these datasets in comparison with Laplacian, Fisher, SC4 and CS scores, and that in case of Accuracy vs. Selected features. The results (Figure 3.13) show that CLS records a comparable performance with other scores when the number of features is inferior to 2500 for "Leukemia" dataset, and 500 for "Colon cancer" dataset, then the performance of CLS is superior to other scores performance when increasing the number of features.

3.6.5 Results on face-image datasets

"PIE10P" and "PIX10P" are face-image datasets, each contains 10 persons. The validation on these datasets is presented in comparison with Laplacian, ReliefF scores on both datasets. In addition, results were compared with (F2+r^4) score on "PIX10P" dataset and with (F3+r) score on "PIE10P" dataset. We chose to

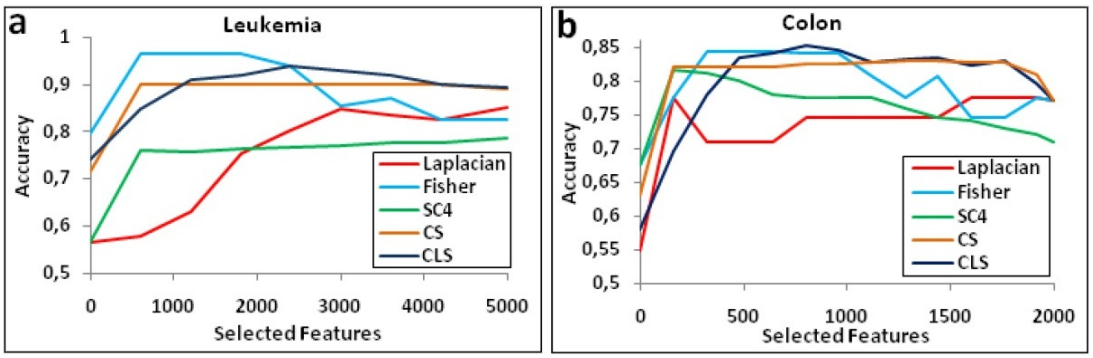


Figure 3.13: Accuracy vs. different numbers of selected features on gene expression datasets.

compare our results with (F3+r) and (F2+r^4) because they achieved best results over the other variant scores proposed by the authors in [Zhao and Liu, 2007b].

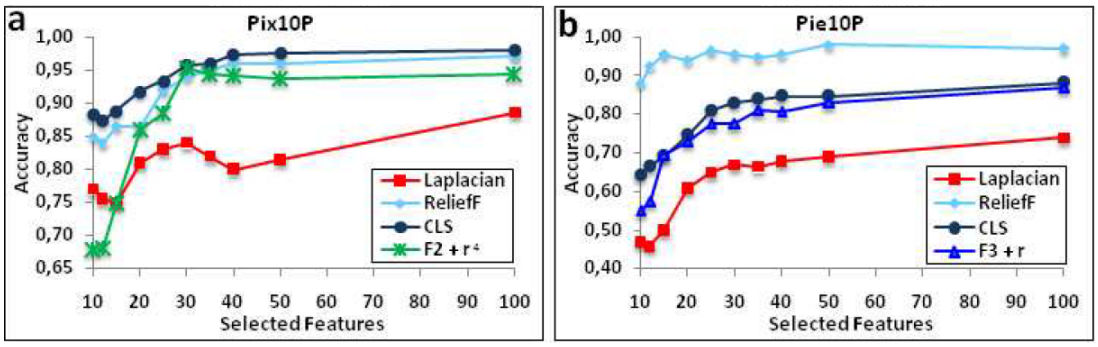


Figure 3.14: Accuracy vs. different numbers of selected features on face-image datasets.

The results in Figure 3.14 show that CLS outperforms significantly the other scores whatever the exploited number of features. Meanwhile, on "PIE10P" dataset, CLS is higher than Laplacian and (F3+r) scores and inferior to ReliefF. Nevertheless, it could be shown that CLS has an excellent accuracy on "PIX10P" dataset and very good one on "PIE10P" dataset.

3.7 Results of CSFS

In this section, we present an empirical study of CSFS framework over some datasets from Table 3.1 (“Iris”, “Ionosphere”, “Sonar”, “Soybean”, “Leukemia” and “Colon Cancer”).

In order to compare our feature selection framework with other methods, we initialize the common parameters with the same values used in CLS (see Section 3.6.1). In addition, the parameters of CSFS are configured as follows:

In order to implement the adaptive k-nearest neighborhood, we cluster the data $(X_L \cup X'_U)$ by an Ascendant Hierarchical Clustering (AHC) [Dash and Liu, 1997]. Then, an internal index, Davies Bouldin [Mali and Mitra, 2003], is used for cutting the dendrogram resulting from AHC in order to obtain the optimal

number of clusters. Note that with this strategy, we obtain several values of k for each dataset. These values are not manually determined, but they are automatically settled based on the structure of each dataset. In addition, we deploy our constraint selection procedure in order to choose the most coherent subset of the generated constraints.

We compare CSFS with a variety of feature selection and extraction methods. In addition, we compare CSFS with CLS in order to verify the efficiency of the concept proposed by CSFS over CLS.

3.7.1 Results on UCI datasets

In this section we assess the relative performance of CSFS over other dimensionality reduction methods for classification. We choose the semi-supervised version of Laplacian score (Section 2.7.1) as the baseline. We compare CSFS results with CS and CLS methods. We also test the performance of the supervised Fisher score, which uses the class labels of all the training data. As mentioned before, after dimensionality reduction, the nearest neighborhood (1-NN) classifier is employed for classification. In addition, the coherent constraints (selected by Algorithm 9) on datasets are: (8 for Iris, 13 for Ionosphere, 11 for Sonar and 7 for Soybean).

(Figure 3.15) shows that CSFS always achieves the highest accuracy on all datasets. In particular, CSFS outperforms constraint and Laplacian score significantly, while it outperforms or achieves similar accuracy to CLS. Note that Fisher uses the full labels of the dataset while CSFS uses a subset of coherent constraints generated originally from a small-labeled data part (25%). Note also that CSFS achieves better results than its ancestor CLS. This validate the three principal ideas which were proposed in CSFS: the adaptive k-neighborhood, the improved scoring function, and finally the constraint selection process. In fact, CSFS achieves such performance using fewer constraints than CLS utilizes.

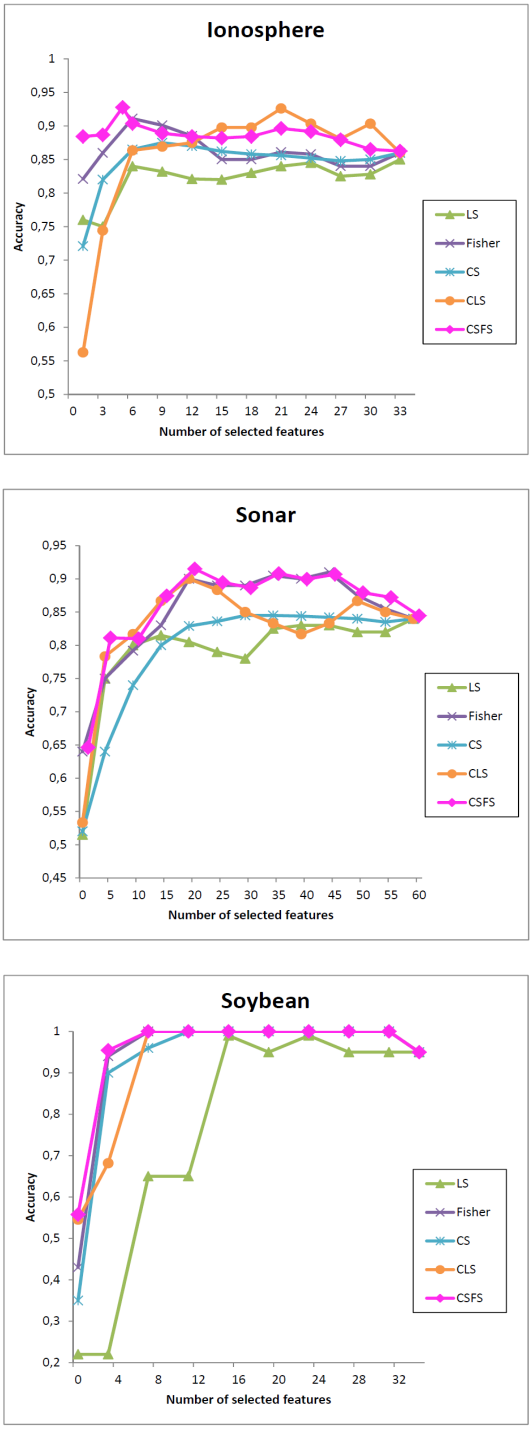


Figure 3.15: Accuracy vs. different numbers of selected features.

It is remarkable too that CSFS provides good accuracy even with a small number of selected features. These results verify that merging *useful* constraints extracted from supervision information with geometrical structure of unlabeled

data is beneficial in learning feature scores. Then, we compare the performance of CSFS with that of PCA, cFLD and SSDR-CMU (Figure 3.16). This comparison concerns the Accuracy vs. different number of constraints (we used 50% of selected features)

Note that the authors in [Zhang et al., 2007] proposed the SSDR score with different variants (SSDR-M, SSDR-CM and SSDR-CMU). We compared our results with SSDR-CMU because it uses the two types of pairwise constraints in addition to the unlabeled data, which means that it uses the same specifications that we consider in our score function. In addition, SSDR-CMU recorded better results than the other SSDR variants. The comparison of our framework with the listed scores is presented under different levels of selected constraints.

Note also that CSFS deploys just the coherent constraints from the whole constraint set generated from the labeled data. This can explain that the maximum number of selected constraints in (Figure 3.16) is much less than the maximum number of possible constraints. This figure shows that CSFS outperforms the PCA and cFLD scores significantly, and it is comparable to SSDR-CMU on Soybean, outperforms it in “Sonar” and “Ionosphere”, but inferior to it on “Iris” when SSDR-CMU exploits the full constraints set. CSFS achieves a high accuracy even when few coherent constraints are deployed.

Another important notice from (Figure 3.16) is that CSFS accuracy on “Sonar” and “Ionosphere” datasets is higher than the other score accuracies even when they deploy the full constraints set; this validates the practically proven fact that the use of more incoherent constraints would have ill effects on learning performance (or it would have no effects in the best cases).

3.7.2 Results on Leukemia and Colon Cancer datasets

In this section, we present our results on these datasets on comparison with Laplacian, Fisher, SC4 and CS scores. This comparison is presented in the form of Accuracy vs. the selected constraints (50% of the selected features were deployed).The coherent constraints used for this comparison are: 7 for

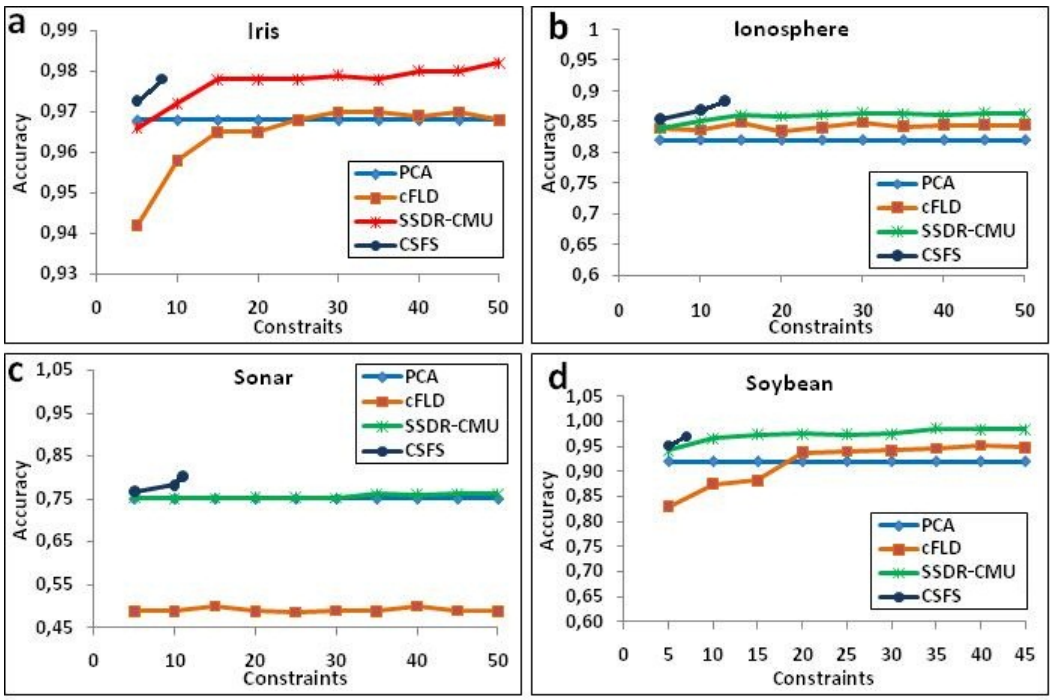


Figure 3.16: Accuracy vs. different numbers of selected constraints (coherent constraints for CSFS).

"Colon Cancer" and 8 for "Leukemia". The results of the classification (Figure 3.17) show that CSFS outperforms other scores when using the full coherent constraint sets, and as on UCI datasets, the accuracy achieved by CSFS on "Leukemia" dataset is not reached by other scores even when using the whole possible constraints set.

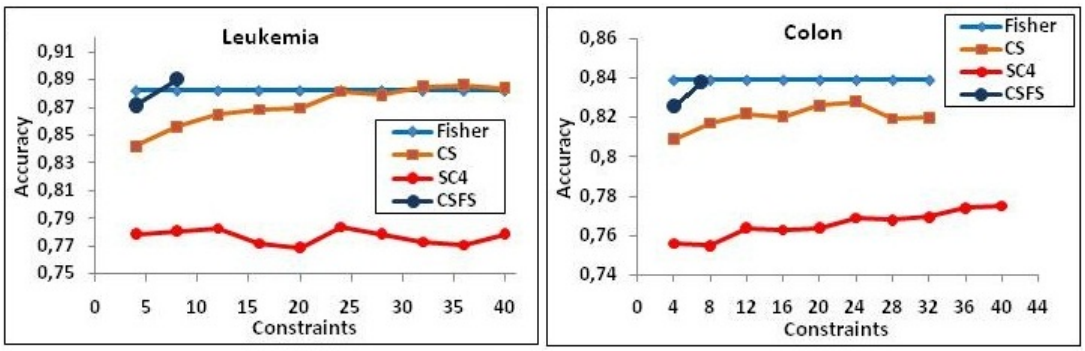


Figure 3.17: Accuracy vs. different numbers of selected constraints.

3.8 Results of CSFSR

In this section, we empirically evaluate the performance of the algorithm derived by CSFSR framework (presented in Figure 3.7). The study is done in the semi-supervised context with redundancy analysis.

