Modélisation de la réponse utilisateur à une campagne de publicité mobile

Faustine Bousquet

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITE DE MONTPELLIER

En Biostatistique

École doctorale I2S - Information, Structures, Systèmes

Unité de recherche UMR 5149 – Institut Montpelliérain Alexander Grothendieck

Modélisation de la réponse utilisateur à une campagne de publicité mobile

Présentée par Faustine Bousquet

Le 15 Décembre 2020

Sous la direction de Christian LAVERGNE et Sophie LEBRE

Devant le jury composé de

Laila BOUGERET, VP Product, TabMo	Invitée
Charles BOUVEYRON, Professeur, Université Côte d'Azur	Rapporteur
Nicolas DUFORET, Docteur en Data Science, SeqOne	Examinateur
Florence FORBES, Directrice de Recherche, INRIA Grenoble	Rapportrice
Christian LAVERGNE, Professeur, Université Montpellier 3	Directeur de thèse
Sophie LEBRE, Maitre de Conférence, Université Montpellier 3	Co-encadrante de thèse
André MAS, Professeur, Université de Montpellier	Président du jury

UNIVERSITÉ DE MONTPELLIER

Il était une fois la thèse....

Il me semble important de reprendre l'histoire à sa genèse, à cette époque où, toute juste diplômée, je m'étais jurée de ne pas poursuivre mes études en thèse. À cette époque, j'étais persuadée que ce n'était pas pour moi : je n'avais pas le profil ni les capacités de mener à bien un doctorat ! C'est donc dans cette logique là, que pendant deux années après mon diplôme de l'INSA de Toulouse, j'ai fait mes premières armes dans le monde du travail. Cette première expérience n'aura pas été un long fleuve tranquille. Professionnellement, cela n'a pas été très épanouissant, moralement non plus. J'ai vite déchanté de ce monde du travail que j'attendais de découvrir avec tant d'impatience. Mais, je ne suis pas rancunière et je me sens finalement reconnaissante de cette expérience puisqu'elle m'aura permis de mettre sur ma route des personnes extraordinaires qui sont aujourd'hui des amis et de la famille. Et puis, un beau jour (ou peut être une nuit), la vie a fait qu'une opportunité inattendue est venue jusqu'à moi : "Salut Faustine, ça te dirait de faire une thèse CIFRE chez TabMo ?" Évidemment, ma première réaction a été de décliner gentiment et poliment l'offre. Mais les jours passants et le mal être s'installant dans ce foutu premier job...l'idée a peu à peu fait son chemin dans ma tête. Je me suis donc autorisée à découvrir le projet, les potentiels encadrants, le contexte d'une thèse CIFRE (dont je ne connaissais ni les tenants ni les aboutissants)... autant de paramètres qui, après avoir pesé le pour et le contre des dizaines de fois, m'ont donné envie de finalement dire...OUI. Un OUI à la fois rempli d'excitation par cette nouvelle étape inattendue mais aussi rempli de stress que quelqu'un découvre l'imposture que je pensais être. C'est ainsi que tout a commencé pour moi, et pour ceux qui s'arrêteront ici dans ces remerciements (promis je ne vous en veux pas), sachez que pour rien au monde je ne regrette mon choix ! Bon, il est vrai qu'à certains moments au cours de ces 3 années, il m'est arrivé de penser que, jamais ô grand jamais, je n'arriverai jusqu'à ce moment : terminer la rédaction de ce manuscrit de thèse en écrivant mes remerciements. Mais ces moments de doute ont tous, tour à tour, été vite balayés à la fois par tout le positif que cette aventure scientifique m'a apportée et l'encadrement exceptionnel dont j'ai eu la chance de bénéficier.

Je vais donc essayer, à travers ces quelques (et longues) lignes, de remercier et exprimer toute ma gratitude à l'ensemble des personnes qui m'ont accompagnées au cours de ce doctorat.

À Christian & Sophie

On connait tous l'expression "le meilleur pour la fin", mais, exceptionnellement, je complèterai l'adage par "le meilleur AUSSI pour le début". Alors je tenais, en premier lieu, à remercier infiniment mes deux directeurs de thèse : **Sophie Lèbre et Christian Lavergne**. J'avais beaucoup entendu dire que le plus important dans la thèse, ce sont les encadrants. Et bien, c'est vrai, je vous le confirme. Dès notre première rencontre, votre bienveillance m'est apparue comme une évidence, et m'a immédiatement rassurée.

Christian, merci d'avoir accepté d'encadrer ma thèse, la dernière dans ton rôle de directeur de thèse et quel sujet original pour terminer ! Merci pour ta présence, tes conseils, ces longues réunions de travail où la bonne humeur et les rires ne nous ont jamais quittés. Merci aussi de m'avoir toujours poussée à franchir les étapes qui me faisaient peur (les conférences notamment) et d'avoir cru en moi. Et puis, je n'oublierai pas de si tôt tout ce que tu m'as appris que ce soit en statistiques, sur le monde de la recherche ou encore sur les chats de Noël. Je me souviendrai aussi de nos longues discussions sur les modèles mixtes où tu as eu la patience de m'expliquer et de me (re)(re)(re) expliquer ce qu'on entendait par "effets aléatoires". Mais ça y est, Christian, je crois que j'ai vraiment compris maintenant !

Sophie, je ne sais même pas par où commencer tant ton encadrement à tous points de vue a été sans faille. Merci pour ton soutien, ta patience, ta rigueur, ta bienveillance. Merci à toi

aussi d'avoir accepté le challenge d'un sujet de thèse bien loin de la biologie ! Ton investissement dans cette thèse a été incroyable, et je mesure ici totalement ma chance de t'avoir eue comme encadrante. Les séances de travail, nos réunions hebdomadaires comme nos échanges de mails réguliers et même nos visios (!) m'ont énormément appris et ont grandement contribué à mon épanouissement scientifique. Et, je dois l'avouer, tout cela va beaucoup me manquer ! Merci aussi pour les moments de vie et discussions diverses et variées que nous avons pu partager au cours de ces 3 années. Enfin, merci pour tes nombreuses relectures et tes conseils toujours avisés. Sans toi je n'aurais pas pu en arriver là, c'est certain !
Sophie, Christian, je m'arrête ici vous concernant, mais sachez que cette thèse sans vous, je ne sais pas si elle aurait valu le coup !

À mon jury

Je tiens à remercier **Florence Forbes et Charles Bouveyron** pour avoir accepté de rapporter mon travail. Je remercie aussi **André Mas** d'avoir accepté le rôle de président de jury et **Nicolas Duforet** celui d'examinateur.

À mes collègues de l'IMAG

Je salue toutes les personnes que j'ai pu croiser à l'IMAG. J'en profite pour remercier en particulier tous les doctorant.e.s pour leur sens de la convivialité. Je garde en mémoire les nombreux cafés avec vous dans la bonne humeur de la salle de pause. Même si je n'étais pas présente à 100% au laboratoire, vous avez toujours su m'accueillir chaleureusement. Alors merci **Thiziri, Zaineb, Benjamin, Julien**... je ne peux pas tous vous citer mais le cœur y est. Une pensée particulière à **Florent**, c'était un plaisir de croiser ta route. Merci pour ta gentillesse. Enfin, merci à **Tiffany**, ma compatriote de thèse CIFRE, celle avec qui j'ai pu partager toutes les étapes clés de cette thèse et qui m'a accueillie à bras ouverts dès le premier jour.

À mes collègues de TabMo

Je remercie **Julien** et **Laila** pour m'avoir aussi bien encadrée et épaulée sur la deuxième moitié de cette thèse. Merci aussi à celles et ceux qui vous ont précédés pour l'encadrement de cette thèse, *ils se reconnaitront*. Un pensée particulière pour **Chloé** : merci pour tout ce que tu m'as appris et la manière dont tu m'as accompagnée dans cette thèse. Et une pensée aussi pour **Nicolas**, si j'ai démarré cette thèse, c'est en grande partie grâce à toi et tes encouragements bienveillants.

Merci à TabMo, spécialement à ses cofondateurs **Renaud & Hakim**, pour m'avoir offert l'opportunité de faire cette thèse. J'ai pu passer 3 années entièrement consacrées à ce projet de recherche et je mesure complètement ma chance d'avoir eu cette liberté de travailler sur un sujet qui me tenait tant à cœur.
Merci à tous mes collègues de TabMo et plus particulièrement à ceux du bureau de Montpellier, quelle team de folie on fait ! Merci pour votre soutien au quotidien que ce soit pour mes différentes présentations mais aussi pour l'intérêt que vous avez porté à mes modèles statistiques.
Évidemment, je salue particulièrement la team "click-prédiction" qui a permis la mise en production et la réalisation de ce projet dans la joie et la bonne humeur. Merci les gars : **Michael, El Hassan, Cédric et Sami** de m'avoir épaulée et appris à devenir (un peu) une dev.
Merci à tous mes relecteurs assidus (Big **big** up **Sébastien** et **Julien** sans oublier **Paul, Julien C, Mag, Mel, Annabel**...et tous les autres !), qui ont découvert avec joie des pages entières de formules où se cachaient de belles fautes de français. Merci pour le temps que vous m'avez accordé, ce manuscrit c'est aussi un peu grâce à vous.
Enfin, spéciale dédicace à **Julien** le best chef, **Magali** la plus adorable des voisines et collègues de

bureau, **Mélou** la plus drôle, **El Hassan** mon acolyte Data, **CédriQ et SimonQ** merci pour ces fourires, votre bonne humeur, nos potins et restos improvisés et votre soutien *presque toujours* quotidien ! Et enfin, un grand merci à **Laila** aussi géniale en tant que collègue, responsable de thèse qu'en tant qu'amie.

À mes ami.e.s

Merci à mes ami.e.s qui depuis des années m'ont entendu dire que *"J'ai trop raté l'exam"*, *"Je stresse"* suivi de... *"moi après l'INSA j'arrête les études c'est SUR"* et qui m'ont soutenue avec bienveillance dans ce choix surprenant quelques années plus tard.
Merci **Daphné** d'être la meilleure des amies. Merci à la "Dream Team", **Manon, Mathilde et Marie**, pour votre soutien après tant d'années, le collège ça commence à dater ! Merci **Anne-Sophie** d'avoir toujours été là pour moi. Merci à ces amies incroyables que sont **Mérédith, ChA, Jessica, Laura, Camilloune, les "Bibis" Hélène & Lise** pour tous les inoubliables moments d'amitié partagés qui ont égayé ces trois dernières années (et celles d'avant). Mes pensées vont également à **Mathou, Salma, Linda, Boris et Zeliha**, je n'oublie rien ! Merci tout particulier à **François**, tu as été mon pilier à l'INSA et tu le restes encore aujourd'hui : je te remercie de m'avoir épaulée au quotidien. Je chéris chacune de nos discussions sans fin, que ce soit pour des questions de stats, pour relire ma thèse, pour avoir ton avis sur des questions de la vie (dark-green ou pas ?), pour se soutenir l'un l'autre, pour discuter de tweets rigolos ou encore refaire le monde qui nous entoure et qu'on ne comprend pas toujours sans oublier toutes nos joyeuses retrouvailles que ce soit à Berlin ou Montpellier !

À ma famille

Je finirai par remercier ma famille au grand complet, que ce soit les **Ribes**, les **Bousquet** et tous les autres...merci d'être la famille extra que vous êtes. Merci tout particulier à ma cousine **Charlotte**. Merci aussi aux membres de ma belle-famille pour leur gentillesse à mon égard. Merci à **Lexie** d'être un rayon de soleil au quotidien. On dit souvent que le rôle de "belle-mère" est ingrat mais nous devons être l'exception qui confirme la règle : c'est un bonheur de t'avoir dans ma vie. Merci à mes grands frères de m'avoir montré le chemin de la réussite : **Damien**, merci de m'avoir transmis ta passion pour les mathématiques et **Florian**, merci de croire en moi et de me montrer tout ce qu'on peut accomplir avec de l'ambition. Merci à **Natalia**, ma belle-soeur, d'être toujours si attentionnée envers moi. Merci à ma grand-mère **Juliette**, qui du haut de ses 97 ans m'offre chaque jour les plus belles leçons de vie. Enfin merci à mes parents : **Jocelyne & Philippe**. Je sais à quel point vous étiez inquiets pour moi au commencement de tout cela mais vous m'avez soutenue de la meilleure des manières durant ces 3 années. À l'image de votre soutien et votre affection depuis 28 ans. Je vous dois tout !

Je ne pourrais pas terminer ces remerciements sans adresser le plus grand et le plus beau des mercis à **Raphaël**. Tu as été présent sans relâche. Tu m'as soutenue et épaulée quand je perdais confiance en moi ou en mon travail. Tu as su être patient et me booster quand il le fallait ! Tu as fait de mon quotidien une parenthèse enchantée. Merci pour tout.

Résumé : La prédiction du taux de clics (CTR) est l'un des défis majeurs de la publicité en ligne au cours de ces dernières années. L'objectif de notre travail est de répondre à un encart publicitaire disponible via un système d'enchère en proposant la publicité la plus pertinente possible. En d'autres termes : il s'agit d'être en mesure de proposer la bonne publicité à la bonne personne au bon moment. Cet objectif prend en considération deux enjeux principaux. Le premier concerne la caractérisation des données à disposition qui sont de natures volumineuses, hétérogènes et clairsemées. Le second objectif concerne la mise en production du modèle : le modèle doit pouvoir être utilisé en temps réel et son déploiement doit être simple à mettre en œuvre. Nous introduisons ici une nouvelle méthode de prédiction du CTR qui repose sur un mélange de modèles linéaires généralisés (GLM). Nous développons tout d'abord une méthode de clustering basée sur un modèle prenant en considération l'aspect longitudinal (afin d'exploiter l'historique de chaque campagne) et non gaussien (la métrique d'intérêt est un taux) des observations du CTR dans les campagnes publicitaires. Cette étape préliminaire permet de grouper les campagnes ayant des profils similaires et offre ainsi une meilleure description des données. Le package R binomialMix disponible sur le CRAN implémente cette approche pour le mélange de données binomiales et longitudinales. Par la suite, en s'appuyant directement sur les clusters inférés, nous proposons un modèle prédictif qui permet de répondre au sujet central de notre problématique métier : estimer une probabilité de clic pour toute campagne en temps réel. Plusieurs modèles sont mis en compétition : des modèles naïfs et un modèle simple de GLM sont ainsi comparés à plusieurs modèles qui se basent sur les résultats du clustering. Deux modèles (parmi ceux qui utilisent les résultats du modèle de mélange) se distinguent par leurs performances prédictives. Des expérimentations menées sur données simulées et réelles ont montré l'importance de l'étape préliminaire de classification non supervisée sur la qualité de la prédiction. L'ensemble de ces étapes a ainsi pu être industrialisé et intégré dans le processus d'enchère déjà existant. Cette intégration est la succession d'un ensemble d'étapes : la récupération des données, leur prétraitement, l'estimation des paramètres du mélange à partir de variables explicatives soigneusement choisies et enfin, la mise en place du modèle prédictif. Un dernier travail a permis l'exploitation des prédictions à partir des probabilités de clic obtenues en sortie des modèles prédictifs. Ainsi, nous avons pu prédire le CTR en temps réel sur la plateforme d'enchère et pour chaque espace publicitaire disponible qui y transite. L'analyse des premiers résultats en production montre que, pour certains contextes d'enchère, l'utilisation du modèle prédictif, couplé à l'étape de clustering au préalable, a permis une amélioration significative du taux de clics.