3.8.1 Datasets and methods

In the experiments, we consider additional datasets "PCMAC", "RELATHE", "TOX-171", "CLL-SUB-111", and "ORL10P", with "Wave" and "PIE10P". These datasets are known to have redundant features. In addition, they were used by several competitive methods in order to validate their redundancy elimination-based approaches [Zhao et al., 2012]. The whole datasets information is detailed in Table 3.3 in which the last column (S) represents the percentage of supervision[5]. The datasets are high dimensional, with different number of

Table 3.3: Additional Datasets

Datasets	n	m	K	S
PCMAC	1943	3289	2	0.3%
RELATHE	1427	4322	2	0.4%
TOX-171	171	5748	4	7%
CLL-SUB-111	111	11340	3	8%
ORL10P	100	10000	10	30%

classes and few supervision; for evaluating the performance of CSFSR and comparing it with other methods. We choose eight representative methods, the first five of them are semi-supervised and mostly based on constraints, and the other three ones represent algorithms that can handle feature redundancy. All concerned methods are listed as follows:

- sSelect [Zhao and Liu, 2007a] (described in Section 2.7.2).
- SSDR [Zhang et al., 2007] (described in Section 2.7.3).

[5] The source for all these datasets is [Zhao et al., 2011] and can be downloaded from (http://featureselection.asu.edu/datasets.php).

- SC4 [Kalakech et al., 2011] (described in Section 2.7.6).
- CLS [Benabdeslem and Hindawi, 2011] (described in Section 3.3).
- CSFS [Hindawi et al., 2011] (described in Section 3.4).
- AROM-SVM [Weston et al., 2003], mRMR [Peng et al., 2005] and SPSF [Zhao et al., 2012] (all described in Section 2.4).

3.8.2 Experimental setting for CSFSR

To simulate the "small labeled-sample" context, we set l, the number of labeled data, by randomly selecting 3 instances per class and the remaining instances are used as unlabeled data. The portion of supervised information is very small for each data set (see the last column ($S = \frac{3K}{n}$) in Table 3.3).

The parameter λ is always set to 0.1 in all our experiments. For the other methods compared, we respect the same parameters taken by the associated authors.

Each data set is split (in a stratified way) into a training partition with 50% of the instances and a test partition with the remaining 50% of instances. After feature selection, a linear SVM classifier [Vapnik, 1995] (using LIBSVM package [Chang and Lin, 2011]) is employed for classification accuracy. The classifier is tuned via 2-fold cross-validation to training data set by repeating the process 20 times on 20 different partitions of the data.

In addition, we evaluate the clustering accuracy by comparing the label obtained from each instance with that provided by the data corpus. To do this, we use Rand index [Rand, 1971] to measure the clustering quality. This index measures the correspondence between two partitions $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ of a data set X. In our case, $\boldsymbol{P}_1$ is the correct partition produced by labels of predefined classes and $\boldsymbol{P}_2$ is the partition obtained from the clustering algorithm. Each partition is regarded as a set of $n(n-1)/2$ pairs of decisions. For each pair of instances (x_i, x_j), $\boldsymbol{P}_k$ assigns them to the same class or to two different classes. Assuming $c_=$ is the number of decisions where x_i belongs to the same class as x_j

in $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$, and $c_{\neq}$ is the number of decisions where x_i and x_j do not belong to the same class in $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$. We then obtain $(c_{=} + c_{\neq})$ correct decisions and the accuracy between $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ is:

$$Rand = \frac{c_{=} + c_{\neq}}{n(n-1)/2} \tag{3.32}$$

This external index is widely used to evaluate the clustering approaches. We used it when comparing our approach with the best known approaches using the same measure.

Finally, for redundancy analysis, we use the same measure used by [Zhao et al., 2012]:

$$RED(F) = \frac{1}{m(m-1)} \sum_{F_i, F_j \in F, i>j} \rho(F_i, F_j) \tag{3.33}$$

where F is the final set of selected features, $\rho_{i,j}$ returns the Pearson correlation between two features F_i and F_j. The measurement assesses the averaged correlation among all feature pairs, and a large value indicates that many selected features are strongly correlated, and thus redundancy is expected to exist in F.

3.8.3 Validation on "Wave" dataset

In this section, we are particularly interested in the waveform of Breiman "Wave" dataset (described in section 3.6.2).

After applying CSFSR, we obtain the results presented in Figure 3.18. In the top side of the figure we present the inverse of feature scores from the dataset. Note that a feature is relevant if its score is low according to our developed score function, but here, we show the inverse of scores for an efficient visualization of feature relevances with three colors. The red color represents the irrelevant features, the blue color represents the relevant features and the green color

represents the relevant and non-redundant ones. We can see that the features (22 to 40) have low values on their inverse scores, so the noise represented by these features is clearly detected.

In the bottom side of the figure, we show the classification accuracy vs. different number of selected features with four curves. The black curve plots the accuracy using all features in the dataset, while the red one represents the

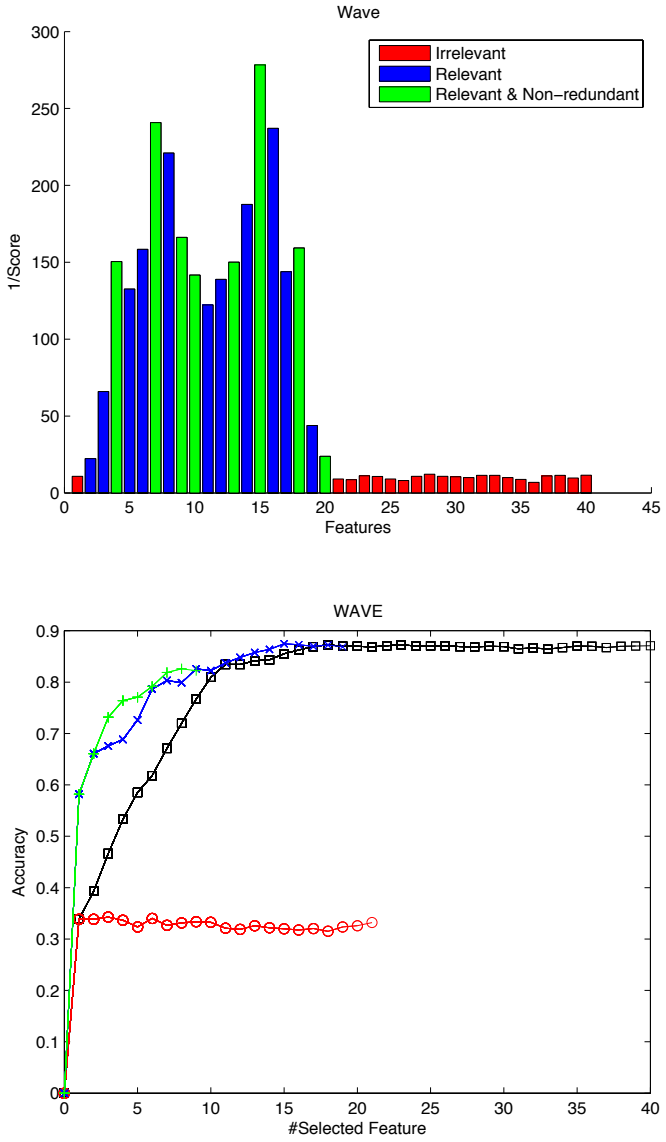


Figure 3.18: Results on "Wave" dataset. Top: Relevance of features. Bottom: Classification accuracy.

accuracy with the irrelevant features detected by CSFSR. We can see that the performance is very weak when the learning is done using noise features only. The other two curves (blue and green) outperforms the black one. Both curves increase steadily over the first twenty features whose the inverse scores are the high ones in the top side of the figure. However, the green curve (accuracy with non-redundant features) is better than the blue one, it increases more rapidly and achieves good performance with few number of features.

3.8.4 Feature quality on high-dimensional data

In this section, we assess the performance of CSFSR framework and compare it with the above cited methods on high-dimensional data. The comparison is conducted by measuring both classification and clustering analysis. Indeed, in the semi-supervised context, the aim could concern the supervised learning according to the labeling of data; and the unsupervised learning according to the geometrical structure of data.

Comparison on classification performance

In this first scenario, we compare the performance of the CSFSR framework with and without redundancy elimination. In addition, we compare the approach with other semi-supervised features selection methods. This comparison concerns the classification accuracy results that we present in both Figure 3.19 and Table 3.4.

Figure 3.19 shows the performance of CSFSR algorithm by classification accuracy (SVM) versus a different number of selected features. For each data set, two curves are plotted. The blue one represents the accuracy on the top relevant features without redundancy analysis. The green curve represents the accuracy with the top relevant and non-redundant features. We can see that generally speaking, the green curves outperform the blue ones, especially in the beginning, with a small number of selected features. This means that the redundancy function applied over the relevant subset of features, is necessary to optimize this subset by providing good learning performance.

Table 3.4 compares the averaged accuracies under different number of selected features of different algorithms on each data set. The measures are obtained by averaging the best accuracies achieved by the SVM classifier using the top 200 features selected by each algorithm. The values after the symbol $\pm$ denote the standard deviation and those between brackets represent the optimal number of selected features which provide the best learning performance. In this table, we can observe that CSFSR outperforms the baseline algorithms. Indeed, by calculating the differences in averaged accuracies among algorithms, we can see that in terms of accuracy gains, CSFSR is 8.24% better than sSelect, 5.34% better than SC4, 5.96% better than CLS and 5.6% better than CSFS. This observation suggests that the compromise between the label information and the geometrical structure of data, is more adopted to semi-supervised feature selection with our method than the others.

For example, in the ORL data set, the result obtained by CSFSR (96.67) is comparable with that obtained by CLS (96.76). However, CSFSR achieves this accuracy with a smaller number of selected features (76) than CLS (93). The results further verify that our proposal can guarantee that the optimal size of the feature subset not only achieves a higher degree of dimensionality reduction but also gives better discriminability (classification).

Comparison on clustering performance

To show how the dimensionality of the projected space affects the locality preserving ability, we compare the clustering accuracies with a fixed number of selected features. Note that this number is automatically determined by CSFSR and is different for each data set. Thus, we use the same number for the other methods and report the clustering results (Rand index) in Table 3.5. For the clustering, we perform k-means algorithm over the selected feature subspaces. The process is repeated 20 times with different initializations and the best result in terms of the objective function is recorded.

As can be noted, CSFSR is very competitive with the other algorithms. For example, it performs much better than SSDR for dimensionality reduction,

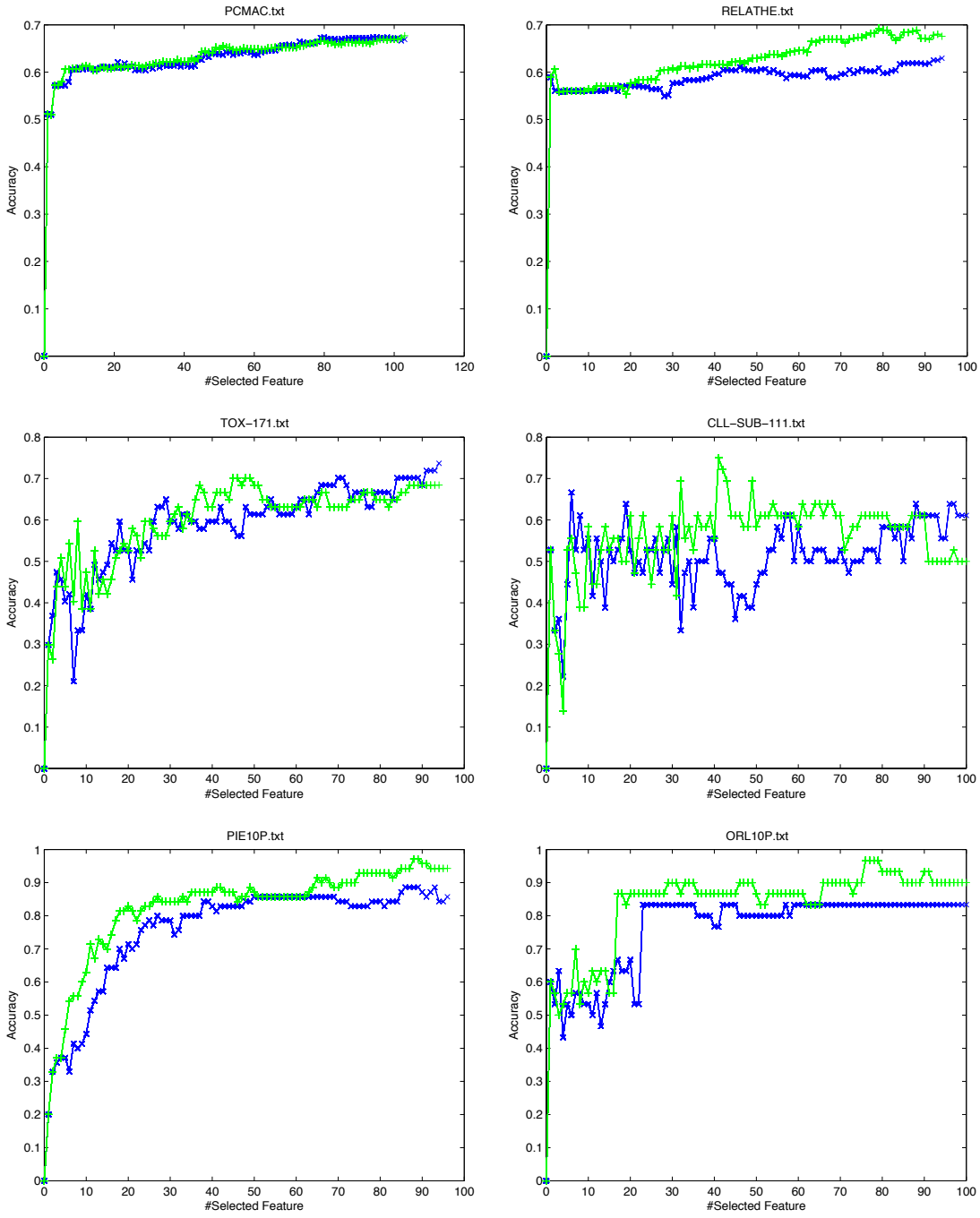


Figure 3.19: Classification accuracy vs. different number of selected features

Table 3.4: Classification Accuracy (SVM in %: the higher the better).

Datasets	sSelect	SC4	CLS	CSFS	CSFSR
PCMAC	66.3±3.75(80)	63.68±2.41(158)	65.26±3.71(110)	65.55±7.45(80)	**67.70±3.01(103)**
RELATHE	68.26±2.52(82)	64.77±0.51(98)	63.79±1.19(81)	64.02±0.47(94)	**69.26±4.25(79)**
TOX-171	60.56±8.36(58)	62.9±7.1(61)	62.39±7.74(48)	**73.56±5.8(93)**	70.18±9.23(45)
CLL-SUB-111	62.03±4.66(61)	69.44±8.55(66)	64.44±6.97(62)	67.9±6.01(7)	**75.22±8.93(40)**
PIE10P	84.33±4.49(88)	86.8±9.77(101)	87.8±5.97(88)	88.76±6.92(86)	**97.14±14.46(88)**
ORL10P	85.25±3.63(82)	96.56±2.95(97)	**96.76±2.5(93)**	82.77±2.6(22)	**96.67±11.63(76)**
WIN	0	0	1	1	**5**

Table 3.5: Clustering Accuracy (Rand index in %: the higher the better).

Datasets	sSelect	SSDR	SC4	CLS	CSFS	CSFSR
PCMAC	49.99±0.02	50.0±0.01	50.1±0.01	50.3±0.02	49.9±0.01	**51.18±2.13**
RELATHE	50.2±0.2	50.3±0.01	50.4±0.1	50.4±0.01	50.4±0.01	**50.91±3.7**
TOX-171	63.1±0.75	61.3±0.26	61.6±1.59	61.5±1.74	61.9±1.72	**64.35±2.17**
CLL-SUB-111	54.8±1.31	51.1±3.39	54.9±2.47	53.8±2.37	54.7±2.7	**56.81±1.35**
PIE10P	82.5±2.05	81.5±1.4	81.3±1.49	82.4±1.03	**82.7±0.84**	82.1±0.14
ORL10P	84.37±3.16	75.8±4.2	79.8±1.2	86.9±2.1	82.9±5.18	**87.3±1.75**
WIN	0	0	0	0	1	**5**

when the number of constraints is limited. This indicates that the semi-supervised feature selection achieved by CSFSR is capable of enhancing clustering performance.

3.8.5 Redundancy rate

In Table 3.6, we present the redundancy rates of the top h features selected by different algorithms, where h is the number of features, which is finally selected by CSFSR. Note that this number is automatically determined from the top 200 relevant features and does not exceed the number of instances (n) for each data set used. Indeed, when $h > n$, any feature can be expressed by a linear combination of the remaining ones, which will introduce unnecessary redundancy in the evaluation [Zhao et al., 2012]. The comparisons are made between the methods that handle redundancy and all these methods select features in a supervised context. This means that they use the whole labels while CSFSR deals with little supervision (Table 3.3). For SPSF, there are three

Table 3.6: Averaged redundancy rate (RED index in %: the lower the better).

Datasets	AROM-SVM	mRMR	SPSF	CSFSR
PCMAC	0.04	0.03	0.03	**0.02**
RELATHE	0.05	0.04	0.04	**0.02**
TOX-171	0.15	0.26	0.16	**0.07**
CLL-SUB-111	0.59	0.26	**0.22**	0.28
PIE10P	0.32	0.29	0.24	**0.06**
ORL10P	0.25	0.25	**0.24**	0.41
WIN	0	0	2	**4**

variants (SPSF-SFS, SPSF-NES, SPSF-LAR). We report in the table the best result between the three variants for each data set. We can see in Table 3.6 that generally, CSFSR efficiently removes redundancy (with low values). For this task, it outperforms AROM-SVM and mRMR and is comparable with SPSF.

3.9 Conclusion

In this chapter we proposed three algorithms to solve the problem of semi-supervised feature selection. We presented the first approach, CLS, in which we constrained the Laplacian score in order to take into consideration the background information about data. We translated this information into pairwise constraints (Must-link and Cannot-link constraints). The scoring function of CLS integrated both labeled and unlabeled parts of data. However, with a review of literature of pairwise constraints, it was practically proven that these constraints may have some noise and thus deteriorate the learning performance. To overcome this problem, we proposed CSFS framework, in which we exploited a constraint selection procedure based on a measure form the literature. In addition, we reduced the number of parameters which were needed from the ancestor method CLS. This parameter is the k-neighborhood, that we proposed to automatically calculate depending on the structure of the data. Moreover, in order to treat the redundancy in the selected features, we proposed CSFSR, in which we extended CSFS by a graph-based approach to eliminate the redundancy in selected features. Finally, we presented a variety of empirical results over high-dimensional data, and compared our methods to other competitive approaches with different scenarios. The results were promising and proved the efficiency of the proposed approaches.

4 Weighting-Based Semi-Supervised Feature Selection

4.1 Introduction

Embedded feature selection methods are locally specific to a model during its construction. They aim to learn the feature relevance with the associated learning algorithm. In other terms, they incorporate feature selection and learning algorithm in the same objective function. In this chapter, we investigate in an embedded semi-supervised feature selection using the well known k-means clustering algorithm [MacQueen, 1967]. We do this in two scenarios, the first one uses a fuzzy variant of k-means based on feature weighting and relaxed integration of pairwise constraints. The second approach uses the traditional form of k-means with a strict application of pairwise constraints. We start by a brief recall about the k-means algorithm, and some approaches which proposed a semi-supervised version of k-means. Then, we present two methods for weighting-based semi-supervised feature selection.

4.2 k-means type clustering

k-means is a very well known partitioning based clustering algorithm [MacQueen, 1967]. It is based on iterative relocation that partitions a dataset into K clusters, locally minimizing the total squared Euclidean distance between the data instances and the cluster centroids. k-means aims at building parti-

tions on the basis of an objective function. The main design challenge lies in formulating an objective function that is capable of reflecting the nature of the problem such that minimizing this function reveals a meaningful structure in the dataset.

Let $X = \{x_1, x_2, .., x_n\}$ be a set of n data instances. Each x_i is characterized by m features $x_{i1}, x_{i2}, ..., x_{im}$. *k*-means' algorithm searches for a partition of X into K clusters $c_1, c_2, ..., c_K$ that minimizes the objective function γ_0:

$$\gamma_0(a, c) = \sum_{q=1}^{K} \sum_{i=1}^{n} \sum_{j=1}^{m} a_{iq}(x_{ij} - c_{qj})^2 \tag{4.1}$$

subject to

$$\sum_{q=1}^{K} a_{iq} = 1,\ i = 1..n \tag{4.2}$$

where a_{iq} is the membership of x_i to the cluster c_q. In the basic *k*-means, the assignment is done in a *hard* manner, where each instance x_i belongs to the cluster c_q if $a_{iq} = 1$. The same instance is excluded from the cluster if $a_{iq} = 0$. In other situations, the assignment can be done *softly*, when each instance belongs to all clusters with different scores. In this case, we consider a fuzzy partition in which the total membership degrees sum to one [Bezdek, 1981]. More formally, in the *hard* version of *k*-means $a_{ik} \in \{0, 1\}$ and in the *fuzzy* version $a_{ik} \in [0, 1]$ with the same constraint in eq.(4.2) for both versions.

4.3 Semi-Supervised *k*-means clustering

The last decade has witnessed extensive works on semi-supervised clustering. The early work in this area [Wagstaff and Cardie, 2000] has proposed a modified version of COBWEB [Fisher, 1987], called COP-COBWEB, which strictly enforced pairwise constrains. It was followed by an enhanced version of the widely used *k*-means algorithm which could also accommodate constraints, called COP-Kmeans [Wagstaff et al., 2001]. The common representation for background information pertaining to X is in the form of pairwise constraint sets: must-link constraints (Ω_{ML}) and cannot-link constraints (Ω_{CL}). The au-

thors in [Basu et al., 2002] proposed two variants of *k*-means algorithms: *seeded k-means*, and *constrained k-means*. Both variants initialize the *k*-means algorithm by assigning the instances with different labels to different clusters. The difference is that Constrained *k*-means keep the same assignment for labeled instances during algorithm execution, while seeded *k*-means does not do that. In addition, the authors in [Bilenko. et al., 2004] proposed to incorporate this information into traditional partitional clustering algorithms by adapting the objective function to include penalties for violated constraints. They proposed to minimize:

$$\gamma_1(a,c) = \gamma_0(a,c) + \vartheta_{ML} + \vartheta_{CL} \tag{4.3}$$

subject to the same constraint in eq.(4.2).

The second and third terms in eq.(4.3) represent the penalty costs of violation constraints in Ω_{ML} and Ω_{CL} respectively. These terms control the influence given to external information during the assignment phase of the algorithm. eq.(4.3) has been shown to have a probabilistic basis related to the assignment of labels in *Hidden Markov Random Fields* [Bilenko. et al., 2004].

Furthermore, the constraints were also introduced into the complete linkage algorithm [Klein et al., 2002], the EM of a Gaussian mixture model [Shental et al., 2003] and more recently the hierarchical clustering [Gilpin and Davidson, 2011], and spectral clustering [Wang and Davidson, 2010].

4.3.1 A fuzzy approach for feature selection (wCKM)

The weighting-based feature selection has been an important research topic in clustering analysis [Green et al., 1990, Makarenkov and Legendre, 2001, Modha and Spangler, 2003, Huang et al., 2005]. The authors in aforementioned works assumed that the main drawback of *k*-means algorithm is that it treats all features equally when calculating the cluster-membership of instances. Such treatment is undesirable when dealing with high dimensional data. In the following, we present an approach that we call Weighted constrained *k*-Means (termed wCKM as a shorthand). In this approach, we propose to follow

the weighting strategy of fuzzy *k*-means version for semi-supervised feature selection.

As defined before, in the context of semi-supervised learning, the dataset X consists of two subsets depending on the label availability: $\{x_1, ..., x_l\}$ for which the labels $\{y_1, ..., y_l\}$ are provided, and $\{x_{l+1}, ..., x_{l+u}\}$ whose labels are not given. Here, each label $y_i \in \{1, 2, ..., C\}$ where C is the number of different labels, and $l + u = n$ (the number of all instances).

Before describing the method, we perform the following initializations:

- We put $K = C$, so we do not have to inform the *a priori* number of clusters as it is done in *k*-means type clustering algorithms. In fact, the choice of the number of clusters K is a critical issue in *k*-means type clustering algorithms, because it generally influences the whole clustering process. In the semi-supervised clustering, it is considered that such information is supplied in the labeled part which carries the background information about the target concept. Strictly speaking, we consider that there is at least one labeled instance in X_L for each desired cluster.
- We construct the different constraints (Ω_{ML} and Ω_{CL}) from the labeled part of data. Ω_{ML} contains pairs of instances that have the same label and Ω_{CL} contains those having different labels. Consequently, the number of all constraints is $|\Omega_{ML} \cup \Omega_{CL}| = \frac{l(l-1)}{2}$.

The idea behind our proposal is to associate to each feature F_j a weight w_j in a new objective function γ_2 to be minimized. The goal is to assign a higher weight to a dimension along which the distance between instances and centroids is smaller.

$$\gamma_2(a, c, w) = \sum_{q=1}^{K} \sum_{i=1}^{n} \sum_{j=1}^{m} a_{iq}^2 w_j^{\beta} (x_{ij} - c_{qj})^2 + \vartheta_{ML} + \vartheta_{CL} \tag{4.4}$$

subject to eq.(4.2), eq.(4.5) and eq.(4.6) :

$$\sum_{j=1}^{m} w_j = 1, \; w_j \in]0, 1[\tag{4.5}$$

$$a_{iq} \in [0,1]; i = 1..n \text{ and } q = 1..K \tag{4.6}$$

where β is a parameter for the feature weights and :

$$\vartheta_{ML} = \sum_{(x_i,x_r)\in\Omega_{ML}} \sum_{p=1}^{K} \sum_{(q=1,l\neq p)}^{K} a_{ip}a_{rq} \tag{4.7}$$

and

$$\vartheta_{CL} = \sum_{(x_i,x_r)\in\Omega_{CL}} \sum_{p=1}^{K} a_{ip}a_{rp} \tag{4.8}$$

Note that β cannot be equal neither to zero nor to one. Indeed, if $\beta = 0$, the weighting is removed and so the feature selection cannot be performed. If $\beta = 1$, w would disappear because of the following derivations for solving the problem. Thus, to solve the optimization problem under these assumptions for β, we minimize eq.(4.4) by solving the following three minimization problems:

- **Optimization: O1**: Minimizing $\gamma_2(a, c, w)$ with respect to c for calculating the centroids of clusters.
- **Optimization: O2**: Minimizing $\gamma_2(a, c, w)$ with respect to a for calculating the cluster-membership values of instances.
- **Optimization: O3**: Minimizing $\gamma_2(a, c, w)$ with respect to w for measuring the weights of features.

O1 represents the centroid updating procedure in the process and can be easily solved, providing the solution:

$$c_{qj} = \frac{\sum_{i=1}^{n} a_{iq}^2 x_{ij}}{\sum_{i=1}^{n} a_{iq}^2}; q = 1..K \; and \; j = 1..m \tag{4.9}$$

O2 represents the assignment step in the process. We use Lagrange multipliers for solving this problem as follows:

$$\zeta(a, c, w, \kappa) = \gamma_2(a, c, w) - \sum_{i=1}^{n} \kappa_i (\sum_{q=1}^{K} a_{iq} - 1) \tag{4.10}$$

With fixed centroids and weights, the pair (κ_i, a_{iq}) is an extremum of the function to optimize when $\frac{\partial \zeta}{\partial \kappa_i} = 0$ and $\frac{\partial \zeta}{\partial a_{iq}} = 0$. The derivations yield the following formulas:

$$\frac{\partial \zeta}{\partial \kappa_i} = \sum_{q=1}^{K} a_{iq} - 1 = 0; i = 1..n \tag{4.11}$$

and

$$\frac{\partial \zeta}{\partial a_{iq}} = 2a_{iq} \sum_{j=1}^{m} w_j^{\beta}(x_{ij} - c_{qj})^2 + \sum_{(x_i,x_r)\in\Omega_{ML}} \sum_{(p=1,p\neq q)}^{K} a_{rp} + \sum_{(x_i,x_r)\in\Omega_{CL}} a_{rq} - \kappa_i = 0 \tag{4.12}$$

From eq.(4.12) we can obtain:

$$a_{iq} = \frac{\kappa_i - \sum_{(x_i,x_r)\in\Omega_{ML}} \sum_{(p=1,p\neq q)}^{K} a_{rp} - \sum_{(x_i,x_r)\in\Omega_{CL}} a_{rq}}{2\sum_{j=1}^{m} w_j^{\beta}(x_{ij} - c_{qj})^2} \tag{4.13}$$

Such that κ_i can be obtained using eq.(4.11) and eq.(4.13):

$$\begin{aligned} \kappa_i &= \frac{1}{\sum_{t=1}^{K} \frac{1}{2\sum_{j=1}^{m} w_j^{\beta}(x_{ij}-c_{tj})^2}} \\ &+ \frac{\sum_{t=1}^{K} \frac{\sum_{(x_i,x_r)\in\Omega_{ML}} \sum_{(p=1,p\neq t)}^{K} a_{rp} + \sum_{(x_i,x_r)\in\Omega_{CL}} a_{rt}}{2\sum_{j=1}^{m} w_j^{\beta}(x_{ij}-c_{tj})^2}}{\sum_{t=1}^{K} \frac{1}{2\sum_{j=1}^{m} w_j^{\beta}(x_{ij}-c_{tj})^2}} \end{aligned} \tag{4.14}$$

We can rewrite eq.(4.13) as:

$$\begin{aligned} a_{iq} = \frac{\frac{1}{\sum_{j=1}^{m} w_j^{\beta}(x_{ij}-v_{qj})^2}}{\sum_{t=1}^{K} \frac{1}{\sum_{j=1}^{m} w_j^{\beta}(x_{ij}-c_{tj})^2}} &+ \frac{1}{2\sum_{j=1}^{m} w_j^{\beta}(x_{ij} - c_{qj})^2} \\ &\times \left[\frac{\sum_{t=1}^{K} \frac{\sum_{(x_i,x_r)\in\Omega_{ML}} \sum_{(p=1,p\neq t)}^{K} a_{rp} + \sum_{(x_i,x_r)\in\Omega_{CL}} a_{rt}}{\sum_{j=1}^{m} w_j^{\beta}(x_{ij}-c_{tj})^2}}{\sum_{t=1}^{K} \frac{1}{\sum_{j=1}^{m} w_j^{\beta}(x_{ij}-c_{tj})^2}} \right. \\ &- \left. \sum_{(x_i,x_r)\in\Omega_{ML}} \sum_{(p=1,p\neq q)}^{K} a_{rp} - \sum_{(x_i,x_r)\in\Omega_{CL}} a_{rq} \right] \end{aligned} \tag{4.15}$$

Note that the instance assignments represented by eq.(4.15) are done in a soft manner where each instance has K membership values *w.r.t* eq.(4.2) and eq.(4.6).