Mots clés : publicité, taux de clics, GLM, classification non supervisée, modèle de mélange, prédiction

Abstract : Click through rate (CTR) prediction is one of the most important challenges in the advertising field over the last years. The objective of our work is to respond to an ad placement via an auction system and with the most relevant content for the person who sees it, at an optimal price. In other words : we want to be able to offer the right ad to the right person at the right time. This objective takes into consideration two main issues. The first concerns the characterization of the available data, which is voluminous, heterogeneous and sparse in nature. The second objective concerns the production of the model, which must be carried out via an easy-to-implement deployment and used in real time. We introduce here a new method of CTR prediction based on a mixture of Generalized Linear Models (GLM). We first develop a clustering method based on a model that takes into account the longitudinal (in order to exploit the history of each campaign) and non-Gaussian (the metric of interest is a rate) aspect of CTR observations. This preliminary step of unsupervised classification offers a better description of the data and allows to group campaigns with similar profiles. The binomialMix R package available on CRAN implements this approach for a mixture of binomial and longitudinal data. Subsequently, by relying directly on the inferred clusters, we develop a predictive model that enables us to address the central issue of our business : estimating the probability of clicks for all campaigns in real time. Several models are put in competition : two naive models and a simple GLM model are compared to several models based on clustering results. Two models (among those using the results of the mixture model) stand out in terms of logloss (predictive performance metric with which we compared the different models). Experiments conducted on simulated and real data have shown the importance of the preliminary unsupervised classification step on the quality of prediction. All of these steps were thus industrialized and integrated into the existing auction process. This integration is the succession of a set of steps : data retrieval, data pre-processing, estimation of the mixture parameters from carefully chosen explanatory variables, and finally, implementation of the predictive model. A last work enabled the exploitation of predictions obtained by the output of the click probabilities predictive models. Thus, we were able to predict the CTR in real time on the auction platform and for each available ad placement that transits through it. The analysis of the first production results shows that, for certain auction contexts, the use of the predictive model, coupled with the clustering step beforehand, has led to a significant improvement in the click rate.

Keywords : advertising, click-through-rate (CTR), GLM, clustering, mixture model, prediction

Sommaire

Liste des Acronymes

ARI	Adjusted Rand Index (Indice de Rand ajusté)
BIC	Bayesian Information Criterion
CTR	Click-Through-Rate (Taux de clics)
DSP	Demand Side Platform
EM	Expectation-Maximisation
FM	Factorization Machines
GLM	Modèle Linéaire Généralisé
GLMM	Modèle Linéaire Généralisé Mixte
IAB	Interactive Advertising Bureau
ICL	Integrated Completed Likelihood
KPI	Key Performance Indicator (Indicateur clé de performance)
LMM	Modèle Linéaire Mixte
OS	Operating System (Système d'exploitation)
RTB	Real-Time-Bidding (enchères en temps réel)
SSP	Supply Side Platform

Liste des Symboles

β	Vecteur des effets fixes
ϵ	Vecteur des erreurs
γ	Paramètres du modèle binomial
λ	Vecteur des proportions du mélange
l	Vraisemblance
L	Logvraisemblance
M	Matrice des variables explicatives (effets fixes)
ϕ	Paramètres du modèle de mélange
Π	Matrice de probabilité d'appartenance
ψ	Paramètre de dispersion
y	Vecteur des observations de la variable Y
θ	Paramètre canonique
U	Matrice des variables à effets aléatoires
ξ	Vecteur des effets aléatoires
Y	Vecteur aléatoire
Z	Vecteur aléatoire caché

Liste des Figures

Liste des Tables

Chapitre 1

Introduction

1.1 L'émergence de la publicité en ligne

1.1.1 Contexte historique

La publicité en ligne a démarré en 1994 lorsqu'un magazine en ligne, HotWired, a voulu vendre une bannière au fournisseur de services téléphoniques le plus populaire des États-Unis : AT&T [Kaye and Medoff (2001)]. Cette publicité a donc été diffusée sur cette page web. Depuis, le revenu généré par la publicité en ligne n'a pas cessé de croître au fil des années. À titre d'exemple, quelques chiffres présentés par [Evans (2009)] montrent qu'aux États-Unis les recettes de la publicité en ligne sont passées de 8 milliards de dollars en 2000 à 21 milliards en 2007.
Au début des années 2010, 15 ans après l'émergence de la publicité en ligne, l'écosystème commence à réellement se dessiner, impliquant un certain nombre d'acteurs. D'un côté, se trouvent les publicitaires qui veulent attirer l'attention et augmenter la visibilité de leur produit ; de l'autre, il y a les internautes qui sont réceptifs ou pas à ces publicités d'un nouveau genre. Entre les deux, des agences de publicités se sont positionnées pour jouer ce rôle d'intermédiaire.

La publicité en ligne, en comparaison avec la publicité dite "traditionnelle", présente un certain nombre d'avantages. Les agences publicitaires possèdent des informations beaucoup plus précises concernant les utilisateurs. Elles peuvent même essayer de les cibler en utilisant les adresses IP. Le niveau d'information à propos d'un utilisateur s'est considérablement affiné au cours du temps. Par exemple, un internaute qui va sur un moteur de recherche, va se voir proposer des publicités en lien avec les mots clés qu'il va saisir. Par ailleurs, l'industrie publicitaire a profité de l'émergence du web pour résoudre une problématique tout à fait centrale pour le business de ce domaine-là : être en mesure de délivrer une publicité à un grand nombre d'utilisateurs en même temps. Les agences publicitaires qui, jusqu'à la fin des années 90, géraient les transactions entre un média type journal et un publicitaire, se voient ainsi offrir un nouveau champ des possibles : la diffusion en masse ainsi qu'une personnalisation possible de leur diffusion publicitaire.

La publicité programmatique s'appuie sur le principe du Real Time Bidding (RTB) qui

permet de proposer et vendre en temps réel des emplacements publicitaires au plus offrant. Certains chiffres sont parlants quant au succès de cette nouvelle méthode d'acquisition d'emplacements publicitaires [Yuan et al. (2014)] : en 2011, parmi tous les publicitaires Nord-Américains, 88% d'entre eux utilisaient le RTB.

1.1.2 Les différents acteurs de la publicité en ligne

La publicité en ligne est un écosystème vaste et complexe dont les acteurs principaux sont décrits ci-après.

Demand side platforms (DSP) : ces entreprises travaillent au service des annonceurs ou agences de publicité en proposant un service d'enchère pour leurs campagnes publicitaires à travers différentes plateformes et de manière automatisée.

Supply side platforms (SSP) : il s'agit d'entreprises qui travaillent au service des éditeurs en répertoriant l'ensemble de leurs inventaires disponibles pour les mettre à disposition des DSP. Aussi, ils récupèrent de manière automatique les publicités proposées au cours d'une enchère par les différents DSP.

Annonceurs (Advertiser en anglais) : il s'agit d'entreprises ou de marques qui souhaitent mettre en avant leur(s) produit(s) à travers des campagnes publicitaires.

Editeurs (Publisher en anglais) : il s'agit des sites web ou applications mobiles qui mettent à disposition des encarts publicitaires en les proposant aux annonceurs via un système d'enchère.

1.1.3 Le processus de Real Timing Bidding

Comme expliqué par [Yuan et al. (2012)] et [Yuan et al. (2013)], le processus d'enchère en temps réel se déroule en plusieurs étapes :

1. L'annonceur crée ses campagnes publicitaires et les met à disposition du DSP avec lequel il travaille. Chaque DSP possède ainsi ce que l'on appelle un inventaire publicitaire avec l'ensemble des publicités à diffuser de ses différents clients.
2. En parallèle, l'éditeur notifie au SSP dès lors qu'il possède un emplacement publicitaire disponible (la présence d'un emplacement publicitaire disponible correspond à l'action d'un internaute qui ouvre une page web). Il lance un processus d'enchère afin de pouvoir, par la suite, sélectionner la publicité la plus offrante.
3. Le DSP reçoit l'appel aux enchères puis choisit parmi son inventaire la publicité qui correspondra le mieux à l'emplacement publicitaire ainsi qu'un prix d'enchère.
4. Ainsi, le SSP reçoit des propositions d'enchères de publicités de différents DSP et va automatiquement choisir celle dont le prix d'enchère est le plus élevé.

5. Le SSP va ainsi pouvoir renvoyer l'information à l'éditeur avec lequel il travaille et diffuser la publicité à l'internaute en train de naviguer sur la page web.

Tout ce processus se doit d'être fait en moins de 200 ms afin de garantir de la transparence vis-à-vis de l'utilisateur qui navigue sur le site web.

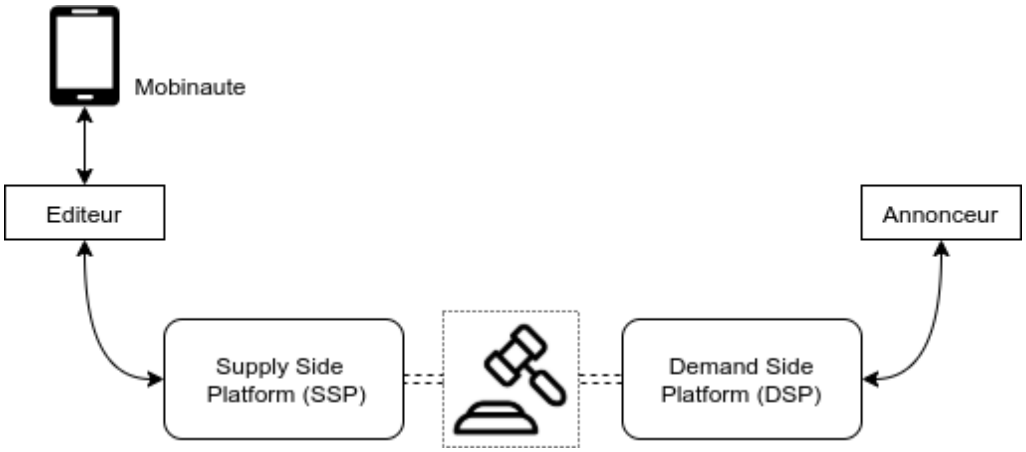


FIGURE 1.1 – Schéma simplifié du fonctionnement du (RTB) : affichage d'une publicité sur le téléphone d'un mobinaute via un système d'enchère en temps réel. Les annonceurs travaillent avec les DSP pour diffuser les campagnes publicitaires. Les éditeurs (applications) avec les SSP afin de mettre à l'enchère les encarts publicitaires.

1.1.4 Le succès d'une campagne publicitaire

Il existe plusieurs métriques standards pour estimer le succès d'une publicité.

— le CTR : la plus connue d'entre elles est le *taux de clics*, terme plus souvent connu et utilisé dans sa version anglophone *click-through rate (CTR)*. Il s'agit du rapport entre le nombre de fois où une publicité est cliquée et le nombre de fois où celle-ci a été diffusée et donc vue. Naturellement, le CTR prend ses valeurs dans l'intervalle $[0, 1]$. Plus cette valeur se rapproche de 1, plus on va considérer que la publicité est attractive.

— le CPC : le *coût par clic* correspond au coût total de la diffusion divisé par le nombre de clics obtenus. En pratique, il est plutôt question de CPC max, qui est un modèle de tarification où l'annonceur fixe un CPC maximal correspondant à l'enchère maximum pour laquelle il est prêt à payer.

— le CPA : il s'agit du *coût par action*. Cette métrique permet de connaître le coût payé par l'annonceur pour atteindre une conversion. Une conversion correspond à une action fixée par l'annonceur (il peut s'agir d'un clic mais aussi d'une vidéo complétée, d'un achat réalisé, d'une application téléchargée...).

— le CPM : le *coût pour mille* impressions est une métrique de facturation qui correspond au montant à payer lorsque la publicité est visualisée mille fois.

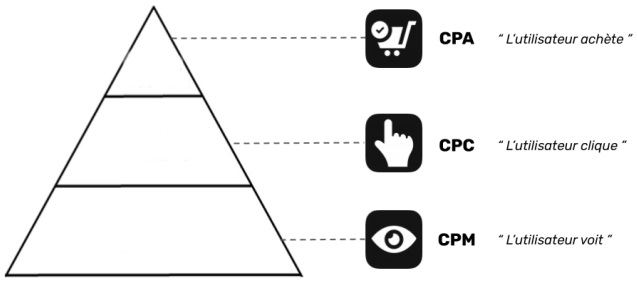


FIGURE 1.2 – Pyramide des coûts - CPM : coût pour mille impressions, CPC : coût par clic, CPA : coût par action

Ces différents indicateurs sont décrits sur la Figure 1.2. Il est important de garder en tête que le type de métrique à quantifier et analyser dépend directement de l'objectif de la campagne publicitaire fixé en amont.

1.1.5 Normaliser l'écosystème via l'IAB

En 1996, l'*Interactive Advertising Bureau* (IAB) a été fondé dans le but d'unifier et standardiser les protocoles entre les différents acteurs de la publicité en ligne. L'IAB, à travers le projet *OpenRTB* a permis d'élaborer des normes techniques ainsi que la mise en place des bonnes pratiques afin d'assurer une communication plus transparente et simplifiée entre les partenaires (acheteurs comme vendeurs).