O3 represents the feature weighting procedure in the process. The solution of this problem allows to update the relevance of features in an embedded manner. We can rewrite eq.(4.4) as follows:

$$\gamma_2(a,c,w) = \sum_{j=1}^{m} w_j^{\beta} \sum_{q=1}^{K} \sum_{i=1}^{n} a_{iq}^2 (x_{ij} - c_{qj})^2 + \vartheta_{ML} + \vartheta_{CL} \tag{4.16}$$

To solve this problem, we can consider the relaxed minimization via a Lagrange multiplier by ignoring the constraint in eq.(4.5). Let ϱ be the multiplier and ξ be the Lagrangian:

$$\xi(a,c,w,\varrho) = \gamma_2(a,c,w) - \varrho(\sum_{j=1}^{m} w_j - 1) \tag{4.17}$$

To minimize eq.(4.17) with respect to w and ϱ, the gradient of the following two variables must equal to zero:

$$\frac{\partial \xi}{\partial \varrho} = \sum_{j=1}^{m} w_j - 1 = 0; j = 1..m \tag{4.18}$$

and

$$\frac{\partial \xi}{\partial w_j} = \beta w_j^{\beta-1} \sum_{q=1}^{K} \sum_{i=1}^{n} a_{iq}^2 (x_{ij} - c_{qj})^2 - \varrho = 0 \tag{4.19}$$

From eq.(4.19), we obtain:

$$w_j = (\frac{\varrho}{\beta \sum_{q=1}^{K} \sum_{i=1}^{n} a_{iq}^2 (x_{ij} - c_{qj})^2})^{\frac{1}{\beta-1}}; j = 1..m \tag{4.20}$$

By substituting eq.(4.20) into eq.(4.18), we obtain:

$$\sum_{p=1}^{m} (\frac{\varrho}{\beta \sum_{q=1}^{K} \sum_{i=1}^{n} a_{iq}^2 (x_{ip} - v_{qp})^2})^{\frac{1}{\beta-1}} = 1 \tag{4.21}$$

From eq.(4.21), we have:

$$(\varrho)^{\frac{1}{\beta-1}} = \frac{1}{\sum_{p=1}^{m}\left(\frac{1}{\beta\sum_{q=1}^{K}\sum_{i=1}^{n}a_{iq}^{2}(x_{ip}-c_{qp})^{2}}\right)^{\frac{1}{\beta-1}}} \tag{4.22}$$

By substituting eq.(4.22) in eq.(4.20), we obtain in the final:

$$w_j = \frac{1}{\sum_{p=1}^{m}\left(\frac{\sum_{q=1}^{K}\sum_{i=1}^{n}a_{iq}^{2}(x_{ij}-c_{qj})^{2}}{\sum_{q=1}^{K}\sum_{i=1}^{n}a_{iq}^{2}(x_{ip}-v_{qp})^{2}}\right)^{\frac{1}{\beta-1}}} \tag{4.23}$$

From eq.(4.23), we find that the weight of a feature is dependent on the value of β. Following, we discuss the impact of the different values of this parameter.

- if $\beta < 0$, a high value of $\sum_{q=1}^{K}\sum_{i=1}^{n}a_{iq}^{2}(x_{ij}-c_{qj})^{2}$ leads to a small value of w_j.
- if $0 < \beta < 1$, a high value of $\sum_{q=1}^{K}\sum_{i=1}^{n}a_{iq}^{2}(x_{ij}-c_{qj})^{2}$ leads to a high value of w_j. This is paradoxical with the principle of feature weighting.
- if $\beta > 1$, a high value of $\sum_{q=1}^{K}\sum_{i=1}^{n}a_{iq}^{2}(x_{ij}-c_{qj})^{2}$ leads to a small value of w_j. So, the weight of the feature is then decreased.

Thus, in order to use the weights as measures for evaluating feature relevance, the value of β must be either negative or greater than 1. Indeed, a relevant feature should reduce the distance between instances and their centroids in the associated cluster.

Note that the weights defined by eq.(4.23) depend on the labeling constraints only indirectly through the cluster membership values. Thus, the semi-supervised feature selection represented by this equation combines implicitly the geometrical structure from the unlabeled data with supervision information of labeled data.

Subsequently, we can summarize all the above mathematical developments in Algorithm 13.

Algorithm 13 wCKM

Input: Dataset $X(n \times m)$
Output: Weighted and ranked features

1: Give the parameter β
2: Construct the constraint sets (Ω_{ML} and Ω_{CL}) from labeled part of X
3: Give K as the number of labels in labeled part of X
4: Randomly choose initial centroids $c_1, c_2, ..., c_K$ from X
5: Randomly generate initial weights $w_1, w_2, ..., w_m$ ($\sum_{j=1}^{m} w_j$=1)
6: Calculate the memberships using eq.(4.15)
7: Update the centroids using eq.(4.9)
8: Update the feature weights using eq.(4.23)
9: Iterate between steps 6: and 8: until convergence
10: Rank the features $\{F_j\}$ according to their weights $\{w_j\}$ in descending order.

Lemma 3. *wCKM is computed in time* $O(m \times max(ntK, Log\, m))$*, where* t *is the number of iterations.*

Proof. Step 2 of the algorithm requires l^2 operations. Step 6 calculates the cluster-membership values by nK operations. Step 7 updates the centroids by mK operations and step 8 provides the feature weights after mnK operations. The last step ranks the features according to their weights with $m\ Log(m)$ operations. □

4.3.2 A Local-to-Global Feature Selection (L2GFS)

In this section, we extend the approach proposed by [Huang et al., 2005], which is basically global and unsupervised, to semi-supervised feature selection. We propose a modification to the objective function of the constrained version of *k*-means [Wagstaff et al., 2001]. We add a new unknown variable to the function, the weights w, which would be used to weigh the features at each iteration, and then reduce the effects of irrelevant ones. We propose a local-to-global semi-supervised feature selection approach termed L2GFS as a shorthand. In the following, we start with a detailed local semi-supervised weighted extension to the objective function of *k*-means described in eq.(4.1). In detail, we first weigh the variables regarding the clusters, this means that each feature would have as much weights as the number of clusters. We

believe that this local feature weighting would help at mining the persistent variables which best describe each cluster, instead of having the same features rating over all the clusters. This method results in selecting a coherent feature subset for each cluster (local feature selection). The cluster in its turn regroups a homogeneous instances (instance selection). The application of such technique (as in co-clustering) can help in studying the effects of important factors (here features) that influences specific subset of population (instances). However, the aim of this approach is to use the local feature selection in the goal of having a better global selection. Finally, it is obvious that a feature that well describe a given cluster might not well describe the other ones.

We propose to minimize the new following objective function:

$$\min_{a,c,w} Q(a,c,w) = \sum_{q=1}^{K}\sum_{i=1}^{n}\sum_{j=1}^{m} a_{iq}\, w_{qj}^{\beta}(x_{ij}-c_{qj})^2 \tag{4.24}$$

subject to

$$\begin{cases} \sum_{q=1}^{K} a_{iq} = 1,\ \ 1 \le i \le n, \\ a_{iq} \in \{0,1\},\ \ 1 \le i \le n,\ \ 1 \le q \le K \\ \sum_{j=1}^{m} w_{qj} = 1,\ \ 0 \le w_{qj} \le 1,\ \ 1 \le l \le K \\ \vartheta_{ML} = 0, \vartheta_{CL} = 0. \end{cases} \tag{4.25}$$

Where ϑ_{ML}, ϑ_{CL} are calculated according to eq.(4.7) and eq.(4.8) respectively.

Similarly to solving eq.(4.1), the eq.(4.24) assigns at iteration $(t=0)$ initial K random weights to each feature then the unknown variables a, c and w are optimized iteratively using the following equations:

$$a_{iq}^{(t)} = \begin{cases} \text{if } \sum_{j=1}^{m} w_{qj}^{\beta,(t-1)}(x_{ij} - c_{qj}^{(t-1)})^2 \leq \\ \qquad \sum_{j=1}^{m} w_{sj}^{\beta,(t-1)}(x_{ij} - c_{sj}^{(t-1)})^2 \\ 1 \quad \text{for } 1 \leq s \leq K \\ \quad \text{and } \sum_{(b=1,(x_i,x_b)\in\Omega_{CL})}^{i-1} a_{bq}^{(t)} = 0 \text{ for } i > 1 \\ \quad \text{and } \prod_{(p=1,(x_i,x_p)\in\Omega_{ML})}^{i-1} a_{pq}^{(t)} = 1 \text{ for } i > 1 \\ 0 \text{ otherwise} \end{cases} \tag{4.26}$$

where (for any iteration t)

- For the first instance ($i = 1$), no assessment of constraint non-violation is required since it is the first assignment at the current iteration. Thus, the assignment is just driven by the distance to the cluster centroids.
- For the following instances ($i > 1$)
  - $\{x_b | 1 \leq b \leq (i-1)\}$ means that in each assignment, the assessment of constraints non-violation involves the so far assigned instances only ($b < i$).
  - $(x_i, x_b) \in \Omega_{CL}$(or Ω_{ML}) means that among the so far assigned instances, the non-violation test concerns only the instances in which the current instance x_i is engaged by a CL (or ML) constraint.
  - ($\sum_{(b=1,(x_i,x_b)\in\Omega_{CL})}^{i-1} a_{bq}^{(t)} = 0$) means that the instances $\{x_b | 1 \leq b \leq (i-1)\}$ which are already assigned at the current iteration t and connected to the current instance x_i via CL constraint. No one of these instances must be already affected to the current cluster q, so $a_{bq}^{(t)} = 0$ for all of them.
  - ($\prod_{(p=1,(x_i,x_p)\in\Omega_{ML})}^{i-1} a_{pq}^{(t)} = 1$) means that the instances $\{x_p | 1 \leq p \leq (i-1)\}$ which are already assigned at the current iteration t and connected to the current instance x_i via ML constraint. All these instances must be already affected to the current cluster q, so $a_{pq}^{(t)} = 1$ for all of them.

In general, the transitive closure is important to be applied to the original constraints. For example, if we have three instances (x_1, x_2, x_3) with $(x_1, x_2) \in$

Ω_{ML} and $(x_1, x_3) \in \Omega_{CL}$, then it is obvious to add (x_2, x_3) to Ω_{CL}. This issue is implicitly done when the constraints are automatically generated from labeled part of data.

The calculation of the new cluster centroids remains the same as in the standard *k*-means version:

$$c_{qj}^{(t)} = \frac{\sum_{i=1}^{n} a_{iq}^{(t)} \, x_{ij}}{\sum_{i=1}^{n} a_{iq}^{(t)}} \text{ for } 1 \leq q \leq K \text{ and } 1 \leq j \leq m \tag{4.27}$$

Theorem 1. *The calculation of the new local weights could be done by the following equation:*

$$w_{qj} = \begin{cases} 0 & if \sum_{i=1}^{n} a_{iq} \, (x_{ij} - c_{qj})^2 = 0 \\ \dfrac{1}{\sum_{s=1}^{h} \left[\frac{\sum_{i=1}^{n} a_{iq} \, (x_{ij} - c_{qj})^2}{\sum_{i=1}^{n} a_{iq} \, (x_{is} - c_{qs})^2} \right]^{\frac{1}{\beta - 1}}} & if \sum_{i=1}^{n} a_{iq} \, (x_{ij} - c_{qj})^2 \neq 0 \end{cases} \tag{4.28}$$

$$1 \leq q \leq K \textit{ and } 1 \leq j \leq m$$

where

h *is the number of features where* $\sum_{i=1}^{n} a_{iq} \, (x_{ij} - c_{qj})^2 \neq 0$.

Proof. We rewrite the objective function eq.(4.24) as follows

$$Q(a, c, w) = \sum_{j=1}^{m} \sum_{q=1}^{K} \sum_{i=1}^{n} w_{qj}^{\beta} a_{iq} \, (x_{ij} - c_{qj})^2 = \sum_{j=1}^{m} \sum_{q=1}^{K} w_{qj}^{\beta} \sum_{i=1}^{n} a_{iq} \, (x_{ij} - c_{qj})^2 \tag{4.29}$$

where $\sum_{i=1}^{n} a_{iq} \, (x_{ij} - c_{qj})^2$ are constants for fixed a and c. If $\sum_{i=1}^{n} a_{iq} \, (x_{ij} - c_{qj})^2 = 0$, this means that the j^{th} variable F_j has the same value for all instances in the cluster q. This is a degenerate solution, so we assign $w_{qj} = 0$ to any feature where $\sum_{i=1}^{n} a_{iq} \, (x_{ij} - c_{qj})^2 = 0$. Note that a 0 weight to a feature in this case is

only related with certain cluster in which the variable have identical value for all instances in this cluster. Moreover, $w_{jq} = 0$ if the variable F_j has a unique value for all instances in the cluster c_l. In this case, we assign 0 to variable in order to satisfy the third constraint in eq.(4.25). Then, at the end of the algorithm, we check variables that have $w_{jq} = 0$ (for all q). The variable is chosen if it has a unique value in each cluster (different from its values in other ones), and it is rejected if it has the same value in all clusters.

For the $h(\leq m)$ feature weights where $\sum_{i=1}^{n} a_{iq}\ (x_{ij} - c_{qj})^2 \neq 0$ (we consider the reasoning for one cluster for simplification purpose), we minimize the function via the Lagrangian multiplier. Let ς be the multiplier and $\Gamma(W, \varsigma)$ be the Lagrangian. By ignoring the constraint $\sum_{j=1}^{m} w_{qj} = 1$ we obtain:

$$\Gamma(w, \varsigma) = \sum_{j=1}^{h}\sum_{l=1}^{K} w_{qj}^{\beta}\ \sum_{i=1}^{n} a_{iq}\ (x_{ij} - c_{qj})^2 - \sum_{q=1}^{K} \varsigma_q \left(\sum_{j=1}^{m} w_{qj} - 1\right) \tag{4.30}$$

The two sets of variable derivatives (w, ς) must vanish and then we would have

$$\frac{\partial \Gamma}{\partial w_{qj}} = \beta w_{qj}^{\beta-1} \sum_{i=1}^{n} a_{iq}\ (x_{ij} - c_{qj})^2 - \varsigma_q = 0 \quad \text{for} \quad 1 \leq j \leq h,\ 1 \leq q \leq K \tag{4.31}$$

$$\frac{\partial \Gamma}{\partial \varsigma_q} = \sum_{j=1}^{h} w_{qj} - 1 = 0. \tag{4.32}$$

From eq.(4.31) we obtain

$$w_{qj} = \left(\frac{\varsigma_q}{\beta \sum_{i=1}^{n} a_{iq}\ (x_{ij} - c_{qj})^2}\right)^{\frac{1}{\beta-1}} \quad \text{for} \quad 1 \leq j \leq h,\ 1 \leq q \leq K \tag{4.33}$$

Substituting eq.(4.33) into eq.(4.32), we have

$$\sum_{s=1}^{h} \left(\frac{\varsigma_q}{\beta \sum_{i=1}^{n} a_{iq}\ d(x_{is}, c_{qs})}\right)^{\frac{1}{\beta-1}} = 1. \tag{4.34}$$

From eq.(4.34), we derive

$$\left(\varsigma_q\right)^{\frac{1}{\beta-1}} = \frac{1}{\left[\sum_{s=1}^{h}\left(\frac{1}{\beta\sum_{i=1}^{n}a_{iq}\,(x_{is}-c_{qs})^2}\right)^{\frac{1}{\beta-1}}\right]}. \tag{4.35}$$

Substituting eq.(4.35) into eq.(4.33), we obtain

$$w_{qj} = \frac{1}{\sum_{s=1}^{h}\left[\frac{\sum_{i=1}^{n}a_{iq}\,(x_{ij}-c_{qj})^2}{\sum_{i=1}^{n}a_{iq}\,(x_{is}-c_{qs})^2}\right]^{\frac{1}{\beta-1}}} \tag{4.36}$$

□

With eq.(4.36) the objective function eq.(4.24) is minimized locally over each cluster, and then the features are ranked in each one by the local weights w_{qj}. These weights express the relevance of each feature F_j regarding each corresponding cluster c_q. The local ranking is suitable when searching the features that best describe each cluster. In this approach, we are interested in a global feature weighting. In order to achieve this goal, we aggregate the weights of each variable over all clusters, to do so we write:

$$w_j = \frac{1}{K}\sum_{q=1}^{K} w_{qj} \tag{4.37}$$

The global weighting rank all variables and then the features are selected regarding to their global weights.

Subsequently, we can summarize all the above mathematical developments in Algorithm 14.

Lemma 4. *L2GFS converges to a local minimal solution in a finite number of iterations.*

Proof. Assume that the cluster-membership for all instances did not change between two different iterations t_1 and t_2 then $a^{(t_1)} = a^{(t_2)}$. We note that given a cer-

Algorithm 14 L2GFS

Input: Dataset $X(n \times m)$
Output: Weighted and ranked features

1: Give the parameter β
2: Construct the constraint sets (Ω_{ML} and Ω_{CL}) from labeled part of X
3: Give K as the number of labels in labeled part of X
4: Randomly choose initial centroids $c_1, c_2, ..., c_K$ from X
5: Randomly generate initial local weights for each variable $w_{j1}, w_{j2}, ..., w_{jq}$ $1 \leq j \leq m$ ($\sum_{j=1}^{m} w_{j,l}$=1)
6: Calculate the memberships using eq.(4.26)
7: Update the centroids using eq.(4.27)
8: Update the local variable weights using eq.(4.28)
9: Iterate between steps 6: and 8: until convergence
10: Calculate the global variable weights using eq.(4.37)
11: Rank the features $\{F_j\}$ according to their weights $\{w_j\}$ in descending order.

tain value of the cluster-membership $a^{(t)}$, we can compute the cluster centroids $c^{(t)}$ by eq.(4.26) which is independent of the variable weight $w^{(t)}$. For $a^{(t_1)}$ and $a^{(t_2)}$, we have the centeroids $c^{(t_1)}$ and $c^{(t_2)}$, respectively. It is clear that $c^{(t_1)} = c^{(t_2)}$ since $a^{(t_1)} = a^{(t_2)}$. Using $a^{(t_1)}$ and $c^{(t_1)}$, and $u^{(t_2)}$ and $c^{(t_2)}$, we can compute their corresponding weights: $w^{(t_1)}$ and $w^{(t_2)}$, respectively (according to eq.(4.28)). Again, as $w^{(t_1)} = w^{(t_2)}$, therefore, $Q_1(a^{(t_1)}, c^{(t_1)}, w^{(t_1)}) = Q_2(a^{(t_2)}, c^{(t_2)}, w^{(t_2)})$. □

4.4 Experimental results

In this section, we present empirical results of wCKM and L2GFS approaches. In the first part, we introduce the results of wCKM versus different competitive methods. In the second part, we compare L2GFS to wCKM in several learning scenarios in order to validate both approaches. The experiments are performed over several high-dimensional datasets, and show the comparison of our proposals with other semi-supervised feature selection ones. In addition, we compare them with some well-known semi-supervised clustering methods since they rely on a clustering algorithm for embedded feature selection.

4.4.1 Datasets and methods

We use the following benchmarking datasets for the comparisons: "Wave", "PCMAC", "RELATHE", "TOX-171", "CLL-SUB-111", "PIE10P" and "PIX10P". The datasets are high dimensional, with different number of classes and few supervision; for evaluating the performance of wCKM and comparing it with other methods. All concerned methods are semi-supervised and most of them are based on constraints:

- cFLD, is a dimensionality reduction method, using equivalence constraints in relevant component analysis (RCA) [Bar-Hillel et al., 2005].
- sSelect [Zhao and Liu, 2007a] (described in Section 2.7.2).
- SSDR [Zhang et al., 2007] (described in Section 2.7.3).
- SC4 [Kalakech et al., 2011] (described in Section 2.7.6).
- CLS [Benabdeslem and Hindawi, 2011] (described in Section 3.3).
- CSFS [Hindawi et al., 2011] (described in Section 3.4).

Since wCKM provides a partition with relaxed constraint satisfaction, other methods are considered for comparison over constrained clustering. Theses methods are :

- COP-KMeans, which performs hard constraint satisfaction in k-means algorithm [Wagstaff et al., 2001].
- MPC-KMeans [1] , is a hybrid approach, performing both soft constraint satisfaction and metric learning [Bilenko. et al., 2004].

4.4.2 Experimental setup for wCKM

To simulate the “small labeled-sample” context, we set l, the number of labeled data, by randomly selecting 3 instances per class and the remaining instances

[1] The code for this method can be found at (http://www.cs.utexas.edu/users/ml/risc/code/).

are used as unlabeled data. The parameter β in wCKM is always set to -0.1[2]. The obtained feature weights are averaged over 10 runs with different initializations of centroids and ranked in a descendant order for selecting the relevant ones. For the other compared methods we respect the same parameters taken by the associated authors.

After feature selection, a linear SVM classifier (using LIBSVM package [Chang and Lin, 2011]) is employed for classification accuracy. Each dataset is split (in a stratified way) into a training partition with 50% of the instances and a test partition with the remaining 50% of instances.

In addition, we evaluate the clustering accuracy by comparing the obtained label of each instance with that provided by the data corpus. For that, we use Rand index [Rand, 1971] to measure the clustering quality.

4.4.3 Validation of feature selection on "Wave" dataset

In this section, we present the result of applying wCKM over the "Wave" dataset. These results are presented in Figure 4.1. We can see in Figure 4.1(a) that the features (22 to 40) have low values on their weights, so the noise represented by these features is clearly detected.

Figure 4.1(b) illustrates the convergence curve of wCKM' algorithm. The horizontal axis represents the number of iterations and the vertical axis represents the number of changed cluster-membership values (u) during the clustering process. We can see that the algorithm converges rapidly until a local minimal value is reached. The final set of weights (w) is obtained after a few number of iterations ($t = 19$).

In Figure 4.1(c), we show the curves of constraint violations (ML, CL and ML+CL) during the process (vs. the number of iterations). The three curves decrease with the convergence of the algorithm. This means that the violation of the constraints also decreases along the minimization of the objective function.

[2]This value of β is chosen after several experiments.

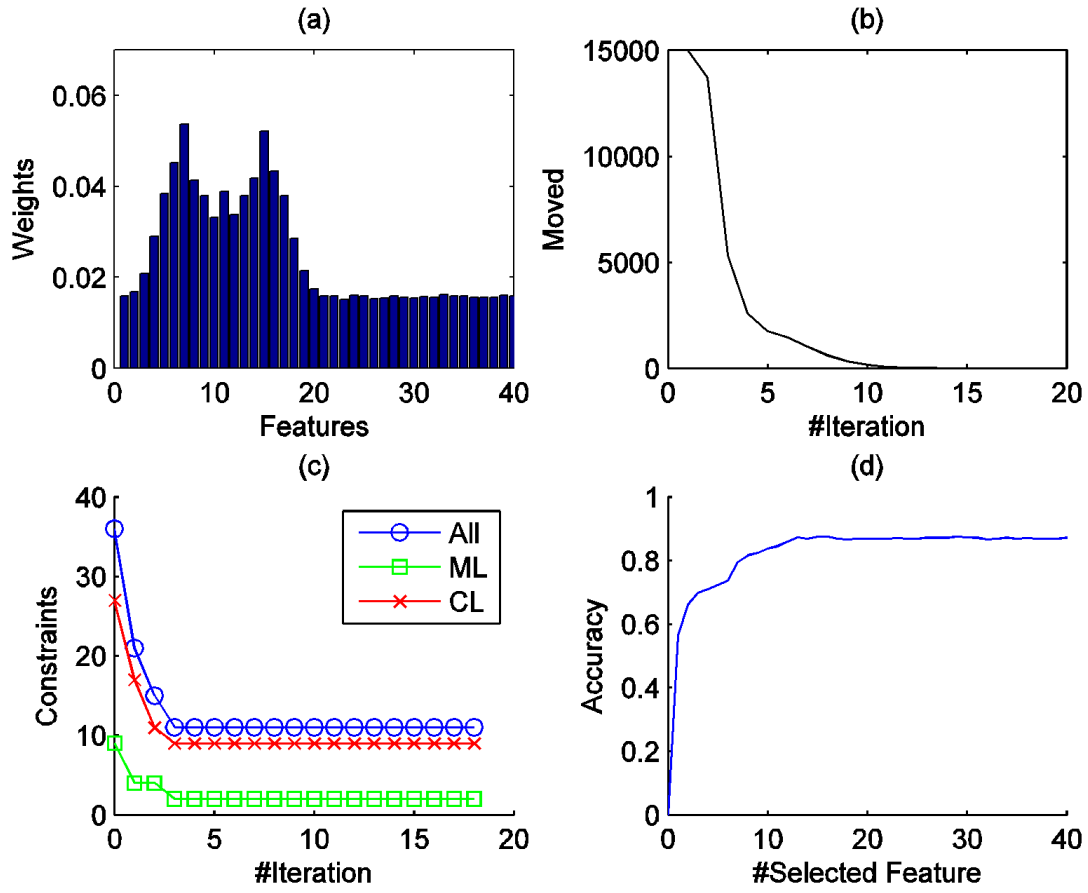


Figure 4.1: Results of wCKM on "Wave" dataset. (a) Feature weights. (b) Convergence curve. (c) Constraint violation. (d) Classification accuracy.

In fact, the algorithm tends to find a compromise between the minimization of the distance between instances and their cluster-centroids on one hand; and the minimization of the constraint violation on the other hand. At the end, both assignment and constraint satisfaction are made in a soft manner.

Figure 4.1(d) presents the classification accuracy vs. different number of selected features. The curve increases steadily and rapidly over the first twenty features whose weights are the high ones in Figure 4.1(a); and it stabilizes over the remaining features that are irrelevant (with low weights). This means that the method is robust to noise.

4.4.4 Comparison of feature quality on high-dimensional data

In this section, we assess the performance of wCKM and compare it with the above cited methods. The comparison is conducted by measuring both classification and clustering accuracies.

Figure 4.2 plots the curves of the whole algorithms (except cFLD and SSDR which do not do feature selection) for classification accuracy vs. different number of selected features. This figure indicates that in most cases wCKM outperforms the other methods, especially for text datasets in which the noise is important. It can be shown that in particular, the performances of SC4 is the worst. The performance of SC4 is weak for small-labeled data and relatively well for the high-labeled ones. We estimate that this is because SC4 naively combines (by multiplying) two scores from both labeled and unlabeled data. wCKM seems to combine more efficiently the labeled and unlabeled parts of data than the other constraint based semi-supervised methods. This shows that the combination is more efficient using an embedded approach than that using a filter one, in which the relevance of features is independently measured. However, wCKM does not perform very well for Face datasets, in which the number of clusters and the number of generated constraints are both high. This is because the algorithm tends to simultaneously optimize both the proximity between instances and their closest centroids on one hand, and the violation of constraints on the other hand. Moreover, It is worth mentioning that the classification accuracy of wCKM generally increases at the beginning (with a small number of features), but such increase lessens at the end.

Table 4.1 compares the averaged accuracy under different numbers of selected features. From this table and Figure 4.2, we can find that, the performance of wCKM is almost always better than that of sSelect, SC4 and CLS, and is comparable with CSFS. More specifically, wCKM is superior to the other methods on all datasets except those with high values of both K and l (as in Image datasets).