1.2 L'évolution du RTB avec le programmatique mobile

1.2.1 L'utilisation d'un nouveau type d'information

Au cours des dernières années, la publicité en ligne a subi une nouvelle transformation avec l'utilisation massive des mobiles et tablettes. Ces nouveaux supports de distribution ont complètement changé la donne avec l'arrivée de nouvelles informations notamment l'identifiant publicitaire unique présent pour chaque support (tablette, mobile), les caractéristiques inhérentes à ceux-ci, ou encore la géolocalisation. Cette transformation du marché a abouti à la création de formats publicitaires adaptés à ces nouveaux médias et toujours plus innovants.
En effet, l'évolution technologique générée par les téléphones mobiles a dû prendre en considération un certain nombre de paramètres qui n'étaient jusqu'alors pas forcément étudiés sur la publicité en ligne dite classique, i.e sur un ordinateur, comme expliqué par [Grewal et al. (2016)]. Par exemple, les publicitaires se sont penchés sur la question de la

taille de la publicité par rapport à la taille de l'écran, ou encore, le fait qu'un mobinaute ne va plus utiliser d'intermédiaire (souris) mais interagir directement avec son doigt sur la publicité. Autant de phénomènes socialo-technologiques qui permettent d'affiner l'expérience utilisateur.
Le contexte global dans lequel se trouve l'utilisateur est aussi un des facteurs importants du succès de la publicité mobile. Le mobile a permis de rendre beaucoup plus facile la création d'un besoin pour l'utilisateur. Et l'un des atouts majeurs servant cet objectif-là réside dans l'utilisation de la géolocalisation. Une publicité pour une réduction dans un magasin se trouvant proche du mobinaute qui la reçoit va avoir beaucoup plus d'impact que s'il la reçoit depuis son ordinateur fixe, loin du-dit magasin. L'une des principales conséquences de ce changement insufflé par le mobile concerne le ciblage publicitaire. L'arrivée de nouveaux types de données et la disponibilité de celles-ci à travers une requête d'enchère a ouvert un champ des possibles incroyable pour les annonceurs, toujours à la recherche d'un meilleur ciblage de leurs publicités.

1.2.2 Le fonctionnement des enchères : *First Price* vs *Second Price* ?

L'enchère au Second Price (à droite sur la figure 1.3) est le système d'enchère historique. Il consiste à octroyer l'emplacement publicitaire à l'annonceur le plus offrant. Cependant le prix payé sera celui du 2ème plus offrant plus 1 centime. Ce système est avantageux pour les annonceurs puisqu'il permet d'enchérir de manière assez élevée sur des emplacements tout en les payant au prix du marché.
Depuis quelques années, le paradigme est en train de changer et peu à peu de nombreux SSP proposent des enchères dites au first price. Dans ce cas, l'annonceur gagnant paie l'emplacement publicitaire au prix exact pour lequel il a enchéri.

FIGURE 1.3 – Principe de l'achat *First Price* vs *Second price* : avec l'achat First Price, l'enchère est remportée au prix du plus offrant. Avec le Second Price, l'enchère est remportée par le plus offrant mais avec le prix du deuxième plus offrant + 1 centime.

1.3 Le Real Time Bidding chez TabMo

1.3.1 Présentation de l'entreprise

Fondée en septembre 2013, TabMo compte actuellement plus de 70 collaborateurs répartis entre Paris, Montpellier, Londres et Cologne. TabMo est un acteur novateur de la publicité programmatique mobile. Avec sa plateforme Hawk, TabMo a créé l'une des premières plateformes au monde d'achat d'emplacement publicitaire entièrement dédiée au mobile.
L'équipe R&D de TabMo, appelée TabMo Labs, est située à Montpellier. Cette équipe a pour objectif le développement d'un ensemble d'outils, qui vont de la plateforme de programmation de campagnes publicitaires à des algorithmes de diffusion. Près d'un million de requêtes sont traitées chaque seconde (soit plusieurs milliards de requêtes par jour) ce qui permet de diffuser les campagnes de publicité mobile des clients de TabMo à travers le monde.
L'un des objectifs majeurs du RTB (et donc de TabMo) est de pouvoir proposer la bonne annonce publicitaire à la bonne personne au bon endroit et au bon moment. Pour quantifier cette adéquation entre un mobinaute et une annonce publicitaire, il est possible d'étudier les interactions entre l'utilisateur et l'annonce publicitaire. Une interaction peut se traduire par un clic sur l'annonce, par le temps de visionnage de la vidéo publicitaire ou toute action faite par le mobinaute sur une publicité interactive. Il est aussi possible d'étudier les actions menées par l'utilisateur suite à la diffusion de la publicité comme l'achat d'un produit sur le site de l'annonceur, le téléchargement d'une application, les visites en magasin appartenant à l'annonceur ou encore la souscription d'un abonnement/d'une newsletter chez l'annonceur. L'ensemble de ces actions caractérise les performances d'une campagne et représente un retour sur investissement que les annonceurs ont besoin de maîtriser.

Les schémas présents dans la figure 1.1 représentent le fonctionnement du programmatique mobile défini chez TabMo de manière chronologique (de haut en bas et de gauche à droite) :

Sur la première image, un emplacement publicitaire est disponible sur l'application mobile que le mobinaute vient d'ouvrir.

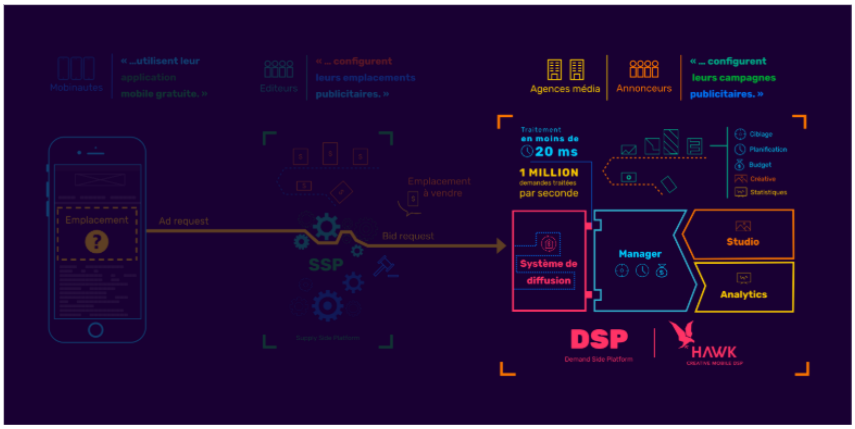
Cette information est directement liée et envoyée à un SSP via ce que l'on nomme une *Ad Request*, comme indiqué sur la deuxième image. Le SSP envoie la notification d'un emplacement à vendre via une *Bid Request* aux différents DSP avec lesquels il travaille. La *Bid Request* possède un certain nombre d'informations comme la taille de l'emplacement et d'autres caractéristiques liées au téléphone mobile.

L'image 3 fait un zoom sur le rôle du DSP. Ici, il s'agit du DSP de TabMo : Hawk.

Comme expliqué précédemment, le DSP met à disposition un outil et de l'expertise pour configurer les campagnes publicitaires des annonceurs avec qui il travaille.

Enfin, sur la dernière image, le DSP répond à la Bid Request via une *Bid Response* qui contient l'offre d'achat (prix d'enchère) avec le visuel de la publicité. Le SSP collecte les différentes offres reçues (une par DSP avec qui il travaille). Le plus offrant remporte l'enchère et ce sera sa publicité qui sera affichée sur l'écran du mobinaute, via une *Ad response*.

L'ensemble de ce processus est transparent pour l'utilisateur puisque toutes ces étapes doivent être faites en moins de 200ms. Du côté de la plateforme Hawk de TabMo, pour chaque *Bid Request* entrante, la *Bid Response* est renvoyée en moins de 20ms.



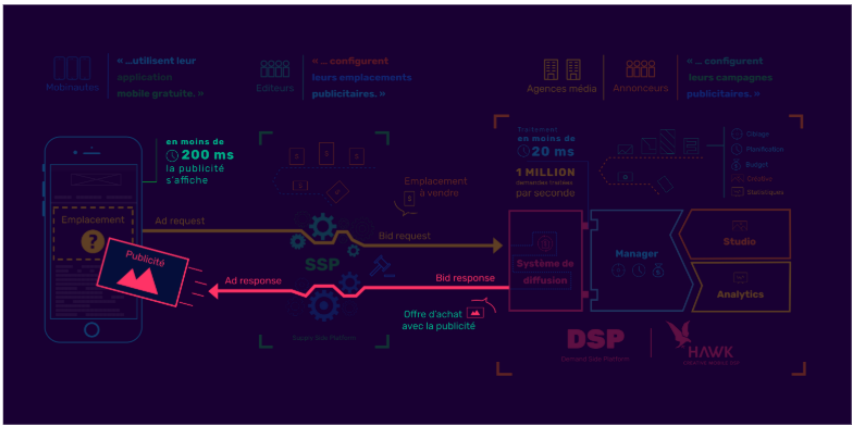


TABLE 1.1 – De haut en bas et de gauche à droite : la chronologie des quatre étapes principales d'une enchère en temps réel : 1-Mise à l'enchère d'un encart publicitaire. 2-Transmission de cette information d'un SSP à un DSP via une Bid Request. 3-Le DSP renvoie une publicité et un prix d'enchère parmi l'inventaire de publicités qu'il possède (annonceurs avec lesquels il travaille) via une Bid Response. 4-Le plus offrant remporte l'enchère et sa publicité s'affiche sur l'écran du mobinaute.

1.3.2 Les données

Dans le cadre de cette thèse, nous utilisons un jeu de données directement extrait de la plateforme d'enchère de TabMo. La plateforme fournit une très grande volumétrie de données avec environ un million de requêtes entrantes toutes les secondes. On considère ici un extrait de ces données réelles. La période d'étude s'étend de Septembre 2019 à Décembre 2019. Durant cette période, 5100 campagnes publicitaires sont observées ce qui représente environ 400 millions d'impressions. Pour ces travaux, un certain nombre de prétraitements ont dû être menés sur les données brutes et seront décrits dans la section 2.1.1.

1.3.2.1 Les données brutes

Nous représentons ici un aperçu des données brutes telles qu'on les reçoit sur la plateforme d'enchère de TabMo. Il s'agit de toutes les informations reçues pour chaque requête entrante, *Bid Request* (emplacement publicitaire disponible et mis à l'enchère) à laquelle on doit répondre.


```
{
        "auctionId": "123456abcd",
        "exchange": "rubicon",
        "appSiteId": "xapi:315268:8Y0GumzrTRl8",
        "appOrSite": "app",
        "publisherId": "22136",
        "deviceModel": "jkm-lx1",
        "deviceMake": "Huawei",
        "deviceType": "1",
        "carrier": "Other",
        "connectionType": "unknown",
        "language": "en",
        "os": "Android",
        "osv": "9",
        "country": "SAU",
        "city": "unknown",
```

```
        "zip": "23416",
        "lat": "25.000000",
        "lon": "45.000000",
        "ifa": "654321a-654321b-654321c",
        "mediaType": "banner",
        "creativeSize": "300x250",
        "time": "2019-11-12T02:22:22.034Z",
}
```


Tous les champs présents sont standardisés à partir de la norme *OpenRTB* comme expliqué dans la section 1.1.5 afin que tous les acteurs puissent communiquer ensemble avec les mêmes notations.
Certaines informations sont relatives au téléphone du mobinaute comme par exemple, le champ *deviceMake* qui indique la marque du mobile ou encore le champ *os* qui permet de connaitre le système d'exploitation. Les champs *mediaType*, *creativeSize* permettent de décrire l'encart publicitaire mis à l'enchère par l'éditeur. L'éditeur, quant à lui, est caractérisé par le champ *publisherId*. Nous ne ferons pas ici la description exhaustive de tous les champs disponibles puisque par la suite, nous analyserons en détail les champs nécessaires à la construction des variables explicatives de nos modèles.

1.3.2.2 Exemple illustratif de transformation des données brutes

On note C un ensemble des campagnes publicitaires qui diffusent sur la plateforme d'enchères Hawk de TabMo pendant une période donnée. Pour chaque campagne c, nous souhaitons comptabiliser le nombre de clics et le nombre d'impressions. On se concentre sur l'observation de deux campagnes notées respectivement c_1 et c_2 où on comptabilise le nombre de clics et d'impressions. Pour chaque clic et chaque impression, nous relevons le jour de la semaine, l'heure et le type de système d'exploitation du téléphone mobile sur lequel se situe l'encart publicitaire. On définit ces trois variables de la manière suivante :

1. l'heure de la journée qui varie de 1 à 24
2. le jour de la semaine qui varie de 1 à 7
3. le type de système d'exploitation qui est soit *Android*, soit *iOS*

On appelle contexte le nombre de clics (et d'impressions) comptabilisés pour un jour de la semaine donné, une heure de la journée donnée et un type de système d'exploitation donné. La table 1.2 permet de visualiser la structure d'observations pour construire ce jeu de données. La campagne c_1 a été vue 245 fois au cours du jour 6 de la semaine (samedi), sur le créneau horaire de 10h et sur les systèmes d'exploitation de type iOS. De manière analogue, nous comptabilisons 102 impressions pour le même jour, même créneau horaire mais sur les Android. Sur cet exemple illustratif, pour chaque campagne c nous avons plusieurs lignes associées avec un compteur d'impressions et de clics. Ces différentes

Campagne	*Date*	Jour	Heure	OS	**Clic**	**Impressions**
c_1	*2019-07-06*	6	10	iOS	**5**	**245**
c_1	*2019-07-06*	6	10	Android	**1**	**102**
c_1	*2019-07-06*	6	11	iOS	**46**	**890**
c_2	*2019-07-07*	6	10	Android	**2**	**512**

TABLE 1.2 – Exemple d'un jeu de données illustratif composé de deux campagnes publicitaires observées entre le 6 et 7 septembre 2019 : comptage du nombre de clics et d'impressions recensés selon le jour, l'heure et la famille de système d'exploitation.

observations d'une même campagne correspondent aux différents contextes observés dans les données brutes durant la période étudiée.

1.4 État de l'art métier : prédiction du CTR

La prédiction du CTR par des modèles statistiques fait partie des sujets les plus étudiés de ces dernières années dans les domaines de la recommandation et de la publicité en ligne. Nous présentons ici un aperçu des grandes catégories de méthodes qui ont été développées pour répondre à cette problématique.

1.4.1 Réseaux de neurones

Ces trois dernières années, le nombre de variables disponibles et la volumétrie des données ont considérablement augmenté. C'est l'une des raisons pour lesquelles un grand nombre de modèles profonds ont été développés récemment. Ces réseaux de neurones ont naturellement émergé dans l'état de l'art de la publicité en ligne pour la prédiction du CTR. [Cheng et al. (2016)] ont développé un modèle prédictif hybride utilisant à la fois les avantages du modèle linéaire et de l'architecture de réseaux profonds. Ils combinent les prédictions des deux composantes pour obtenir une prédiction finale et globale. Dans les travaux de [Wang et al. (2017)], le modèle proposé se base aussi sur un réseau de neurones profond (DNN pour Deep Neural Network). L'architecture du modèle se concentre sur la prise en compte des interactions entre les variables au-delà de l'ordre 2. Ainsi, sans prétraitement manuel des données, et avec une implémentation assez simple pour ce type de réseau, les auteurs obtiennent des résultats convaincants avec une réduction de leur critère d'évaluation qu'est la logloss. Cette dernière se base sur le comparatif entre une probabilité prédite et la valeur réelle de la variable à prédire. Elle est décrite en détail dans la section 3.2.1. Il existe encore d'autres modélisations de la prédiction du CTR qui se basent sur les réseaux de neurones [Liu et al. (2018); Chan et al. (2018); Zhou et al. (2018)]. Globalement, les réseaux de neurones profitent de leur architecture multicouche pour obtenir de bons et prometteurs résultats prédictifs. Cependant, très souvent, leur

complexité rend leur compréhension et leur explicabilité très difficile pour l'utilisateur.