In addition, from Table 4.1, we can calculate de differences of the averaged

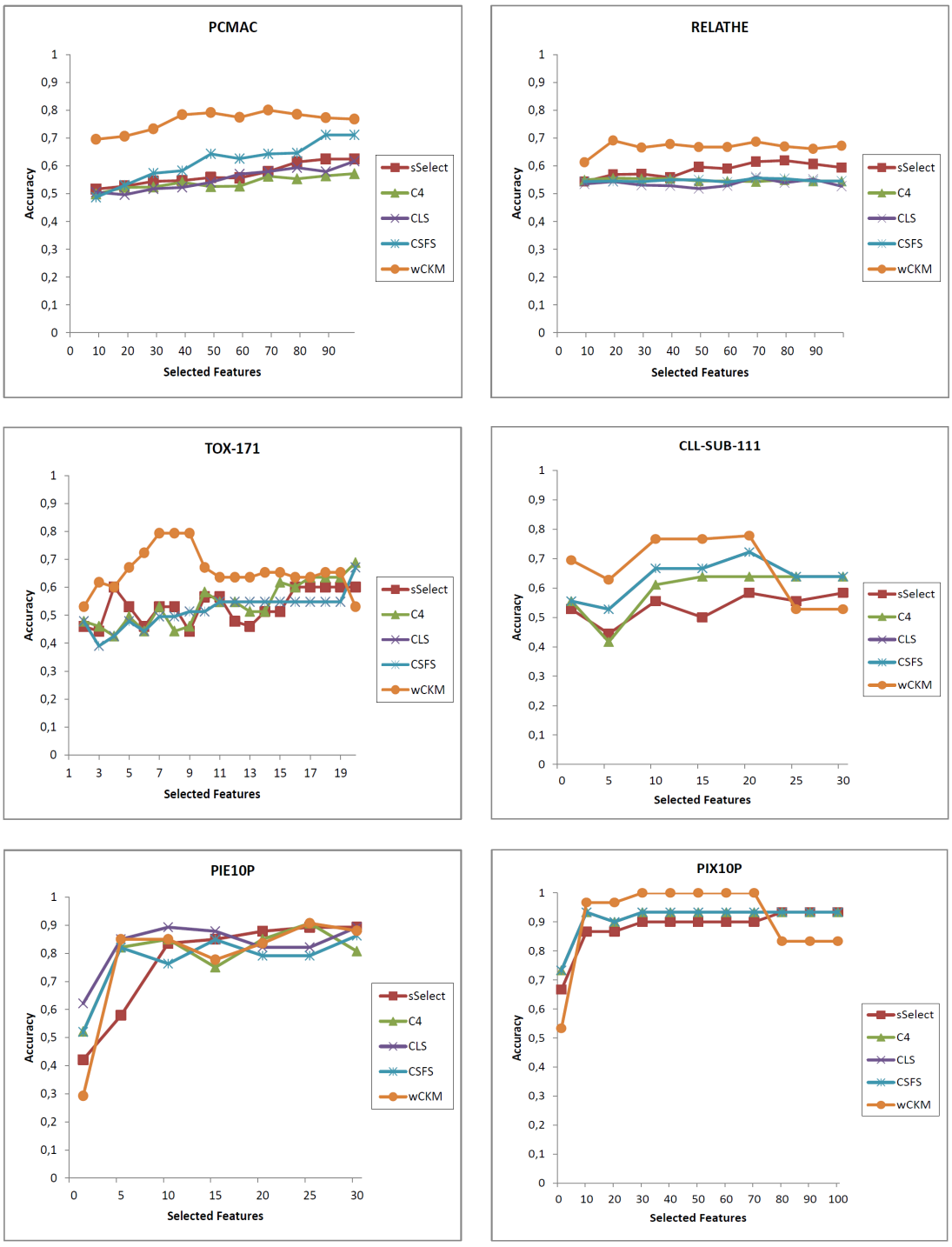


Figure 4.2: Performance on classification accuracy vs. different number of selected features

accuracies among algorithms. We can see that in terms of accuracy gains, wCKM is 10.26% better than sSelect, 8.59% better than SC4, 7.82% better than CLS and 7.565% better than CSFS. This observation suggests that the label information is more adopted for semi-supervised feature selection with our method than the other ones. This is also consistent with our understanding that the *emdedded* character of the method has an important role for feature selection comparing to the filter based approaches.

Indeed, with wCKM, the label information is explicitly learned by the minimization of constraint violation in the associated objective function. This minimization is simultaneously performed with the minimization of the weighted distance between data instances and their closest prototypes in the different clusters. Both minimizations are required for providing both, relevant features and an efficient constrained clustering.

Furthermore, we compare the performance between all methods when different levels of supervision are used. Figure 4.3 shows the plots for accuracy under desired number of selected features versus different number of constraints. The desired number of selected features is fixed to 10. Here, cFLD and SSDR are included in the comparisons as they represent dimensionality reduction methods. A particular study on Figure 4.3 reveals that, generally, the accuracy of wCKM increases steadily in the beginning when the number of constraints is limited, and decreases at the end. It implies that only a limited supervision is required for wCKM to provide high performance. This corresponds exactly to our initial problem concerning "small labeled-sample" data. To show how the dimensionality of the projected space affects the performance, we com-

Table 4.1: Classification Accuracy (in %).

Datasets	sSelect	SC4	CLS	CSFS	wCKM
PCMAC	56.3±3.75	53.68±2.41	55.26±3.71	60.55±7.45	**75.9±4.96**
RELATHE	58.26±2.52	54.77±0.51	53.79±1.19	55.02±0.47	**66.46±3.05**
TOX-171	53.16±5.87	53.99±7.73	51.78±5.91	51.78±5.91	**65.90±7.25**
CLL-SUB-111	52.03±4.66	59.44±8.55	64.44±6.97	64.9±6.01	**69.61±5.4**
PIE10P	76.43±17.35	81.53±14.06	**82.55±8.80**	77.10±10.74	77.04±19.85
PIX10P	89.8±3.63	92.56±2.95	92.76±2.5	**92.77±2.6**	92.6±1.11
WIN	0	0	1	1	**4**

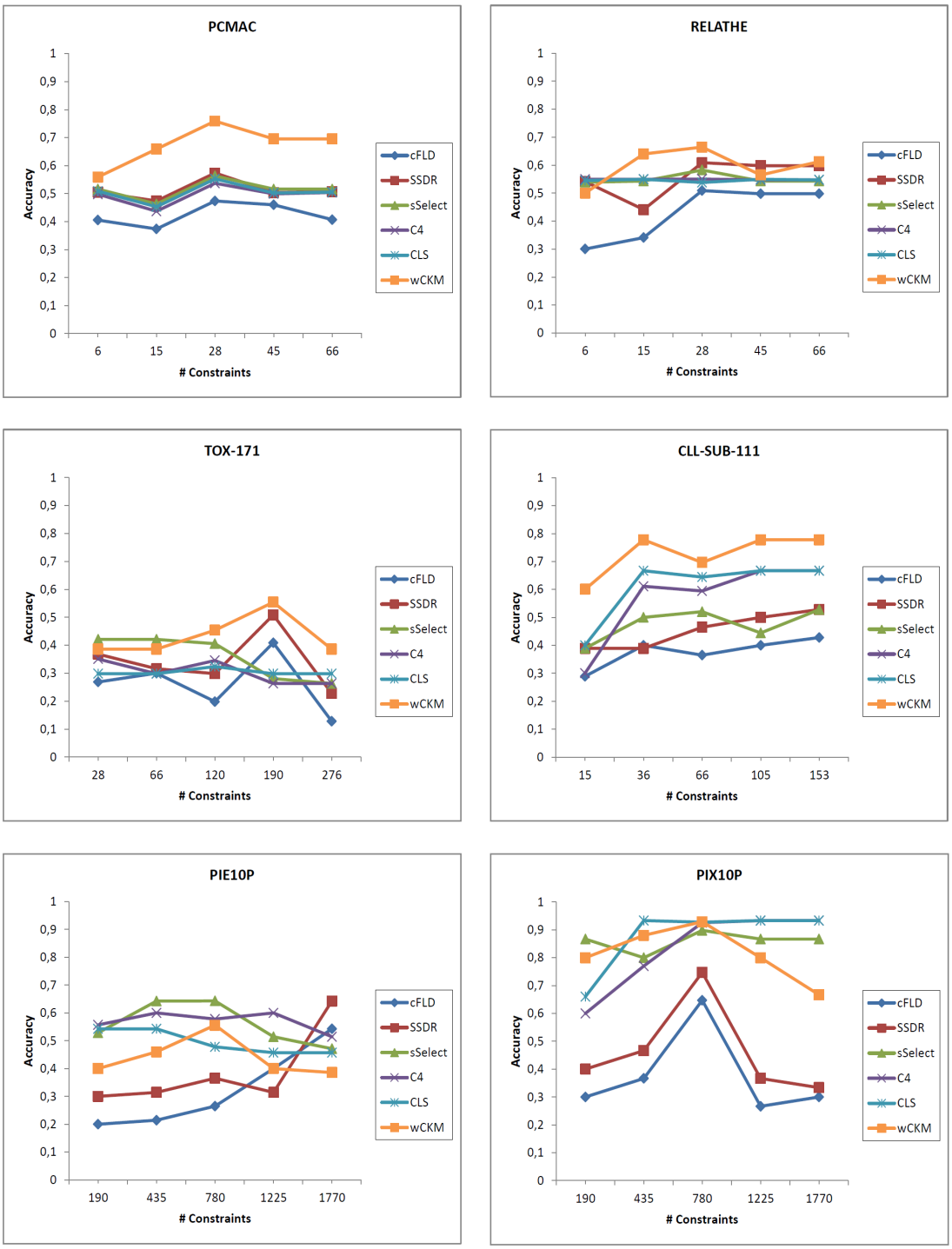


Figure 4.3: Classification accuracy vs. different number of constraints.

Table 4.2: Clustering Accuracy (Rand index in %) with the ten best selected features.

Datasets	cFLD	sSelect	SSDR	SC4	CLS	CSFS	wCKM
PCMAC	48.8±0.03	49.9±0.02	50.0±0.01	50.1±0.01	50.3±0.02	49.9±0.01	**52.05±0.24**
RELATHE	48.9±0.02	50.2±0.2	50.3±0.01	50.4±0.1	50.4±0.01	50.4±0.01	**52.2±2.1**
TOX-171	60.0±0.3	63.1±0.75	61.3±0.26	61.6±1.59	61.5±1.74	61.9±1.72	**66.4±1.36**
CLL-SUB-111	50.9±1.4	54.8±1.31	51.1±3.39	54.9±2.47	53.8±2.37	54.7±2.7	**57.08±0.1**
PIE10P	80.6±1.2	82.5±2.05	81.5±1.4	81.3±1.49	82.4±1.03	**82.7±0.84**	81.6±0.93
PIX10P	84.2±1.36	91.1±2.92	84.6±1.3	90.5±2.71	92.1±3.19	**95.3±3.31**	95.3±2.19
WIN	0	0	0	0	0	2	**4**

Table 4.3: Performance of wCKM vs. two known constrained clustering algorithms.

Datasets	COP-Kmeans Unc / Con	MPC-Kmeans Unc / Con	wCKM Unc / Con
PCMAC	49.98 50.04	49.97 49.98	49.98 **51.38**
RELATHE	50.40 50.68	50.32 50.39	50.41 **51.66**
TOX-171	63.38 64.75	64.02 65.04	63.39 **65.9**
CLL-SUB-111	52.60 55.32	54.46 55.79	55.32 **56.34**
PIE10P	70.12 79.90	78.40 **81.62**	70.93 80.39
PIX10P	87.80 90.14	88.48 92.20	90.54 **94.63**

pare the clustering accuracies with a fixed number of selected features (the same as above). The percentage of supervision is the same as indicated in Table 3.3. Since our proposal is based on k-means paradigm, we can obviously calculate its clustering accuracy. For the other methods, we use their results from Table 3.5, and for PIX10P dataset, we perform k-means algorithm in the selected feature subspace. The clustering is repeated 20 times with different initializations and the best result in terms of the objective function is recorded in Table 4.2.

As can be noted, wCKM is very competitive with the other algorithms. For example, it performs much better than cFLD or SSDR for dimensionality reduction, when the number of constraints is limited. This indicates that the semi-supervised feature selection achieved by wCKM is capable of enhancing clustering performance, which is provided by the same algorithm.

4.4.5 Results on constrained clustering

In this section, we present some comparisons of wCKM vs. two known constrained clustering algorithms, COP-Kmeans and MPC-Kmeans. These comparisons were done without feature selection, and are presented in Table 4.3. We compare the results for each algorithm in terms of its unconstrained and constrained performance, when provided with the constraints exacted from the labeled part of data according to the last column of Table 3.3. We evaluated these algorithms on the datasets with all their features, since the objective here is not feature selection but to show the performance that can provide the proposal on constrained clustering when the features are weighted as explained previously. Table 4.3 shows the accuracy (Rand index) on the held-out test sets which are subsets of data composed of instances that are not directly or transitively affected by the constraints.

On the one hand, wCKM provides a clear improvement to clustering accuracy, despite the violation of some constraints. On the other hand, the results obtained by wCKM are similar and sometimes better than the other constrained clustering methods. The most important remark is that with our proposal the clustering performance increases significantly with a few number of constraints comparing to other ones. For example, in Table 4.3 , for "PIE10P", wCKM (80.39%) is not better than MPC-Kmeans (81.62%) but wCKM yields an improvement of 9.46% over the baseline while MPC-Kmeans achieves 3.22% increase in accuracy.

4.5 Results of L2GFS

In this section, we present an experimental study for L2GFS against wCKM in order to position each one regarding the other. By the end of this section, we will present different ways of constraint integration in the proposed approach, and we will show how they affect considerably the quality of selected features. In these experiments, we use the same configurations as in the experiments of wCKM. This includes: the same datasets, the same parameter values, and the

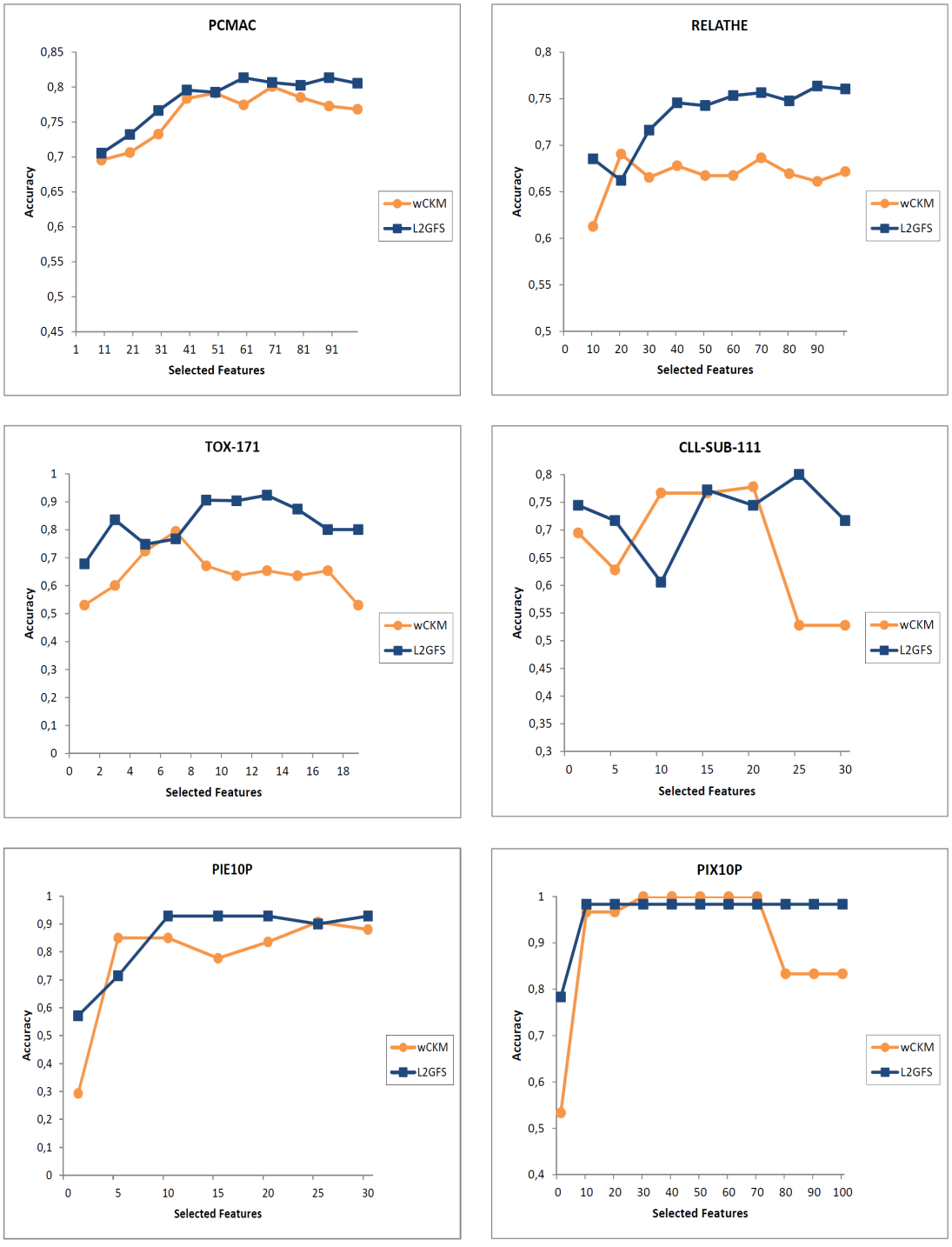


Figure 4.4: Classification accuracy vs. different number of selected features

same evaluation measures.

Figure 4.4 plots the curves of both algorithms for classification accuracy vs.

different number of selected features. This figure indicates that, in most cases, L2GFS outperforms wCKM or achieves a similar performance. Although wCKM was practically proven to effectively integrate the pairwise constraints into feature selection. L2GFS seems to better benefit from information supplied by these constraints. In addition, we showed how wCKM outperforms some well-known semi-supervised feature selection methods, then it is expected that L2GFS can achieve a similar (or even better) performance if compared with them.

Table 4.4 compares the averaged accuracy under different numbers of selected features. From this table and Figure 4.4, we can find that the performance of L2GFS is almost always better than that of wCKM. This can be explained by the fact that L2GFS drives the learning by the constraint preserving. These constraints are generated from labeled data in our case. Therefore, the classification ability in features selected by L2GFS might be better than those selected by wCKM. Note that wCKM relies on locality preserving while trying to minimize the violation of constraints.

Table 4.4: Classification Accuracy (in %).

Methods	PCMAC	RELATHE	TOX-171	CLL-SUB-111	PIE10P	PIX10P
wCKM	75.9±4.96	66.46±3.05	64.3±1.02	69.61±5.4	77.04±19.85	92.6±1.11
L2GFS	**80.78±3.54**	**73.89±4.46**	**82.38±0.94**	**72.91±3.9**	**86.29±13.25**	**93.52±2.75**

Furthermore, we compare the performance between the two methods when different levels of supervision are used. Figure 4.5 shows the plots for accuracy under desired number of selected features versus different number of constraints. The desired number of selected features is fixed to 10. A particular study on Figure 4.5 reveals that, generally, the accuracy of L2GFS is more stable while increasing the number of constraints. In fact, this is expected because wCKM is flexible with constraint violation (though the goal of the method is to minimize this violation). Such violation is susceptible to worsen when increasing the number of constraints.

To show how the dimensionality of the projected space affects the performance,

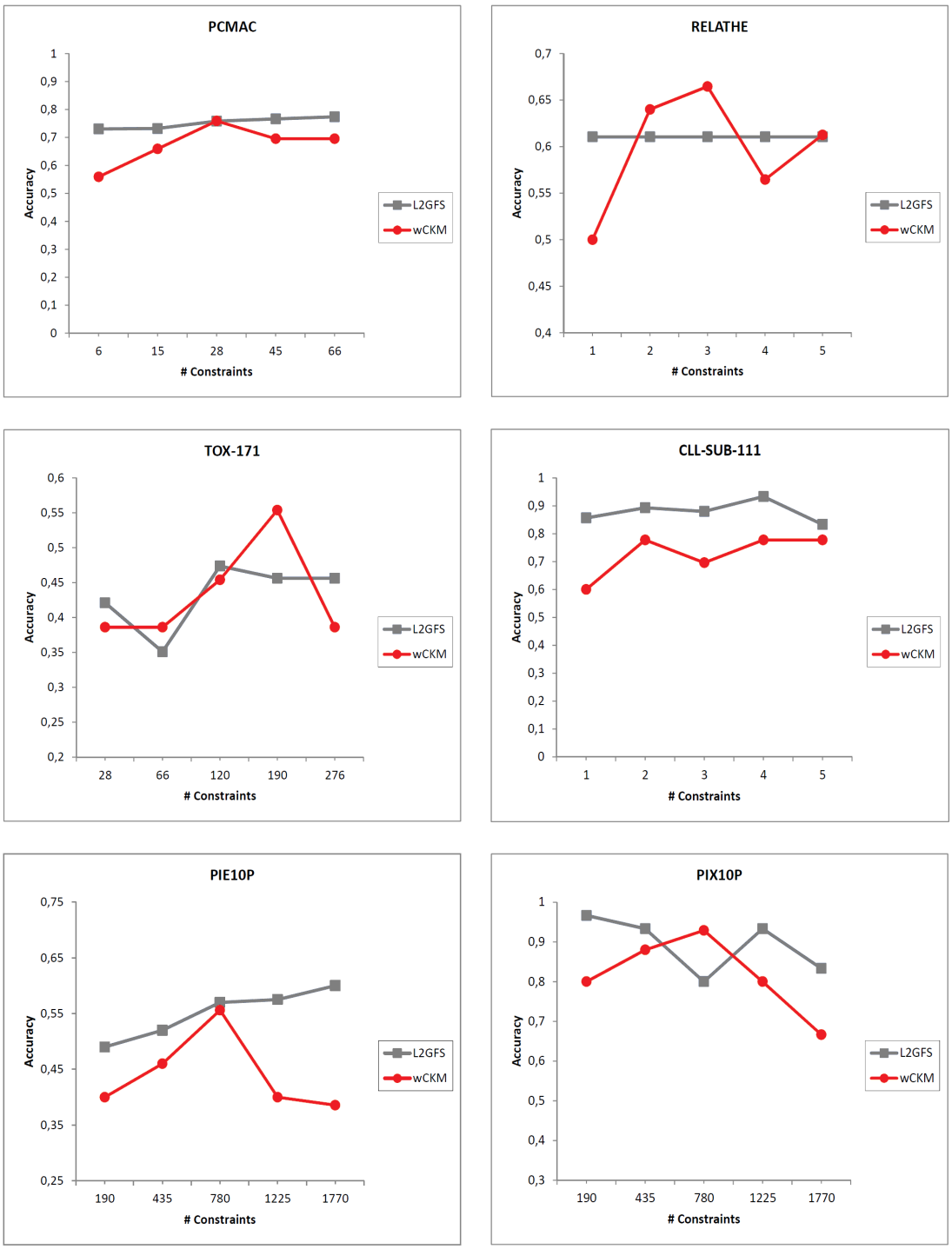


Figure 4.5: Classification accuracy vs. different number of constraints.

we compare the clustering accuracies with a fixed number of selected features (the same as above). The percentage of supervision is the same as indicated in Table 3.3. Since wCKM and L2GFS are based on k-means paradigm, we can obviously calculate their clustering accuracies. The clustering is repeated 10 times with different initializations and the best result in terms of the objective function is recorded in Table 4.5.

Table 4.5: Clustering Accuracy (Rand index in %).

Methods	PCMAC	RELATHE	TOX-171	CLL-SUB-111	PIE10P	PIX10P
wCKM	**52.05±0.24**	**52.2±2.1**	**66.4±1.36**	57.08±0.1	81.6±0.93	**95.3±2.19**
L2GFS	49.98±0.00	50.34±0.13	57.81±2.13	**62.91±3.9**	**81.73±1.46**	90.02±1.55

As can be noted, in major cases, wCKM records better results than L2GFS. This might be explained by the fact that wCKM can assess the locality preserving ability of features better than L2GFS. In fact, the objective function of wCKM prioritizes the data structure, while in L2GFS the constraint preservation is prioritized. Therefore, we can specify that L2GFS is more suitable for feature selection if constraints are guaranteed to be useful and noise-free (this might be achieved by constraint selection or any other techniques). Typically, in semi-supervised data, the size of labeled data is small and labels (or constraints) are well selective. Hence, we believe that L2GFS might be more convenient than wCKM in order to cope with the “small labeled-sample” problem. However, wCKM might be more adequate in cases where labeling process (or constraints acquisition) is not confident.

4.6 Conclusion

In this chapter, we proposed two weighting-based approaches for semi-supervised feature selection. Both approaches are integrated with the well-known k-means algorithm, by adding the feature weighting principle to the objective function. The first approach is wCKM, in which we utilized a "fuzzy" version of k-means where the assignment of examples is done in *soft* manner (i.e. an instance belongs to all clusters but with different scores). The other approach

is L2GFS, in which we adopted the *hard* assignment (an instance belongs to only one cluster). In wCKM, the constraints are added directly to the objective function, and the algorithm is permitted to violate constraints with a penalty. Thus, wCKM approach might be more robust towards labeling error. The objective in this case is to minimize this penalty by minimizing the constraint violation. In L2GFS, the constraints are added indirectly as a condition to which the objective function is subjected. As a result, all constraints are certain to be preserved, and no violation is permitted. In addition, wCKM is a global approach, where a feature has to be "good" in describing all clusters in order to be selected. While L2GFS is a local-to-global approach, in which a feature has more chance to be selected if it describes certain clusters but not all ones. The empirical results over high-dimensional benchmarking datasets proved the efficiency and effectiveness of the proposed approaches.

5 Conclusion and Perspectives

In this thesis, we presented different approaches for handling the problems of feature selection, which is one of dimensionality reduction strategies. In special, we studied the problem in semi-supervised paradigm. We reviewed the literature of dimensionality in general, which consists of feature extraction and feature selection techniques. We illustrated some of key methods in both domains. Being motivated in presenting feature selection, we focused more on feature selection methods which could be roughly divided in supervised, unsupervised and semi-supervised. We reviewed the representative supervised methods which depend on correlation with class labels for determining feature relevance. Then, we also illustrated the representative methods in unsupervised feature selection which is considered as a much harder problem due to the absence of labels, we viewed how methods in this domain try to investigate in the intrinsic properties of data, and evaluate the relevance of a feature regarding its ability in preserving certain locality properties.

In addition, we showed how the task of feature selection became more challenging with the so-called “small labeled-sample” problem, in which the amount of data that is unlabeled could be much larger than the amount of labeled data. Indeed, for such problem the traditional supervised and unsupervised feature selection methods are not convenient. On the one hand, supervised feature selection algorithms require a large amount of labeled

training data. On the other hand, unsupervised feature selection algorithms ignore label information, thus may lead to performance deterioration. For all these reasons, the usefulness of semi-supervised feature selection is more adapted and its effectiveness has been demonstrated.

Moreover, we demonstrated how the supervision information offered by the labeled part of data could be transformed into background knowledge to be integrated into the feature selection process, along with the geometric structure exploited from the unlabeled part of data. Such background information is generally expressed by pairwise constraints, that specify if two instances are to be in the same class when both have the same label (must-link constraint), otherwise in different classes if not (cannot- link constraint). Then, we reviewed the state-of-the-art of semi-supervised feature selection methods, and we illustrated in details several recent works which have attempted to exploit pairwise constraints or any other prior information in feature selection.

In order to tackle the problem of semi-supervised feature selection, we presented several approaches in both filter and embedded forms. In filter approaches, we first proposed CLS score in which we tried to compromise between the information presented by labeled part of data, and structure properties presented by the unlabeled part. Then the exploitation of both parts helped in improving the performance over the other competitive methods. This was expected because the importance of constraints is practically proven. Nevertheless, and unlikely to what might be expected, some constraints can decrease the learning performance.

To overcome the effects of noisy constraints, we tried to solve the problem by the exploitation of a constraint selection procedure which resulted in more useful constraint set to be presented to the data. Then, we proposed a more specific framework (CSFS) which achieved a considerable performance over its ancestor CLS.

Another enhancement over feature relevance is the redundancy elimination. In this sense we illustrated the existing approaches which treat redundancy in feature selection, then we proposed a feature selection score with graph-based redundancy elimination (CSFSR). The experimental results showed that eliminating redundant features can considerably improve the learning process.

In addition to filter methods, we presented two embedded approaches for feature selection (wCKM) and (L2GFS). Both methods modify the original k-means objective function, and extend it for semi-supervised feature selection. In wCKM, we integrated the pairwise constraints directly in the objective function. The algorithm proceeds in a soft manner and penalizes the constraint violation. In L2GFS, we applied a hard fashion of constraint integration, and the execution of k-means algorithm is done while respecting the non-violation of any constraint. In both methods, we developed a weighting approach over constrained k-means. The difference between them is that the former approach is global and selects the relevant features over all data instances, while the latter first selects the relevant features to each cluster locally, this gives more chance for features that best select certain clusters but not all, then the global relevance of feature is calculated over the whole data instances. Empirical results were presented in both methods which proved the efficiency of the underlying algorithms.