1.4.2 Factorization Machines

Les Factorization Machines (FM), concept introduit par [Rendle (2010)], modélisent un polynôme de second ordre avec un vecteur latent pour chaque variable explicative. Les FM modélisent les interactions entre variables en les projetant sur un espace de faible dimension. Par exemple, on considère l'interaction du second ordre entre deux dimensions d_i et d_j représentée par $w_{ij} * x_i * x_j$ où w_{ij} est le paramètre à estimer. Les FM font l'hypothèse que le paramètre w_{ij} s'écrit comme le produit scalaire de deux vecteurs latents v_i et v_j. La dimension d des vecteurs v_i et v_j est un hyper paramètre, choisi avant l'entraînement du modèle. En considérant que la dimension des données d'entrainement est égale à N, alors au lieu d'estimer les $N \times N$ paramètres d'interactions du second ordre, il suffit d'estimer les paramètres $N \times d$ (éléments des vecteurs de dimension d). Cette hypothèse est forte et difficile à vérifier en pratique. Mais elle permet d'une part de réduire le nombre de paramètres à estimer et d'autre part, d'avoir une prédiction pour des interactions non observées.
Par la suite, [Rendle (2012)] a étendu son travail sur les FM au cas de la prédiction du CTR sur les réseaux sociaux. De nombreuses extensions ont émergé de ces Factorization Machines notamment avec les travaux de [Juan et al. (2016)] puis de [Guo et al. (2017)] qui répondent aux problématiques de prédiction du CTR dans la publicité en ligne où la prise en compte des interactions entre variables peut s'avérer très pertinente. [Juan et al. (2016, 2017)] ont proposé une modélisation basée sur les interactions entre les modalités de variables qualitatives tandis que [Guo et al. (2017)] ont développé un modèle qui combine à la fois l'architecture profonde des réseaux neuronaux avec celle des Factorization Machines. [Pan et al. (2016)] centrent la modélisation sur l'aspect clairsemé des données de publicités où une grande partie des CTR observés valent zéro. Ils introduisent ainsi une méthode *Sparse Factorization Machine* en remplaçant la distribution gaussienne par une distribution de Laplace pour modéliser le caractère clairsemé des données du CTR. Une autre extension des Factorization Machines a été développée par [Oentaryo et al. (2014)]. Cette dernière ajoute à la modélisation classique des FM des notions de hiérarchie entre les variables issues des données à disposition. La modélisation va par exemple prendre en considération les liens existants entre un annonceur A et ses différentes publicités. Cela va permettre à une publicité qui a peu diffusé d'apprendre des autres publicités issues du même annonceur. Ceci permet notamment de résoudre la problématique des nouvelles publicités et sur lesquelles aucun historique d'apprentissage n'est disponible.

1.4.3 Régression logistique

La prédiction du CTR par régression logistique est également l'un des modèles les plus étudiés dans la littérature comme en témoignent les travaux de [Chapelle et al. (2015); Kondakindi et al. (2014); Kumar et al. (2015)]. [Richardson et al. (2007)] développent un modèle prédictif pour le CTR à partir d'une régression logistique. Mais leur objectif principal réside dans la prédiction du clic pour les publicités qui n'ont encore jamais été vues. Il s'agit là d'un problème récurrent lorsque les modèles de prédiction du CTR sont utilisés en temps réel dans un moteur d'enchère. Dans l'approche proposée par [Yan et al. (2014)], la régression linéaire est étendue afin de prendre en considération les liens non linéaires entre les variables et ainsi étudier conjointement des informations liées à l'utilisateur et celles liées à la publicité. Pour cela, la méthode Group lasso est introduite afin de faire de la sélection de variables en bloc parmi toutes les nouvelles variables créées par le croisement des deux types d'informations. Dans [Ren et al. (2016)], la prédiction du CTR utilise classiquement une régression logistique. Mais l'objectif ici ne porte pas seulement sur la prédiction du CTR mais également sur l'optimisation de l'ensemble du processus d'enchère au second prix (voir figure 1.3) en modélisant une nouvelle fonction d'enchère prenant en compte le prix d'enchère.
Les modèles de régression logistique présentent l'avantage d'une mise en œuvre facile pour la mise en production de modèles de prédiction. Cependant, pour modéliser des structures de données complexes, clairsemées et hétérogènes, issues d'un contexte industriel par exemple, l'utilisation de la régression logistique peut être limitée.

1.4.4 Les modèles prédictifs du CTR déployés dans les entreprises

De nombreuses entreprises diffusant des publicités ont étudié la modélisation de la prédiction du CTR afin d'optimiser les performances de leur système publicitaire. [He et al. (2014)] présentent des travaux réalisés pour l'optimisation de la prédiction de clics chez Facebook. Le modèle développé est un modèle de classification qui combine arbre de décision (et plus précisément les boosted decision trees introduits par [Friedman (2001)]) et régression logistique. Le prétraitement des données et la construction des variables explicatives sont particulièrement détaillés ici afin d'appuyer l'idée que cette partie préalable à la modélisation statistique est tout aussi essentielle pour l'obtention de bons résultats prédictifs. Les modèles prédictifs développés par Twitter [Li et al. (2015)] se basent sur l'apprentissage obtenu sur l'historique des clics engendrés par les différentes publicités mais également sur le contenu du flux de tweet afin d'améliorer le taux de prédiction. [Graepel et al. (2010)] développent un modèle bayésien de régression binomiale pour la prédiction du clic sur les publicités de leur moteur de recherche *Microsoft Bing*.

1.5 État de l'art des méthodes statistiques utilisées

1.5.1 Modèles linéaires généralisés

Les modèles linéaires généralisés (GLM) ont été initialement introduits par [Nelder and Wedderburn (1972)] et sont une extension des modèles linéaires classiques. Ils permettent de traiter des problèmes de régression dont la distribution de la variable réponse ne suit pas nécessairement une loi normale. Depuis, ces modèles se sont largement popularisés et dans des domaines très variés. Un GLM est caractérisé par trois éléments : sa distribution, son prédicteur linéaire ainsi que sa fonction de lien reliant la variable à expliquer aux variables explicatives.

Distribution de la variable à expliquer On note Y le vecteur de taille N des observations de la variable à expliquer. On considère que les composantes Y_i de Y ($i = 1, ..., N$) sont indépendantes et qu'elles sont distribuées selon une loi appartenant à la famille exponentielle définie par [Nelder and Wedderburn (1972)]. La densité de Y_i peut donc s'écrire sous la forme :

$$f_{Y_i}(y_i, \theta_i) = \exp\left(\frac{y_i\theta_i - b(\theta_i)}{a_i(\psi)} + c(y_i, \psi)\right) \tag{1.1}$$

où θ_i est un paramètre canonique, ψ un paramètre de dispersion. Les fonctions b et c sont connues et spécifiques à chaque distribution. La fonction a_i s'écrit $a_i(\psi) = \frac{\psi}{\omega_i}$ avec ω_i correspondant au poids de chaque observation.
L'espérance et la variance de Y_i sont définies à partir des fonctions a_i et b. On définit $L(y, \theta) = \log f_Y(y, \theta)$ la fonction de logvraisemblance associée aux observations de Y.
En se basant sur les résultats classiques :

$$\begin{cases} \mathbb{E}(\frac{\partial l}{\partial \theta}) & = 0 \\ \mathbb{E}(\frac{\partial^2 l}{\partial \theta^2}) + \mathbb{E}((\frac{\partial l}{\partial \theta})^2) & = 0 \end{cases}$$

on obtient,

$$\begin{cases} \mathbb{E}(Y_i) & = b'(\theta_i) \\ \mathbb{V}(Y_i) & = a_i(\psi)b''(\theta_i) \end{cases}$$

Il existe une relation directe entre l'espérance (notée μ_i) et la variance Y_i :

$$\mathbb{V}(Y_i) = a_i(\psi)b''(b'^{-1}(\mu_i)) = a_i b''(b'^{-1}(\mu_i)) = a_i v(\mu_i) \tag{1.2}$$

avec $v = b'' \circ b'^{-1}$ que l'on notera fonction de variance dans la suite.

Définition du prédicteur linéaire On définit le prédicteur linéaire :

$$\eta = M\beta \tag{1.3}$$

où M est la matrice des variables explicatives de dimension $N \times p$ et β le vecteur des paramètres à estimer de taille p. Il existe un lien entre l'expérience de Y_i et la $i^{ème}$ composante du prédicteur linéaire η_i. Il s'agit de la fonction de lien g :

$$\eta_i = g(\mu_i) \tag{1.4}$$

Les modèles linéaires généralisés sont caractérisés par deux fonctions :

— la fonction de lien g qui permet l'introduction de la linéarité. Quelques exemples de fonctions de lien sont présentes dans la table 1.3.

— la fonction de variance v qui définit le lien entre la variance et l'espérance.

	$\frac{\mathcal{B}(n,p)}{n}$	$\mathcal{N}(\mu, \sigma^2)$	$\mathcal{P}(\lambda)$
Fonction de lien	$g(x) = \log(\frac{x}{1-x})$	$g(x) = x$	$g(x) = \log(x)$

TABLE 1.3 – Fonctions de lien des lois populaires de la famille exponentielle : binomiale, normale, poisson (de gauche à droite)

1.5.2 Modèles de mélange

Les modèles de mélange font partie des grands classiques de la classification non supervisée. L'hypothèse établie suppose que dans une population donnée, les individus peuvent être séparés en plusieurs sous-groupes. Nous sommes donc dans un cas où la population étudiée n'est pas décrite par une unique distribution mais plutôt par un mélange de K distributions. L'une des premières analyses statistiques qui utilise un modèle de mélange est celle de [Pearson (1894)] qui a créé un mélange de deux densités de probabilités gaussiennes pour un jeu de données concernant des mesures de proportions de la taille du crâne par rapport à la longueur du corps de 1000 crabes.
Les modèles de mélange se sont rapidement popularisés notamment grâce à des ouvrages très complets sur le sujet comme celui de [McLachlan and Peel (2004)].

1.5.2.1 Définition d'un modèle de mélange

On considère $Y_1, ..., Y_n$ un échantillon aléatoire de taille n où Y_i est une variable aléatoire de densité $f(y_i)$. Le modèle de mélange suppose que $f(y_i)$ s'écrit sous la forme :

$$f(y_i) = \sum_{k=1}^{K} \lambda_k f_k(y_i|\gamma_k) \tag{1.5}$$

avec $\forall k = 1, ..., K$ où K est le nombre de composants fixé. Les valeurs $\lambda_1, ..., \lambda_K$ correspondent aux proportions du mélange et sont comprises entre 0 et 1 avec $\sum_{k=1}^{K} \lambda_k = 1$.

Enfin, $f_k(.|\gamma_k)$ correspond à une densité de probabilité de paramètres notés γ_k.

Afin d'estimer les paramètres $(\gamma_k, \lambda_k)_{k=1,..,K}$ du modèle, on peut résoudre de manière itérative les équations du maximum de vraisemblance pour l'échantillon $Y_1, ..., Y_n$. La vraisemblance l s'écrit :

$$l(Y; \phi, \lambda) = \prod_{i=1}^{N} \sum_{k=1}^{K} \lambda_k f_k(y_i; \gamma_k) \tag{1.6}$$

Le logarithme de la vraisemblance, que l'on note L, est défini :

$$L(Y; \gamma, \lambda) = \sum_{i=1}^{N} \log \left\{ \sum_{k=1}^{K} \lambda_k f(y_i; \gamma_k) \right\} \tag{1.7}$$

Le modèle de mélange tel qu'il est décrit dans l'équation (1.7) est considéré comme un modèle avec une structure de données incomplètes. Nous ne connaissons pas pour chaque individu i à quelle composante k il appartient. La notion de variable cachée est alors introduite et se note Z_{ik} :

$$Z_{ik} = \begin{cases} 1 & \text{si l'individu } i \text{ appartient à la composante } k \\ 0 & \text{sinon} \end{cases}$$

La probabilité que la variable aléatoire Z_i soit égale à 1 s'écrit : $P(Z_{ik} = 1) = \lambda_k$. Il s'agit de la probabilité à priori d'appartenance à la composante k et représente la probabilité qu'une observation i appartienne à la composante k. La somme des événements possibles pour une observation vaut 1 d'où : $\sum_{k=1}^{K} \lambda_k = 1$.
Ainsi, la variable Z_i suit une distribution multinomiale avec comme paramètres les probabilités à priori d'appartenance aux composantes k : $Z_i \sim \mathcal{M}(1, \lambda_1, ..., \lambda_K)$. Le couple (Y_i, Z_i) correspond donc à ce que l'on appelle la structure des données complètes où seuls les Y_i sont observés. Ce que l'on cherche à calculer est la probabilité que l'individu i appartienne à la composante k sachant que l'individu a été observé selon la valeur y_i de Y. Cette probabilité se nomme probabilité à posteriori et se note :

$$\pi_{ik} = P(Z_{ik} = 1 | Y_i = y_i) \tag{1.8}$$

On peut ainsi utiliser la logvraisemblance des données complètes à l'aide de la variable Z cachée :

$$\begin{aligned} L(Y, Z; \gamma, \lambda) &= \log \left\{ \prod_{i=1}^{N} \prod_{k=1}^{K} \left(\lambda_k f(y_i; \gamma_k)^{z_{ik}}\right) \right\} \\ &= \sum_{i=1}^{N} \left\{ \sum_{k=1}^{K} z_{ik} \log \left(\lambda_k f(y_i; \gamma_k)\right) \right\} \\ &= \sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \log \left(\lambda_k\right) + \sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \log \left(f(y_i; \gamma_k)\right) \end{aligned} \tag{1.9}$$

Lorsqu'il n'est pas possible de calculer une expression analytique de la logvraisemblance, les algorithmes itératifs de type Expectation-Maximization (EM) introduits par [Dempster

et al. (1977)] sont les plus populaires et efficaces pour obtenir l'estimation des paramètres $(\gamma_k, \lambda_k)_{k=1,..,K}$ du mélange. Les principales caractéristiques (points forts, point faibles, comportement théorique et pratique) de l'algorithme EM sont documentées dans les livres de [McLachlan and Krishnan (2007); Titterington et al. (1985)]. Les modèles de mélange restent l'un des terrains d'application majeurs pour les algorithmes EM.