Finally, the approaches presented in this thesis are not exempt from limitations. In addition, the proposed approaches inspired us important avenues for future works. These works include but are not limited to:

- Adapting the proposed methods for dealing with very high-dimensional datasets (with hundred of thousands of features).
- The pairwise constraints are not the only type of constraints that might exist, an interesting work might be the investigation of other constraint types. In addition, the measure which we adopted is independent from the learning algorithm. There exist other measures that could be adopted

in order to evaluate the utility of the constraint set.

- The k-means algorithm is well known, we tried to review its semi-supervised versions (constrained k-means approaches). However, we believe that the extension of these approaches to the self-organizing maps might be an interesting work for semi-supervised feature selection.

- In CSFS, with constraint selection, we have coped with the problem of inefficiency in pairwise constraints generated from data. Typically, in semi-supervised data, labels are relatively few, so the number of generated constraints is rather small. This explains why the noise in these constraints has an important effect over the quality of selected feature. In CSFS, we tried to overcome this noise by constraint selection. A possible avenue may be to tackle this problem by ensemble-based approach, then the bagging of constraints can create a diversity, and thus improving the constraint-based learning performance.

- An interesting direction is to investigate how our methods can be extended to deal with regression problems in which the classes contain continuous values instead of categorical labels.

A Appendix A: List of Publications

- Efficient Semi-supervised Feature Selection: Constraint, Relevance and Redundancy (under peer reviewing).
- M. Hindawi, K. Benabdeslem. Une approche embedded pour la sélection de variables en mode semi-supervisé. **SFC12**, pages 29-31, Marseille, Octobre, 2012.
- M. Hindawi, K. Allab, and K. Benabdeslem. Constraint selection based semi-supervised feature selection. In Proceedings of **ICDM**. IEEE International Conference on Data Mining, pages 1080–1085, 2011.
- K. Benabdeslem, M. Hindawi. Constrained Laplacian score for semi-supervised feature selection, In the proceedings of **ECML/PKDD**, LNAI 6911, pages 204-218, 2011.
- M. Hindawi, K. Benabdeslem. Un score Laplacien sous contraintes pour la sélection de variables en mode semi-supervisé. Journées "Fouille de Données Complexes et de Grands Graphes (**FDC - FGG**)", pages 20-21, Paris, Juin, 2011.
- M. Hindawi, L. Morel, R. Aubry, et J.-L. Sourrouille (2008). Description and implementation of a UML style guide. In M. R. V. Chaudron (Ed.), Volume 5421 of **LNCS**, pp. 291–302. Springer.
- Sourrouille J.-L., Hindawi M., Morel L., Aubry R., Specifying consistent

subsets of UML, Educator symposium (co-located with **Models'08**), Warsaw University of Technology, Toulouse, France, Warsaw University of Technology , pp. 26-38.

Bibliography

[Allab and Benabdeslem, 2011] K. Allab and K. Benabdeslem. Constraint selection for semi-supervised topological clustering. In *Proceedings of the Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011*, pages 28–43, 2011.

[Alon et al., 1999] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Natl Acad Sci*, 96(12):6745–6750, 1999.

[Bar-Hillel et al., 2005] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research*, 6:937–965, 2005.

[Barkia et al., 2011] Hasna Barkia, Haytham Elghazel, and Alex Aussem. Semi-supervised feature importance evaluation with ensemble learning. In *Proceedings of the 11th International Conference on Data Mining*, ICDM '11, pages 31–40. IEEE Computer Society, 2011.

[Basu et al., 2002] Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. Semi-supervised clustering by seeding. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, pages 27–34. Morgan Kaufmann Publishers Inc., 2002.

[Belkin and Niyogi, 2002] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems*, 2002.

[Bellal et al., 2012] Fazia Bellal, Haytham Elghazel, and Alex Aussem. A semi-supervised feature ranking method with ensemble learning. *Pattern Recognition Letters*, pages 1426–1432, 2012.

[Benabdeslem and Hindawi, 2011] K. Benabdeslem and M. Hindawi. Constrained laplacian score for semi-supervised feature selection. In *Proceedings of the Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011*, pages 204–218, 2011.

[Bezdek, 1981] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Kluwer Academic Publishers, Norwell, MA, USA, 1981.

[Bilenko. et al., 2004] M. Bilenko., S. Basu, and R.J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the Twenty-First International Conference on Machine Learning ICML '04*, pages 11–18, 2004.

[Bishop, 1995] CM. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[Blum and Mitchell, 1998] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, COLT' 98, pages 92–100. ACM, 1998.

[Brazma and Vilo, 2000] Alvis Brazma and Jaak Vilo. Gene expression data analysis. *FEBS Lett*, 480:17–24, 2000.

[Breiman, 2001] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[Chang and Lin, 2011] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[Chapelle et al., 2006] O Chapelle, B. Sch"olkopf, and A. Zien. *Semi-supervised learning.* The MIT Press, 2006.

[Chung, 1997] F. Chung. Spectral graph theory. *AMS*, 1997.

[Cormen et al., 2001] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[Dash and Liu, 1997] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1:131–156, 1997.

[Davidson and Ravi, 2005] Ian Davidson and S. S. Ravi. Clustering with constraints: Feasibility issues and the k-means algorithm, 2005.

[Davidson et al., 2006] Ian Davidson, Kiri L. Wagstaff, and Sugato Basu. Measuring constraint-set utility for partitional clustering algorithms. In *In: Proceedings of the Tenth European Conference on Principles and Practice of Knowledge Discovery in Databases PKDD '06*, pages 115–126. Springer, 2006.

[Dempster et al., 1977] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[Ding and Peng, 2003] C. Ding and H.C. Peng. Minimum redundancy feature selection from microarray gene expression data. In *Proceedings of IEEE Computational Systems Bioinformatics Conference*, pages 523–528, 2003.

[Duda et al., 2000] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley Interscience, 2000.

[Dy and Brodley., 2004] J.G. Dy and C. E. Brodley. Feature selection for unsupervised learning. *Journal of Machine Learning Research*, (5):845–889, 2004.

[Fisher, 1987] D. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine learning*, 2:139–172, 1987.

[Fisher, 1936] R. Fisher. The use of multiple measurements in taxonomic problems. *Annals Eugen*, 7:179–188, 1936.

[Frank and Asuncion, 2010] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[Gilpin and Davidson, 2011] S. Gilpin and I. Davidson. Incorporating sat solvers into hierarchical clustering algorithms - an efficient and flexible approach. In *Proceedings of of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1136–1144, 2011.

[Golub et al., 1999] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M. Loh, L.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science 15*, 286(5439):531–537, 1999.

[Green et al., 1990] P.E. Green, F.J. Carmone, and J. Kim. A preliminary study of optimal variable weighting in k-means clustering. *Journal of Classification*, 7: 271–285, 1990.

[Gu et al., 2011] Q. Gu, Z. Li, and J. Han. Generalized fisher score for feature selection. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence UAI '11*, pages 266–273, 2011.

[Guan et al., 2011] Y. Guan, JG. Dy, and MI. Jordan. A unified probabilistic model for global and local unsupervised feature selection. In *Proceedings of the Twenty-Eight International Conference on Machine Learning ICML '11*, 2011.

[Guyon and Elisseeff, 2003] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, (3):1157–1182, 2003.

[He and Niyogi, 2004] X. He and P. Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems*, 2004.

[He et al., 2005] X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems, 17*, 2005.

[Hindawi et al., 2011] M. Hindawi, K. Allab, and K. Benabdeslem. Constraint selection based semi-supervised feature selection. In *Proceedings of 11th IEEE International Conference on Data Mining, ICDM '11*, pages 1080–1085, 2011.

[Huang et al., 2005] J.Z. Huang, M.K. Ng, H. Rong, and Z. Li. Automated variable weighting in k-means type clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:657–668, 2005.

[Jain and Zongker, 1997] A. Jain and D. Zongker. Feature selection: Evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):153–158, 1997.

[Jolliffe, 2002] I. Jolliffe. *Principal Component Analysis*. Springer, 2002.

[Kalakech et al., 2011] M. Kalakech, P. Biela, L. Macaire, and D. Hamad. Constraint scores for semi-supervised feature selection: A comparative study. *Pattern Recognition Letters*, 32(5):656–665, 2011.

[Kira and Rendell, 1992] Kenji Kira and Larry A. Rendell. A practical approach to feature selection. In *Proceedings of the ninth international workshop on Machine learning*, ML92, pages 249–256, 1992.

[Klein et al., 2002] D. Klein, S. Kamvar, and C. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proceedings of the Nineteenth International Conference on Machine Learning ICML '02*, pages 307–314, 2002.

[Kohavi and John, 1997] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(12):273–324, 1997.

[Kohonen, 2001] T. Kohonen. *Self organizing Map*. Springer Verlag, Berlin, 2001.

[Kononenko, 1994] Igor Kononenko. Estimating attributes: analysis and extensions of relief. In *Proceedings of the Machine Learning: ECML-94, European Conference on Machine Learning*, ECML '94, pages 171–182, 1994.

[Kononenko et al., 1997] Igor Kononenko, Edvard Šimec, and Marko Robnik-Šikonja. Overcoming the myopia of inductive learning algorithms with relieff. *Applied Intelligence*, 7(1):39–55, January 1997. ISSN 0924-669X.

[Kullback, 1959] S. Kullback. *Information theory and statistics*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, 1959.

[Law et al., 2005] M. H. C. Law, A. Topchy, and A. K. Jain. Model-based Clustering with Probabilistic Constraints. In *Proceedings of the 2005 SIAM International Conference on Data Mining (SDM)*, pages 641–645, 2005.

[Law et al., 2004] Martin H. Law, Alexander Topchy, and Anil K. Jain. *Clustering with Soft and Group Constraints*, volume 3138. January 2004.

[MacQueen, 1967] J.B. MacQueen. Some methods for classification and analysis of multivariate observations. In *proceeding of the fifth symposium on Math, statistics ans probability, Berkley, CA, USA.*, pages 281–297, 1967.

[Makarenkov and Legendre, 2001] V. Makarenkov and P. Legendre. Optimal variable weighting for ultrametric and additive trees and k-means partitioning: Methods and software. *Journal of Classification*, 18:245–271, 2001.

[Mali and Mitra, 2003] Kalyani Mali and Sushmita Mitra. Clustering and its validation in a symbolic framework. *Pattern Recognition Letters*, pages 2367–2376, 2003.

[Modha and Spangler, 2003] D.S. Modha and W.S. Spangler. Feature weighting in k-means clustering. *Machine Learning*, 52:217–237, 2003.

[Peng et al., 2005] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1226–1238, 2005.

[Press, 2007] William H. Press. *Numerical recipes : the art of scientific computing*. Cambridge University Press, 3 edition, September 2007.

[Pudil et al., 1994] P. Pudil, J. Novovičová, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119–1125, November 1994. ISSN 0167-8655.

[Rand, 1971] W.M. Rand. Objective criteria for the evaluation of clustering method. *Journal of the American Statistical Association*, 66:846–850, 1971.

[Ren et al., 2008] J. Ren, Z. Qiu, W. Fan, H. Cheng, and Philip S. Yu. Forward semi-supervised feature selection. In *Proceedings of PAKDD Conference*, pages 970–976, 2008.

[Robnik-Šikonja and Kononenko, 2003] M. Robnik-Šikonja and I. Kononenko. Theoretical and empirical analysis of relief and relieff. *Machine Learning*, 53: 23–69, 2003.

[Roweis and Saul, 2000] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by local linear embedding. *Science*, (290):2323–2326, 2000.

[Saeys et al., 2007] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, September 2007. ISSN 1367-4803.

[Schölkopf et al., 1998] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, pages 1299–1319, 1998.

[Shental et al., 2003] N. Shental, A. Bar-hillel, T. Hertz, and D. Weinshall. Computing gaussian mixture models with em using equivalence constraints. In *Advances in Neural Information Processing Systems, 15*, pages 465–472, 2003.

[Song et al., 2012] L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt. Feature selection via dependence maximization. *Journal of Machine Learning Research*, 13:1393–1434, 2012.

[Sun and Zhang, 2010] D. Sun and D. Zhang. Bagging constraint score for feature selection with pairwise constraints. *Pattern Recognition*, 43(6):2106–2118, 2010.

[Vapnik, 1995] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[Wagstaff and Cardie, 2000] Kiri Wagstaff and Claire Cardie. Clustering with instance-level constraints. In *Proceedings of the Seventeenth International Conference on Machine Learning ICML*, pages 1103–1110, 2000.

[Wagstaff et al., 2001] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 577–584, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[Wang and Davidson, 2010] Xiang Wang and Ian Davidson. Flexible constrained spectral clustering. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 563–572, New York, NY, USA, 2010. ACM.

[Weston et al., 2003] J. Weston, A. Elisseff, B. Schölkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

[Xu et al., 2009a] Z. Xu, R. Jin, MR. Lyu, and I. King. Discriminative semi-supervised feature selection via manifold regularization. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1303–1308, 2009a.

[Xu et al., 2009b] Zenglin Xu, Rong Jin, Irwin King, and Michael R. Lyu. An extended level method for efficient multiple kernel learning. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1825–1832. Curran Associates, Inc., 2009b.

[Yu and Liu, 2003] L. Yu and H. Liu. Feature selection for high-dimensional data: a fast correlation-based filter solution. In *Proceedings of the Twentieth International Conference on Machine Learning ICML '03*, 2003.

[Yu and Liu, 2004] Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

[Zhang et al., 2007] D. Zhang, Z.H. Zhou, and S. Chen. Semi-supervised dimensionality reduction. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, 2007.

[Zhang et al., 2008] D. Zhang, S. Chen, and Z. Zhou. Constraint score: A new filter method for feature selection with pairwise constraints. *Pattern Recognition*, 41(5):1440–1451, 2008.

[Zhao and Liu, 2007a] Z. Zhao and H. Liu. Semi-supervised feature selection via spectral analysis. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, pages 641–646, 2007a.

[Zhao and Liu, 2007b] Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning ICML '07*, 2007b.

[Zhao and Liu, 2012] Z. Zhao and H. Liu. *Spectral Feature Selection for Data Mining*. Chapman and Hall-CRC Data Mining and Knowledge Discovery Series, 2012.

[Zhao et al., 2010] Z. Zhao, L. Wang, and H. Liu. Efficient spectral feature selection with minimum redundancy. In *Proceedings of AAAI Conference*, 2010.

[Zhao et al., 2011] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anaud, and H. Liu. Advancing feature selection research- asu feature selection repository. Technical report, University of Arizona, 2011.

[Zhao et al., 2012] Z. Zhao, L. Wang, H. Liu, and J. Ye. On similarity preserving feature selection. *IEEE Transactions On Knowledge And Data Engineering*, page to appear, 2012.