1.5.2.2 L'algorithme EM

Etape E : à chaque itération, l'étape E de l'algorithme EM permet de calculer l'espérance de la logvraisemblance associée aux données complètes conditionnellement aux données observées $(Y_i)_{i=1,...,N}$ et aux valeurs courantes $\phi_k^{(m)} = (\gamma_k^{(m)}, \lambda_k^{(m)})_{k=1,...,K}$. On note :

$$Q(\phi|\phi^{(m)}) = E(L(Y, Z; \gamma, \lambda)|Y = y, \phi^{(m)}) \tag{1.10}$$

Par construction, on sait que $E(Z_{ik}|Y_i, \phi_k^{(m)}) = P(Z_{ik} = 1|Y_i, \phi_k^{(m)})$.
L'équation (1.10) devient donc :

$$Q(\phi|\phi^{(m)}) = \sum_{i=1}^{N}\sum_{k=1}^{K} \pi_{ik}^{(m)} \log(\lambda_k^{(m)}) + \sum_{i=1}^{N}\sum_{k=1}^{K} \pi_{ik}^{(m)} \log f(y_i; \gamma_k^{(m)}) \tag{1.11}$$

avec

$$\pi_{ik}^{(m)} = P(Z_{ik} = 1|Y_i, \phi_k^{(m)}) \tag{1.12}$$

$$= \frac{P(Y_i|Z_{ik} = 1, \phi^{(m)})P(Z_{ik} = 1)}{\sum_{l=1}^{K} P(Y_i|Z_{il} = 1, \phi^{(m)})P(Z_{il} = 1)} \tag{1.13}$$

$$= \frac{f_{\phi_k^{(m)}}(y_i)\lambda_k^{(m)}}{\sum_{l=1}^{K} f_{\phi_l^{(m)}}(y_i)\lambda_l^{(m)}} \tag{1.14}$$

La probabilité π_{ik} représente la probabilité que l'individu i appartienne à la composante k du mélange à l'itération (m).

Etape M : cette étape consiste à mettre à jour les paramètres du mélange en maximisant la quantité $Q(\phi|\phi^{(m)})$. À l'itération $(m+1)$, les proportions du mélange s'écrivent :

$$\lambda_k^{(m+1)} = \frac{1}{n}\sum_{i=1}^{N} \pi_{ik}^{(m)} \tag{1.15}$$

La mise à jour des paramètres ϕ_k du modèle s'effectue à partir de l'équation (1.11) et en résolvant l'équation suivante pour tout $k = 1, ..., K$ et pour tout $j = 1, ..., s$:

$$\sum_{i=1}^{N} \pi_{ik}^{(m)} \frac{\partial \log f(y_i; \phi_k^{(m)})}{\partial \phi_{kj}^{(m)}} = 0 \tag{1.16}$$

avec $\phi_k = (\phi_{kj}; j = 1, ..., s)$.
Cet algorithme et la mise à jour des paramètres s'adaptent en fonction de la famille de distribution (f_ϕ) choisie pour le mélange.

1.5.2.3 Les métriques pour le clustering

Choix du nombre de composantes : le choix du nombre de composantes d'un modèle de mélange est une question qui a engendré un certain nombre de travaux de recherche. Parmi les critères existants, le Critère d'information bayésien (BIC) proposé par [Schwarz et al. (1978)] est le plus populaire. En plus de prendre en compte le nombre de paramètres du modèle et la logvraisemblance maximisée, le critère prend en compte le nombre d'observations et se définit :

$$BIC = -2 \times \hat{L} + m \times \log(N) \tag{1.17}$$

où $\hat{L}$ représente la valeur maximale de la logvraisemblance des données incomplètes, m est le nombre de paramètres à estimer dans le modèle et N le nombre total d'observations. Au début du siècle, [Biernacki et al. (2000)] proposent un nouveau critère ICL (Integrated Completed Likelihood) qui se base sur l'idée du BIC mais approxime la logvraisemblance des données complètes du modèle. Les deux critères diffèrent notamment à cause de la pénalisation induite par l'entropie moyenne estimée pour le critère ICL. L'avantage de ce dernier réside dans sa tendance à être moins enclin à discriminer les groupes qui se chevauchent, ce qui le rend plus robuste face à certains jeux de données.

$$ICL = BIC - 2 \times \sum_{k=1}^{K} \sum_{i=1}^{N} \pi_{ik} \log(\pi_{ik}) \tag{1.18}$$

Critère de robustesse pour le clustering L'indice de [Rand (1971)] permet de mesurer la similarité entre deux partitions obtenues à la suite d'un clustering. L'indice calcule le pourcentage de paire d'individus classés pareil. Un indice de Rand égal à 1 correspond à deux clustering identiques en terme de partition. L'indice de Rand ajusté introduit par [Hubert and Arabie (1985)] est une version corrigée de l'indice de Rand. On considère la table de contingence décrite dans la table 1.4 pour deux partitions A et B avec respectivement k et l composantes. L'indice de Rand ajusté (ARI) se définit :

$$ARI = \frac{\sum_{l,k} \binom{n_{lk}}{2} - \left[\sum_l \binom{n_{l.}}{2} \sum_k \binom{n_{.k}}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_l \binom{n_{l.}}{2} + \sum_k \binom{n_{.k}}{2}\right] - \left[\sum_l \binom{n_{l.}}{2} \sum_k \binom{n_{.k}}{2}\right] / \binom{n}{2}} \tag{1.19}$$

	Partition 1				
Partition 2	p_1	p_2	...	p_k	Somme
q_1	n_{11}	n_{12}	...	n_{1k}	$n_{1.}$
q_2	n_{21}	n_{22}	...	n_{2k}	$n_{2.}$
...	...	...	$\ddots$	...	$\vdots$
q_l	n_{l1}	n_{l2}	...	n_{lk}	$n_{l.}$
Somme	$n_{.1}$	$n_{.2}$	...	$n_{.k}$	$\mathbf{n} = n_{..}$

TABLE 1.4 – Table de contingence pour deux partitions P_1 et P_2 obtenues pour deux clustering distincts : comparatif des groupes obtenus et similitudes entre chaque clustering

1.5.3 Modèles linéaires mixtes

Origine et définitions : on appelle modèle linéaire mixte (LMM) un modèle qui comporte à la fois des variables à effets fixes (comme ceux définis pour les GLM dans les sections précédentes) et des variables à effets aléatoires. Les variables à effets fixes ont un nombre figé de modalités et définissent la moyenne du modèle. À l'inverse, les variables à effets aléatoires vont avoir un grand nombre (que l'on peut même considérer comme infini) de modalités possibles. Sur un jeu de données étudié, seulement une partie de ces modalités sont représentées puisque bien souvent nous n'avons qu'un échantillon de celles-ci. Ainsi, on ne cherche pas à étudier chaque modalité mais plutôt étudier l'effet global de chaque variable à effets aléatoires en termes de variabilité. Un effet aléatoire est ainsi caractérisé par un paramètre de variance qu'il faut estimer. L'intérêt des effets aléatoires réside dans la différenciation que l'on peut faire entre la variation induite par ces effets et celle que l'on attribue aux erreurs du modèle.
L'origine des modèles mixtes repose sur les travaux de [Fisher (1919)] qui a étudié l'analyse de la variance afin de déterminer l'origine des différences observées en moyenne entre différents sous-groupes de données. Son objectif était de déterminer les sources de cette variabilité.

Un exemple pour mieux comprendre : pour bien comprendre la différence entre effet fixe ou aléatoire, nous pouvons prendre un exemple fictif. L'équipe marketing d'un grand groupe de soda souhaite comprendre dans quelle mesure le système d'exploitation (OS) sur lequel sera diffusé ses publicités influe sur la performance de ses campagnes publicitaires (meilleure visibilité). Dans un premier temps, l'équipe marketing s'intéresse à trois types de système d'exploitation : iOS, Android et Autre (qui réunit tous les autres systèmes possibles). On considère alors dans cette modélisation que le type de système d'exploitation est associé à un effet fixe. Et on cherche à savoir lequel parmi les 3 est le plus performant.
Par la suite, l'équipe marketing est en charge d'une nouvelle étude et doit donc modéliser une nouvelle problématique. Elle cherche à savoir si, parmi toutes les plateformes d'en-

chère qui existent sur le marché de la publicité en ligne, il existe des différences notables concernant la performance des campagnes publicitaires (meilleure visibilité). Autrement dit, l'équipe se questionne sur l'utilisation d'une plateforme plutôt qu'une autre et se demande si cela pourrait induire une variabilité dans la performance de ses campagnes publicitaires. Il n'est pas raisonnable d'envisager l'étude de tous les concurrents, d'une part parce qu'obtenir une liste exhaustive semble compliqué et d'autre part car cela coûterait trop cher à l'équipe marketing. De ce fait, l'équipe va simplement analyser un échantillon de ces plateformes (celles avec qui l'entreprise de soda travaille déjà et qui sont présentes dans le jeu d'apprentissage) en considérant un effet plateforme. On peut alors considérer la variable plateforme comme une variable à effets aléatoires.
Pour plus de précisions sur le choix d'un effet fixe ou aléatoire, il faut se référer aux travaux de [Searle et al. (2009)].

Les effets aléatoires pour les données répétées : les effets aléatoires peuvent aussi être intéressants pour définir des structures de dépendances entre les observations. Par exemple dans le cas où on collecte plusieurs mesures pour un même individu, il est légitime de penser que deux observations issues d'un même individu vont être corrélées et de fait ne seront pas indépendantes. Dans ce genre de problématique, on peut donc considérer un effet aléatoire de type "individu" pour lier entre elles les observations d'un même individu.

Notations : un modèle linéaire mixte (LMM) peut se formaliser de la manière suivante :

$$Y = M\beta + U\xi + \epsilon \tag{1.20}$$

où

- Y est le vecteur de la variable à expliquer de taille N
- β est le vecteur des effets fixes à estimer de taille p avec M la matrice des variables explicatives associée aux effets fixes et de dimension $N \times p$.
- ξ le vecteur des effets aléatoires de taille q. Si on considère L effets aléatoires dans le modèle on a $\xi = (\xi_1^T, ..., \xi_L^T)$ avec ξ_l la composante modélisant un effet aléatoire de dimension q_l et $\sum_{l=1}^{L} q_l = q$. On note U la matrice des variables avec effets aléatoires de dimension $N \times q$. Tout comme le vecteur des effets aléatoires ξ, elle se décompose en L matrices U_l chacune de dimension $N \times q_l$: $[U_1|...|U_L]$.
- ϵ est le vecteur aléatoire des erreurs de taille N. On suppose que pour tout l, ξ_l et ϵ sont indépendants.

Cas des modèles linéaires généralisés mixtes : conditionnellement à ses effets aléatoires, un modèle linéaire généralisé mixte (GLMM) possède les mêmes propriétés qu'un modèle linéaire généralisé [McCulloch and Neuhaus (2005)]. Cependant, contrairement

aux GLM, le prédicteur linéaire du GLMM contient une partie aléatoire et se définit comme une combinaison d'effets fixes et aléatoires :

$$\eta_\xi = M\beta + U\xi \tag{1.21}$$

Nous avons vu précédemment qu'un GLM peut être défini par sa fonction de lien et sa fonction de variance. Dans le cadre des GLMM, ces deux fonctions restent tout aussi essentielles, mais sont exprimées conditionnellement aux effets aléatoires ξ :

— La fonction de lien g s'écrit : $\eta_\xi = g(\mu_\xi)$ avec l'espérance conditionnelle $\mu_\xi = E(Y|\xi)$

— La fonction de variance v est définie pour toute observation Y_i $\forall i \in 1, ..., N$: $var(Y_i|\xi) = a_i v(\mu_{\xi,i})$

L'hypothèse de distribution est portée par la loi de Y conditionnellement à ξ. D'une part on a que, conditionnellement à ξ, les composantes de Y sont indépendantes. D'autre part, $\forall i \in 1, ..., N$, $Y_i|\xi$ est distribuée selon une loi issue de la famille exponentielle.

1.6 Motivations et contributions de cette thèse

TabMo reçoit plusieurs centaines de milliers de requêtes par seconde (soit plusieurs milliards de requêtes par jour) sur lesquelles l'objectif est de répondre avec la publicité la plus pertinente possible pour celui qui la reçoit et avec un prix d'enchère associé. À l'aide de ces téraoctets de données, il est possible d'extraire de la connaissance et modéliser des comportements utilisateurs pour améliorer le fonctionnement de la plateforme d'enchère en temps réel d'emplacements publicitaires de TabMo. Cette thèse s'inscrit donc dans une problématique industrielle globale et relative à la publicité en ligne. Nous souhaitons mener à bien tout le processus de mise à disposition de probabilités de clic en temps réel via un algorithme prédictif. Ce travail se décompose en plusieurs étapes : à commencer par la récolte de données, en passant par le prétraitement de celles-ci puis la modélisation statistique et enfin l'industrialisation du modèle en temps réel sur la plateforme d'enchère.

Caractérisation des données : les données sur lesquelles ces modélisations sont effectuées présentent un certain nombre de caractéristiques. L'objectif est d'étudier l'évolution d'une métrique (dans notre cas : le click-through-rate, taux de clics) au cours du temps. Il s'agit de données longitudinales puisque pour un même individu (ici des campagnes publicitaires), nous disposons d'observations à différents instants. Par ailleurs, le taux de clics se déduit du nombre de clics comparativement au nombre de fois où la campagne publicitaire a été exposée. On peut naturellement imaginer que le nombre de clics est bien inférieur au nombre de "non-clics". La modélisation doit ainsi prendre en compte le caractère déséquilibré du taux de clics que l'on cherche à prédire. À travers l'analyse des données issues des campagnes de

publicité mobile et des informations spécifiques qui transitent via notre plateforme telles que les caractéristiques de l'emplacement publicitaire ou la famille de système d'exploitation, ce travail est mené en étroite collaboration avec les experts métier. Ces derniers permettent par exemple d'apporter de l'information empirique lors de la création et du prétraitement affiné de variables explicatives pour la modélisation du comportement utilisateur face à une publicité.

Les choix méthodologiques : notre objectif est de développer un modèle prédictif mais qui prend en considération deux enjeux principaux. Le premier concerne le format des données de TabMo, nous devons prendre en compte les caractéristiques des données (décrites juste au-dessus) dans notre choix de modèle. Par ailleurs, et c'est ici l'autre enjeu, le modèle se doit d'être simple à mettre en production et capable de répondre dans le temps imparti et imposé par le contexte métier de l'enchère en temps réel (cf section 1.3.1 : *"la Bid Response est renvoyée en moins de 20ms"*). La conjugaison de ces deux enjeux majeurs ainsi que l'étude (non exhaustive) de l'état de l'art sur la prédiction du CTR (présenté dans la section 1.4) nous a conduits à adopter la méthodologie décrite dans les deux paragraphes qui suivent. Les modèles GLM ont en effet retenu notre attention dans le cadre de cette problématique industrielle car ce sont des modèles efficaces et faciles à mettre en production. Un modèle GLM commun à toutes les campagnes ne permet pas une modélisation suffisamment fine. Un modèle par campagne n'est pas envisageable en terme de qualité d'estimation. Afin de s'adapter aux données complexes, clairsemées et hétérogènes, nous avons opté pour un mélange de GLMs. Ce modèle de mélange nous a tout d'abord donné une meilleure description des données via une classification non supervisée. Il nous a également permis un compromis intéressant (en terme de réduction de dimension) pour la prédiction du taux de clics.

La classification non supervisée comme étape préliminaire : une étape préliminaire, mais essentielle, de classification non supervisée permet de grouper les campagnes publicitaires ayant des profils similaires selon un critère choisi (ici le taux de clics). Cette première étape se base sur un mélange de modèles linéaires généralisés qui sera l'objet du Chapitre 2. Les modèles de régression logistique, comme évoqué au cours de l'état de l'art (section 1.4), présentent l'avantage d'être explicables pour l'utilisateur, efficaces dans le cadre d'une classification binaire et dans un contexte où l'ensemble du processus décrit doit être exécuté en quelques secondes. Le modèle de mélange développé ici a été implémenté dans un package R nommé *binomialMix*. Il se base sur un algorithme EM et permet l'estimation de paramètres d'un modèle de mélange pour données longitudinales et non gaussiennes.

La prédiction du taux de clics, cœur de la problématique : en s'appuyant sur l'étape de clustering, l'objectif est de prédire, pour chaque emplacement publicitaire qui arrive sur la plateforme, la probabilité de clic des campagnes publicitaires pré-

sentes dans notre inventaire. Cette optimisation au moment du choix de la campagne à diffuser permet d'enchérir avec celle qui a la plus grande chance d'être cliquée et d'améliorer le CTR des annonceurs. Les résultats obtenus à l'issue de l'étape préliminaire de classification non supervisée se montreront cruciaux pour l'amélioration de la performance prédictive comme nous le verrons au cours du chapitre 3.

La mise en production de l'ensemble du processus : l'objectif final de cette thèse doit pouvoir servir l'objectif métier de l'entreprise TabMo. L'idée est donc de pouvoir utiliser le modèle de prédiction de clics en temps réel sur la plateforme d'enchère afin d'améliorer les KPI (notamment le CTR ici) des annonceurs, qui sont les clients et utilisateurs directs de la plateforme. L'ensemble des étapes décrites ci-dessus ont donc dû être industrialisées et intégrées dans le processus de développement déjà en place comme détaillé dans le chapitre 4.

Chapitre 2

Classification non supervisée de campagnes de publicité mobile : Modèle de mélange pour données longitudinales et non gaussiennes

L'objectif de ce chapitre concerne la classification de campagnes de publicité mobile à partir d'un modèle de mélange de distribution binomiale et pour données répétées. Les données correspondent à la diffusion de campagnes publicitaires dont on suit l'évolution de certaines métriques au cours du temps. Dans ce chapitre, nous nous concentrons sur l'analyse des métriques suivantes : le nombre d'impressions, le nombre de clics et le taux de clics. Cette étape de clustering est une étape préliminaire à un objectif plus global qui concerne la prédiction du taux de clics que nous verrons dans le chapitre 3. Elle est néanmoins importante et nécessaire afin de pouvoir regrouper des campagnes aux profils similaires en matière de CTR.

2.1 Modèle binomial pour le taux de clics

2.1.1 Construction du jeu de données

On se base sur le jeu de données décrit dans la section 1.3.2 et illustré dans la table 1.2. Ce jeu de données s'étend de septembre à décembre 2019. Plusieurs étapes de prétraitement ont été nécessaires avant de pouvoir utiliser les données dans le cadre du modèle de mélange. Ces étapes, résumées sur la figure 2.1, se décomposent de la manière suivante :

Extraction des données de la base de données TabMo : les requêtes entrantes et les réponses aux enchères sont stockées en temps réel dans une base de données. Ces données sont par la suite enrichies avec ce que l'on appelle *les évènements* qui sont survenus après l'enchère. Il peut s'agir d'un clic ou d'une publicité affichée (impression). De fait, pour chaque publicité affichée (conséquence d'une enchère gagnée), un label "impression" lui est assigné. Si la publicité a par la suite été cliquée, le label "clic" est également ajouté. À partir de ces données brutes stockées en base, il est possible d'en extraire une partie pour la modélisation.

Prétraitement des données brutes : ces traitements se divisent en différentes étapes.

— La première consiste à supprimer toutes les informations qui ne sont pas nécessaires pour la modélisation à venir mais aussi de repérer les données qui semblent incohérentes. Il arrive qu'il y ait par exemple un clic observé alors que l'évènement lié à l'impression n'a pas été remonté en base. Il est également possible d'observer plusieurs clics pour une même impression. Ce type d'incohérence peut avoir plusieurs origines. Étant donné la volumétrie des données traitées, un problème peut apparaître lors de l'enregistrement des événements. La récupération des notifications d'évènement peut également avoir une certaine latence et aboutir à des données temporairement erronées sur certaines campagnes. Enfin, la fraude liée à la génération massive de clics pour une impression est une problématique connue du programmatique mobile. Mais cela ne fait pas partie du scope de ces travaux. En conséquence, et en connaissant ce type d'erreur dans les données, nous avons pu développer des filtres afin de supprimer automatiquement ces incohérences.

— L'étape de prétraitement permet également d'extraire de l'information de données déjà existantes. Prenons l'exemple de la date présente dans chaque requête entrante (et sortante). Le format de la date est le suivant $AAAA-MM-JJHH:MM:SS$. Il est facile de créer de nouvelles variables calendaires à partir de celle-ci telles que le mois de l'année, le jour de la semaine, l'heure de la journée.

— Enfin, la dernière étape de ce processus de prétraitement s'appuie un travail collaboratif mené avec les experts métier. En se basant sur leur expertise, de nouvelles variables ont été créées. Prenons l'exemple des dimensions des publicités (d'un point de vue visuel). Il existe un grand nombre de formats différents. Mais en échangeant avec eux et en s'appuyant sur leur expérience, certains formats ont pu être regroupés. S'agissant des heures de la journée, l'expertise métier a permis de regrouper les heures en six plages horaires distinctes en faisant l'hypothèse empirique que ces plages horaires sont homogènes quant au comportement du mobinaute vis-à-vis des publicités auxquelles il est confronté.

Agrégation des données par contexte : pour cette dernière étape, il est nécessaire de définir la notion de **contexte**. Considérons les variables citées ci-dessus : plage horaire (H modalités) et format de la publicité (F modalités). On appelle contexte toutes les combinaisons possibles entre les différentes modalités de ces variables. Dans cet exemple, pour une campagne donnée, on peut observer chaque jour $H \times F$ contextes. Plus il y a de variables dans le jeu de données, plus il y aura de contextes potentiellement observables. L'objectif est ainsi d'agréger les impressions et clics par contexte et par campagne publicitaire du jeu de données. C'est cette dernière étape qui permet d'aboutir à la structure de données souhaitée et étudiée par la suite.

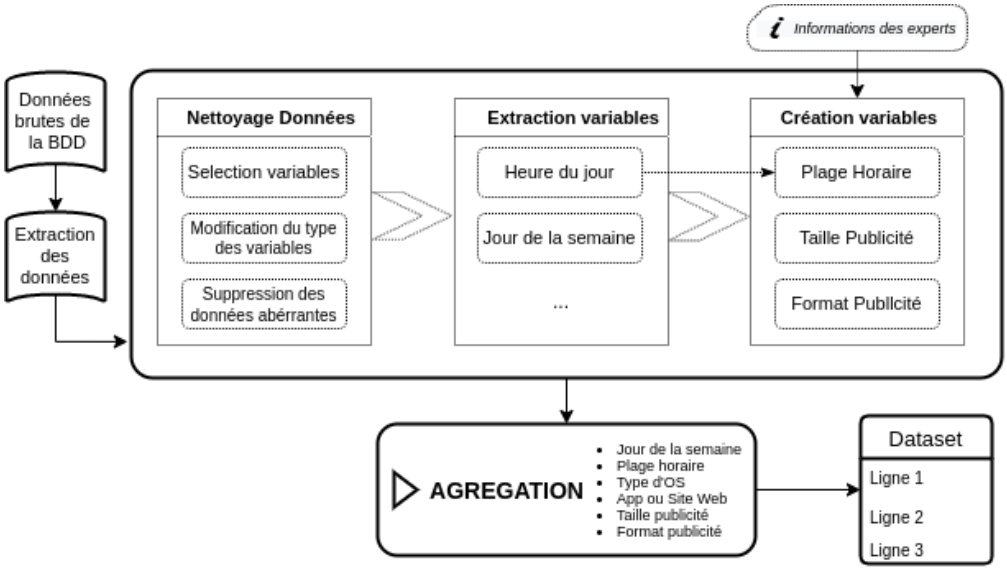


FIGURE 2.1 – Prétraitement des données : nettoyage des données brutes, extraction des variables nécessaires pour la modélisation, création de nouvelles variables à partir de variables existantes, agrégation selon le schéma de variables explicatives désirées pour la modélisation.

2.1.2 Les variables

Nous présentons ici les variables qui ont été sélectionnées et utilisées pour le mélange. Au total, 6 variables seront utilisées pour la modélisation, il s'agit de variables catégorielles uniquement : jour de la semaine, plage horaire, type de système d'exploitation (OS), type de support, type de publicité et format.

2.1.2.1 Des variables calendaires

Jour de la semaine variable catégorielle à 7 modalités représentant les jours de la semaine.

Plage horaire variable catégorielle à 6 modalités définies avec l'aide d'experts du domaine : (00h-4h,4h-8h,8h-12h,12h-16h,16h-20h,20h-00h). La proportion du nombre d'impressions en fonction des plages horaires est disponible sur la figure 2.2.

2.1.2.2 Des variables liées au contexte de l'enchère

Type de système d'exploitation (OS) variable catégorielle à 3 modalités correspondant aux différents systèmes d'exploitation possibles du mobile : iOS, Android ou autre.

Type de support variable catégorielle à 2 modalités correspondant aux différents supports de diffusion : application ou site web.

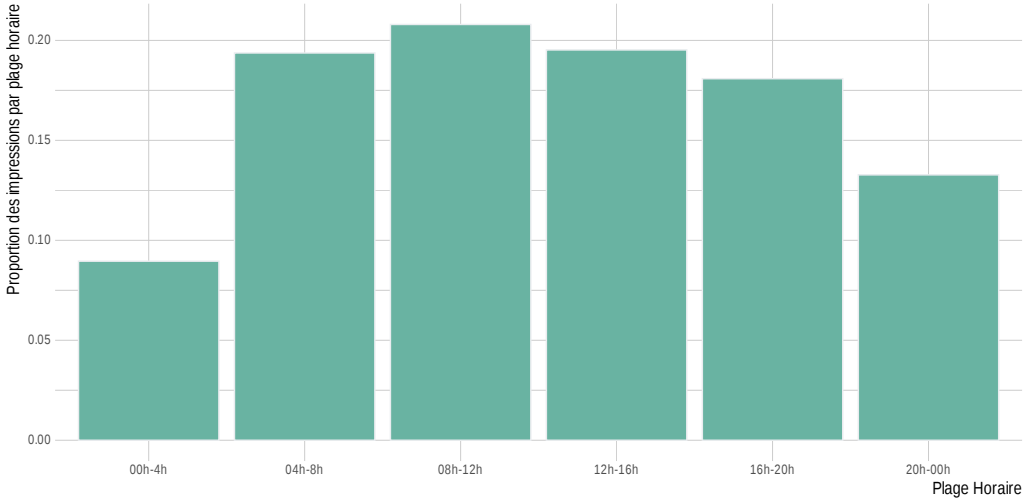


FIGURE 2.2 – Représentation pour chaque plage horaire de la proportion d'impressions observées dans l'intervalle.

2.1.2.3 Des variables liées à la campagne publicitaire

Type de publicité variable catégorielle avec 4 modalités possibles :

- Type 1 : format de publicité riche exploitant des vidéos, des animations et des scripts personnalisés pour créer de l'interaction avec l'utilisateur.
- Type 2 : format de publicité ayant accès à une interface de programmation (API) permettant d'interagir avec des fonctionnalités du téléphone, telles que l'appareil photo, le gyroscope, l'orientation de l'écran. Ce format, qui n'est disponible que dans le cadre d'une application, permet de concevoir des expériences publicitaires plus riches.
- Type 3 : format de publicité développé par les équipes Tabmo permettant de diffuser en toute simplicité des publicités innovantes et complexes (publicité dites "drive to store", avec des effets 3D, etc).
- Type 4 : format de publicité sous forme d'une simple image (animée ou fixe).

La figure 2.3 présente un exemple de publicité pour le type 4 (banner) à gauche et le type 3 à droite. On appelle *GPStore* le format de droite, puisqu'il s'agit d'une publicité où le mobinaute peut cliquer sur l'onglet entouré en pointillé "Y ALLER" pour être redirigé vers un plan où il peut géolocaliser le magasin le plus proche.

Format de la publicité variable catégorielle à 3 modalités : bannière, pavé ou interstitielle.

FIGURE 2.3 – Type de format publicitaires. De gauche à droite : type 4 (banner) et type 3 (GPStore). Le format GPStore est un format interactif qui permet de rediriger le mobinaute vers un itinéraire pouvant le mener jusqu'au magasin proposant la publicité.

2.1.2.4 La variable réponse

Pour la suite, la **variable réponse** est le taux de clics, nommé aussi Click-Through-Rate (CTR) dans le domaine de la publicité. Cette variable correspond au rapport entre le nombre de clics observés et le nombre d'impressions observées pour un contexte donné.

2.1.3 Modélisation du taux de clics

Les données étudiées décrivent l'évolution du nombre de clics et d'impressions pendant la durée de vie d'une campagne publicitaire. Cette structure de données est appelée *données répétées* ou *données longitudinales*. Chaque jour est divisé en H plages horaires. Chaque campagne publicitaire c est observée pendant J_c jours. Il est possible que certaines plages horaires ne soient pas observées au cours des J_c jours de diffusion de la campagne publicitaire. On note Y_{cjh} le nombre de clics observés pour une campagne publicitaire c sur une plage horaire définie (j, h), pour $j = 1, ..., J_c$ et $h = 1, ..., H$. On considère ainsi que Y_{cjh} suit une distribution binomiale de paramètre n_{cjh} et $p_{hs(c,j)}$:

$$Y_{cjh} \sim \mathcal{B}(n_{cjh}, p_{hs(c,j)}) \tag{2.1}$$

où n_{cjh} correspond au nombre d'impressions observées pour la campagne c durant la plage horaire (j, h), $p_{hs(c,j)}$ représente la probabilité de clic de la campagne c sur la plage horaire h lors du jour de la semaine $s(c, j)$ avec $s(c, j) = 1, ..., S$. Ici, nous considérons le taux de clics décrit par $\frac{Y_{cjh}}{n_{cjh}}$.
Classiquement, la densité d'une loi binomiale de paramètre $(n_{cjh}, p_{hs(c,j)})$ s'écrit : $f(y_{cjh}) =$

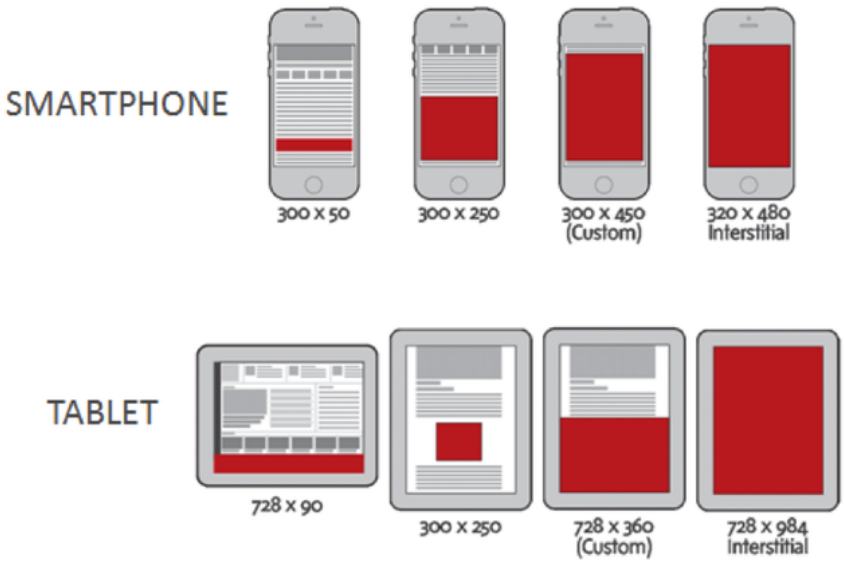


FIGURE 2.4 – Extrait de différentes tailles possibles pour afficher une publicité sur mobile et tablette (Source : Lien)

$\binom{n_{cjh}}{y_{cjh}} p_{hs(c,j)}^{y_{cjh}} (1 - p_{hs(c,j)})^{(n_{cjh} - y_{cjh})}$. On peut retrouver l'expression d'une distribution exponentielle présentée dans l'équation (1.1) ainsi que la valeur des différentes fonctions associées à ce type de distribution à partir de l'expression de la densité :

$$\log f(y_{cjh}) = \log \binom{n_{cjh}}{y_{cjh}} + \log \left(p_{hs(c,j)}^{y_{cjh}}\right) + \log \left(1 - p_{hs(c,j)}\right)^{n_{cjh} - y_{cjh}} \tag{2.2}$$

$$\Leftrightarrow f(y_{cjh}) = \exp \left(\frac{\frac{y_{cjh}\theta_{cjh}}{n_{cjh}} - \log \left(1 + \exp \theta_{cjh}\right)}{1/n_{cjh}} + \log \binom{n_{cjh}}{y_{cjh}} \right) \tag{2.3}$$

On en déduit ainsi les valeurs du paramètre canonique $\theta_{cjh} = \log \left(\frac{p_{hs(c,j)}}{1 - p_{hs(c,j)}} \right)$, de la fonction de poids $a_{cjh}(\psi) = \frac{1}{n_{cjh}}$ ainsi que des fonctions spécifiques à la distribution binomiale que sont $b(\theta_{cjh}) = \log \left(1 + \exp \theta_{cjh}\right)$ et $c(y_{cjh}, \psi_{cjh}) = \log \binom{n_{cjh}}{y_{cjh}}$. On note g la fonction de lien avec $g(\mu) = \log \frac{\mu}{1-\mu}$ où μ est l'espérance de taux de clics étudié : $\mu = E(\frac{Y_{cjh}}{n_{cjh}})$.
On obtient ainsi la relation suivante entre le CTR et les variables explicatives du modèle :

$$\log \left(\frac{E(Y_{cjh}/n_{cjh})}{1 - E(Y_{cjh}/n_{cjh})} \right) = \beta_0 + \beta_h^H + \beta_{s(c,j)}^S \tag{2.4}$$

Ici, nous illustrons cette relation en utilisant simplement β_h^H and $\beta_{s(c,j)}^S$, les coefficients associés aux variables calendaires présentées en section 2.1.2.1. Le vecteur β des coefficients du modèle binomial contient toutes les variables explicatives choisies pour la modélisation.

2.2 Mélange de distributions binomiales

L'objectif de ce travail est d'obtenir un mélange de distributions binomiales. Une introduction aux modèles de mélange et à l'algorithme EM est disponible en section 1.5.2. L'algorithme EM de l'une des méthodes les plus célèbres pour estimer les paramètres d'un modèle de mélange. Il s'agit d'un algorithme itératif introduit par [Dempster et al. (1977)]. Nous allons voir ici qu'un modèle de mélange de distributions issues de la famille exponentielle présente des spécificités dans la mise à jour des paramètres du modèle, que l'on nomme β (en référence à l'équation (2.4)). Le vecteur β ne possède dans ce cas-là pas de solution explicite. Nous allons ainsi dérouler les étapes E pour *Expectation* et M pour *Maximization* de l'algorithme et détailler la méthodologie afin de pouvoir estimer les paramètres du mélange.

2.2.1 Etape E pour le mélange de binomiale

Cette première étape a pour objectif de calculer et mettre à jour la probabilité d'appartenance de chaque campagne publicitaire C du jeu de données à un cluster k. En reprenant les notations introduites dans la section introductive 1.5.2, on considère l'espérance de la logvraisemblance L associée aux données complètes conditionnellement aux données observées $(Y_c)_{c=1,\dots,C}$ et aux valeurs courantes. On a donc :

$$Q(\phi^{(m)}|\phi^{(m-1)}) = E(L(Y, Z; \gamma, \lambda)|Y = y, \gamma^{(m-1)}) \tag{2.5}$$

où γ correspond aux paramètres du modèle binomial, λ aux proportions du mélange et ϕ correspond à l'ensemble de ces paramètres $\phi = (\gamma, \lambda)$ à mettre à jour. On note $\phi^{(m)}$ l'estimation des paramètres mis à jour à l'itération (m). On peut ainsi réécrire Q sous la forme suivante :

$$Q(\phi^{(m+1)}|\phi^{(m)}) = \sum_{c=1}^{C}\sum_{k=1}^{K} \pi_{kc}^{(m)} \log {\lambda_k}^{(m)} + \sum_{c=1}^{C}\sum_{k=1}^{K} \pi_{kc}^{(m)} \log f(y_c; \gamma_k^{(m)}) \tag{2.6}$$

$$= \sum_{c=1}^{C}\sum_{k=1}^{K} \pi_{kc}^{(m)} \log \lambda_k^{(m)} + \sum_{c=1}^{C}\sum_{k=1}^{K} \pi_{kc}^{(m)} \log \Big(\prod_{j=1}^{J_c}\prod_{h=1}^{H} f(y_{cjh}; \gamma_k^{(m)})\Big) \tag{2.7}$$

$$= \underbrace{\sum_{c=1}^{C}\sum_{k=1}^{K} \pi_{kc}^{(m)} \log \lambda_k^{(m)}}_{A} + \underbrace{\sum_{c=1}^{C}\sum_{j=1}^{J_c}\sum_{h=1}^{H}\sum_{k=1}^{K} \pi_{kc}^{(m)} \log f(y_{cjh}; \gamma_k^{(m)})}_{B} \tag{2.8}$$

avec la contrainte sur les proportions du mélange $\sum_{k=1}^{K} \lambda_k = 1$. Ainsi, au cours de cette première étape E, nous mettons à jour la matrice Π décrite dans l'équation (2.9) et qui correspond à la matrice d'appartenance des campagnes C aux différentes composantes K du modèle. On note $\pi_{kc}^{(m)} = P(Z_{kc} = 1|Y_c, \gamma_k^{(m)})$ où la variable cachée Z_{kc} correspond à la

probabilité d'appartenance à la composante k pour une campagne c.

$$\Pi = \begin{array}{c} \\ \mathrm{k}_1 \\ \mathrm{k}_2 \\ \vdots \\ \mathrm{k}_K \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & \dots & c_C \end{array} \\ \begin{pmatrix} \pi_{11} & \pi_{12} & \pi_{13} & \dots & \pi_{1C} \\ \pi_{21} & \pi_{22} & \pi_{23} & \dots & \pi_{2C} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \pi_{K1} & \pi_{K2} & \pi_{K3} & \dots & \pi_{KC} \end{pmatrix} \end{array} \tag{2.9}$$

La mise à jour de chaque élément $\pi_{kc}^{(m)} = P(Z_{kc} = 1|Y_c, \gamma_k^{(m)})$ de la matrice Π découle du théorème de Bayes. D'où le calcul qui suit :

$$\begin{aligned} P(Z_{kc} = 1|Y_c, \gamma^{(m)}) &= \frac{P(Y_c|Z_{kc} = 1, \gamma^{(m)})P(Z_{kc} = 1)}{\sum_{l=1}^{K} P(Y_c|Z_{lc} = 1, \gamma^{(m)})P(Z_{lc} = 1)} \\ &= \frac{f_{\gamma_k^{(m)}}(y_c)\lambda_k^{(m)}}{\sum_{l=1}^{K} f_{\gamma_l^{(m)}}(y_c)\lambda_l^{(m)}} \\ &= \pi_{kc}^{(m)} \end{aligned}$$

2.2.2 Etape M pour le mélange de binomiale

L'étape M, quant à elle, consiste à mettre à jour les paramètres du mélange $\phi = (\lambda, \gamma)$. Il s'agit de résoudre :

$$\phi^{(m+1)} = \underset{\phi^{(m)}}{Argmax} E(L(Y, Z; \gamma, \lambda)|Y = y, \gamma^{(m)}) \tag{2.10}$$

2.2.2.1 Estimation des proportions λ du mélange

La mise à jour du vecteur des proportions de chaque composante du modèle se fait à partir de la maximisation sous contrainte ($\sum_{k=1}^{K} \lambda_k = 1$). On maximise Q défini dans l'équation (2.8). De fait, on a

$$\begin{aligned} &\frac{\partial(Q + \alpha \sum_{k=1}^{K} \lambda_k)}{\partial \lambda_k} = 0 \\ \Leftrightarrow\ &\frac{\sum_{c=1}^{C} \pi_{kc}}{\lambda_k} + \alpha = 0 \\ \Leftrightarrow\ &\lambda_k = -\frac{\sum_{c=1}^{C} \pi_{kc}}{\alpha} \end{aligned}$$

avec $\alpha = -C$ (voir Appendice l'équation (B.4) de l'Appendice (B.1)) où C est le nombre total de campagnes à classifier. Au final, l'estimation du vecteur λ des proportions pour chaque composante s'écrit : $\lambda_k^{(m)} = \frac{\sum_{c=1}^{C} \pi_{kc}^{(m)}}{C}$

2.2.2.2 Estimation du vecteur β des coefficients associés aux variables du modèle binomial

L'estimation du vecteur des paramètres du modèle se base sur l'équation (2.8). Seule la partie B de l'équation dépend du vecteur β. Ainsi, le calcul se décompose de la manière suivante :

$$\frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k} = 0 \Leftrightarrow \frac{\partial \log f(y_c;\gamma_k)}{\partial \beta_{kp}} = 0 \Leftrightarrow \frac{\partial \eta_{ck}}{\partial \beta_{kp}} \frac{\partial \mu_{ck}}{\partial \eta_{ck}} \frac{\partial \theta_{ck}}{\partial \mu_{ck}} \frac{\partial \log f(y_c;\beta_k,\psi_k)}{\partial \theta_{ck}} = 0 \quad (2.11)$$

avec β le vecteur des paramètres à estimer de taille p, η le prédicteur linéaire de dimension C, θ le paramètre canonique et ψ le paramètre de dispersion. L'équation (2.11) se décompose de la manière suivante :

- $\frac{\partial \eta_{ck}}{\partial \beta_{kp}} = M_c$ où M_c est la matrice de design des variables explicatives.
- $\frac{\partial \mu_{ck}}{\partial \eta_{ck}} = \frac{1}{g'(\mu_{ck})}$ puisque $\frac{\partial \eta_{ck}}{\partial \mu_{ck}} = \frac{1}{\partial \mu_{ck}} g(\mu_{ck}) = g'(\mu_{ck})$
- $\frac{\partial \theta_{ck}}{\partial \mu_{ck}} = \frac{1}{b''(\theta_{ck})}$ puisque $\frac{\partial \mu_{ck}}{\partial \theta_{ck}} = \frac{1}{\partial \theta_{ck}} b'(\theta_{ck}) = b''(\theta_{ck})$
- $\frac{\partial \log f(y_c;\beta_k,\psi_k)}{\partial \theta_{ck}} = \frac{y_c - b'(\theta_{ck})}{a_c(\psi)}$

On obtient ainsi à partir de l'équation (2.11) et pour $k = 1, ..., K$:

$$\frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k} = 0 \Leftrightarrow \sum_{c=1}^{C} \pi_{ck}^{(m)} \left(M_c \frac{1}{g'(\mu_{ck})} \frac{1}{b''(\theta_{ck})} \frac{y_c - b'(\theta_{ck})}{a_c(\psi)} \right) = 0 \quad (2.12)$$

$$\Leftrightarrow \sum_{i=1}^{N} \pi_{ck}^{(m)} M_c W_{c\beta_k}^{-1} \frac{\partial \eta_{ck}}{\partial \mu_k} (Y_c - \mu_k) = 0 \quad (2.13)$$

$$\Leftrightarrow M^t W_{\beta_k}^{-1} \frac{\partial \eta_k}{\partial \mu_k} (y - \mu_k) = 0 \quad (2.14)$$

où on note les matrices W et $\frac{\partial \eta_k}{\partial \mu_k}$ de la manière suivante $\forall c$, $W_{c\beta_k} = diag\{(a_c(\psi_k) b''(\theta_{ck}) g'(\mu_{ck})^2)\}$ et $\frac{\partial \eta_k}{\partial \mu_k} = diag\left(\frac{\partial \eta_{ck}}{\partial \mu_{ck}}\right)$.

Dans le cadre d'un mélange de binomiale, ces matrices s'écrivent (voir les équations (B.8) et (B.9) pour le détail des calculs) : $\forall c, j, h$,

$$W_{c\beta_k} = diag\left(\frac{1}{n_c} \frac{(1 + \exp M_c \beta_k)^2}{\exp M_c \beta_k} \right) \quad (2.15)$$

$$\frac{\partial \eta_k}{\partial \mu_k} = diag\left(\frac{(1 + \exp M_c \beta_k)^2}{\exp M_c \beta_k} \right) \quad (2.16)$$

2.2.2.3 Algorithme des scores de Fisher pour l'estimation des β

Comme β_k apparaît dans plusieurs termes de l'équation à résoudre ($W_{c\beta_k}, \frac{\partial \eta_{ck}}{\partial \mu_{ck}}, \mu_k$), il n'y a pas de solution analytique pour estimer β_k, on va donc utiliser une maximisation

itérative. On se base ainsi sur l'algorithme des scores de Fisher [McCullagh and Nelder (1989)]. Cette formule se base sur la formule de Taylor à l'ordre 1 :

$$f(a + \Delta x) = f(a) + f'(a)\Delta x + o(\Delta x)$$

où $a = \beta_k^{(m)}$, $\Delta x = (\beta_k^{(m+1)} - \beta_k^{(m)})$, $f(a) = f(\beta_k^{(m)}) = \frac{\partial Q(\phi|\phi^m)}{\partial \beta_k}$ et $f'(a) = \frac{\partial^2 Q(\phi|\phi^m)}{\partial^2 \beta_k}$. En reprenant la formule de Taylor à l'ordre 1, on obtient :

$$\begin{aligned}
& f(\beta_k^{(m+1)}) = f(\beta_k^{(m)}) + (\beta_k^{(m+1)} - \beta_k^{(m)}) f'(\beta_k^{(m)}) + o(\beta_k^{(m+1)} - \beta_k^{(m)}) \\
\Leftrightarrow \quad & 0 = \frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k^{(m)}} + (\beta_k^{(m+1)} - \beta_k^{(m)}) \frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial^2 \beta_k^{(m)}} + o(\beta_k^{(m+1)} - \beta_k^{(m)}) \\
\Leftrightarrow \quad & 0 = \frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k^{(m)}} \left(\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial^2 \beta_k^{(m)}} \right)^{-1} + (\beta_k^{(m+1)} - \beta_k^{(m)}) + o\left((\beta_k^{(m+1)} - \beta_k^{(m)}) \left(\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial^2 \beta_k^{(m)}} \right)^{-1} \right) \\
\Leftrightarrow \quad & \beta_k^{(m+1)} = \beta_k^{(m)} - \left(\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial^2 \beta_k^{(m)}} \right)^{-1} \frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k^{(m)}} + o\left((\beta_k^{(m+1)} - \beta_k^{(m)}) \left(\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial^2 \beta_k^{(m)}} \right)^{-1} \right) \\
\Leftrightarrow \quad & \beta_k^{(m+1)} \approx \beta_k^{(m)} - \left(\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial^2 \beta_k^{(m)}} \right)^{-1} \frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k^{(m)}}
\end{aligned}$$

Ce qui différencie l'algorithme des scores de Fisher que l'on va utiliser de l'algorithme de Newton-Raphson, c'est que l'on calcule l'espérance de $\left(\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial^2 \beta_k}\right)^{-1}$ (on retrouve donc la formule de l'information de Fisher) au lieu de la simple dérivée seconde. Pour estimer β_k, on actualise donc la valeur à partir de l'équation qui suit :

$$\beta_k^{(m+1)} = \beta_k^{(m)} - \left(E\left[\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial \beta_k \partial \beta_{k'}} \right] \right)^{-1} \frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k} \tag{2.17}$$

Ce système d'équation peut être facilement réécrit en terme d'équations normales. On pose $z_k = M^t \beta_k + \frac{\partial \eta_k}{\partial \mu_k}(y - \mu_k)$. Ainsi, à partir de l'équation (2.17), on a :

$$\frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k} = \sum_{c=1}^{C} \pi_{ck}^{(m)} M_c^t W^{-1}_{c\beta_k^{(m)}} (z_{ck}^{(m)} - M_c \beta_k^{(m)}) \tag{2.18}$$

$$\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial \beta_k} = \frac{\partial}{\partial \beta_k} \left(\sum_{c=1}^{C} \pi_{ck}^{(m)} M_c^t W^{-1}_{c\beta_k^{(m)}} (z_{ck}^{(m)} - M_c \beta_k^{(m)}) \right) \tag{2.19}$$

$$= -\sum_{c=1}^{C} \pi_{ck}^{(m)} M_c^t W^{-1}_{c\beta_k^{(m)}} M_c \tag{2.20}$$

Au final, la mise à jour du paramètre β_k à l'itération $(m+1)$ dans le cadre d'un mélange de distributions binomiales s'écrit :

$$\beta_k^{(m+1)} = \beta_k^{(m)} - \left(E\left[\frac{\partial^2 Q(\phi|\phi^{(m)})}{\partial \beta_k \partial \beta_{k'}} \right] \right)^{-1} \frac{\partial Q(\phi|\phi^{(m)})}{\partial \beta_k} \tag{2.21}$$

$$= \beta_k^{(m)} + \left(\sum_{c=1}^{C} \pi_{ck}^{(m)} M_c^t W^{-1}_{c\beta_k^{(m)}} M_c \right)^{-1} \sum_{c=1}^{C} \pi_{ck}^{(m)} M_c^t W^{-1}_{c\beta_k^{(m)}} \frac{\partial \eta_{ck}}{\partial \mu_k} (Y_c - \mu_k) \tag{2.22}$$

$$= \left(\sum_{c=1}^{C} \pi_{ck}^{(m)} M_c^t W^{-1}_{c\beta_k^{(m)}} M_c \right)^{-1} \sum_{c=1}^{C} \pi_{ck}^{(m)} M_c^t W^{-1}_{c\beta_k^{(m)}} \underbrace{\left[M_c \beta_k^{(m)} + \frac{\partial \eta_{kc}}{\partial \mu_k} (Y_c - \mu_k) \right]}_{z_{ck}^{(m)}} \tag{2.23}$$

Cette réécriture permet de résoudre de manière itérative la mise à jour des paramètres β du GLM. À l'aide du vecteur β_k à l'itération (m), il est possible de calculer respectivement les matrices W_{β_k} et le vecteur des données de travail z_k.

2.2.3 Développement d'un package R : *binomialMix*

Ce modèle de mélange pour données répétées et de distribution binomiale est implémenté et disponible dans le package R *binomialMix* et détaillé via l'algorithme [1]. Le critère d'arrêt proposé pour cette implémentation se base sur l'écart en valeur absolue des paramètres du modèle à un temps (m) et $(m+1)$. De plus, pour assurer une convergence en pratique du modèle, le nombre d'itération de l'EM est limité à 30. De manière analogue, le critère d'arrêt de l'algorithme des scores de Fisher, utilisé pour l'étape M de l'EM, se base sur la différence en valeur absolue des paramètres β. L'appendice (A) permet d'avoir un aperçu de l'utilisation du package avec une vignette consacrée à sa description et prise en main.

2.3 Expérimentations

2.3.1 Résultats de simulation

Avant d'évaluer le modèle sur données réelles, nous effectuons une étude de simulation en deux étapes : dans un premier temps, nous essayons de trouver la bonne partition lorsque nous connaissons le modèle. Il s'agit dans cette première étape de valider le modèle de mélange implémenté via l'algorithme [1]. Dans un second temps, nous évaluerons le modèle dans le cadre de simulations plus proches des données réelles.

Évaluation de l'algorithme EM pour un mélange de distribution binomiale

Nous évaluons la capacité de notre approche à trouver la bonne partition lorsque le modèle linéaire généralisé est connu.

Algorithm 1: Algorithme EM pour données répétées de distribution binomiale

Data: Données binomiales
Fixer $\epsilon_1 > 0$, $\epsilon_2 > 0$ et $K > 1$;
$CRIT_1 \leftarrow 1$;
$m \leftarrow 1$;
$it \leftarrow 1$;
Initialiser arbitrairement les paramètres du modèle $\phi^{(m)} = (\beta^{(m)}, \lambda^{(m)})$;
while $(CRIT_1 > \epsilon_1)$ *and* $(it < 30)$ **do**
  $\forall k, c\ \pi_{kc}^{(m+1)} \leftarrow \frac{\lambda_k^{(m)} f_{\phi_k^{(m)}}(y_c)}{\sum_{l=1}^{K} \lambda_l(m) f_{\phi_l^{(m)}}(y_c)}$;
  $\forall k\ \lambda_k^{(m+1)} \leftarrow \frac{1}{C} \sum_{c=1}^{C} \pi_{kc}^{(m+1)}$;
  for *(k in* $1 : K$*)* **do**
    $it_f \leftarrow 1$;
    $CRIT_2 \leftarrow 1$;
    while *(*$CRIT_2 > \epsilon_2$*) and (*$it_f < 10$*)* **do**
      $it_f \leftarrow it_f + 1$;
      $W_{c\beta_k^{(m)}}^{(m+1)} \leftarrow diag\left(\frac{1}{n_c} \frac{(1+\exp M_c\beta_k^{(m)})^2}{\exp(M_c\beta_k^{(m)})}\right)$;
      $z_{ck}^{(m+1)} \leftarrow M_c\beta_k^{(m)} + \frac{\partial \eta_{kc}}{\partial \mu_k}(Y_c - \mu_k)$;
      $\beta_k^{(m+1)} \leftarrow \left(\sum_{c=1}^{C} \pi_{ck}^{(m+1)} M_c^t W_{c\beta_k^{(m)}}^{-1} M_c\right)^{-1} \sum_{c=1}^{C} \pi_{ck}^{(m+1)} M_c^t W_{c\beta_k^{(m)}}^{-1} z_{ck}^{(m+1)}$;
      $CRIT_2 \leftarrow \max|\beta^{(m+1)} - \beta^{(m)}|$;
    end
  end
  $\phi^{(m+1)} = (\beta^{(m+1)}, \lambda^{(m+1)})$;
  $CRIT_1 \leftarrow \max|\phi^{(m+1)} - \phi^{(m)}|$;
  $m \leftarrow m + 1$;
  $it \leftarrow it + 1$;
end

Construction du plan d'expérience : le plan d'expérience est basé sur un sous-ensemble de la matrice des variables exploratoires issue des données réelles. Il y a **138 campagnes publicitaires**. Leur durée de diffusion est très variable pour chacune d'entre elles comme en témoigne la figure 2.5 qui représente la distribution du nombre de jours de diffusion par campagne pour l'ensemble des données utilisées pour la simulation.

Simulation du vecteur des paramètres β : nous simulons le ratio de clics pour les $C = 138$ campagnes publicitaires réparties en $K = 2$ jusqu'à $K = 6$ clusters. Pour cela, nous simulons le vecteur des paramètres du modèle (voir équation (2.24)) pour chaque cluster à partir d'une loi uniforme dont les paramètres correspondent à l'intervalle de taux de clics que l'on souhaite tester. Ainsi, la matrice des paramètres β est simulée uniformément pour 4 intervalles différents ($[0.4, 0.6]$, $[0.2, 0.5]$, $[0.1, 0.2]$ et $[0.01, 0.1]$) afin que nous puissions estimer l'impact du taux de clics dans la

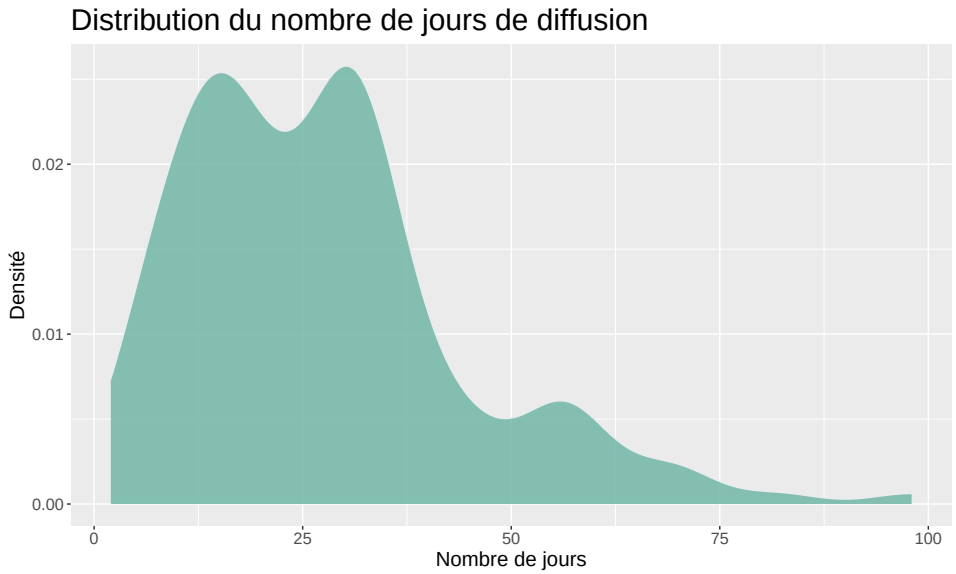


FIGURE 2.5 – Distribution du nombre de jours de diffusion pour l'ensemble des 138 campagnes du jeu de simulation étudié avec en abscisse un nombre de jours allant de 2 à 98 jours

modélisation.

Simulation du vecteur des valeurs de CTR : dans chaque cluster, le vecteur des valeurs du CTR (associé à la matrice des variables exploratoires) est simulé selon une distribution binomiale (voir équation (2.24)) avec 4 variables explicatives : jour de la semaine (7 modalités), plage horaire (6 modalités), type d'OS (2 modalités) et type de support (2 modalités) respectivement associés aux coefficients $\beta^S_{s(c,j)}$, β^H_h, β^{OS}_{os} et β^{AS}_{as}.

$$\log\left(\frac{E(Y_{cjh}/n_{cjh})}{1-E(Y_{cjh}/n_{cjh})}\right) = \beta_0 + \beta^H_h + \beta^S_{s(c,j)} + \beta^{OS}_{os} + \beta^{AS}_{as} \tag{2.24}$$

Nombre de simulations : pour chaque intervalle de valeur de CTR et pour chaque valeur de K simulée, 100 simulations ont été réalisées pour l'analyse de l'algorithme implémenté.

D'après la figure 2.6, le nombre de clusters est correctement estimé lorsque 2 ou 3 clusters sont simulés, et cela quelle que soit la probabilité du taux de clics. À partir de 4 clusters simulés, l'estimation du nombre optimal de clusters par critère BIC commence à se dégrader, d'autant plus quand la valeur de la probabilité de clic est faible. Il s'agit d'un comportement attendu du modèle car il y a moins de campagnes impliquées dans l'estimation des paramètres dans chaque classe. On peut également noter une tendance à la minoration du nombre de clusters dans le cas où la probabilité de clic est faible et à partir de 4 clusters simulés. L'indice de Rand ajusté est présenté dans la figure 2.7. Les conclusions sont pour la plupart identiques à celles concernant la figure 2.6. En effet, jus-

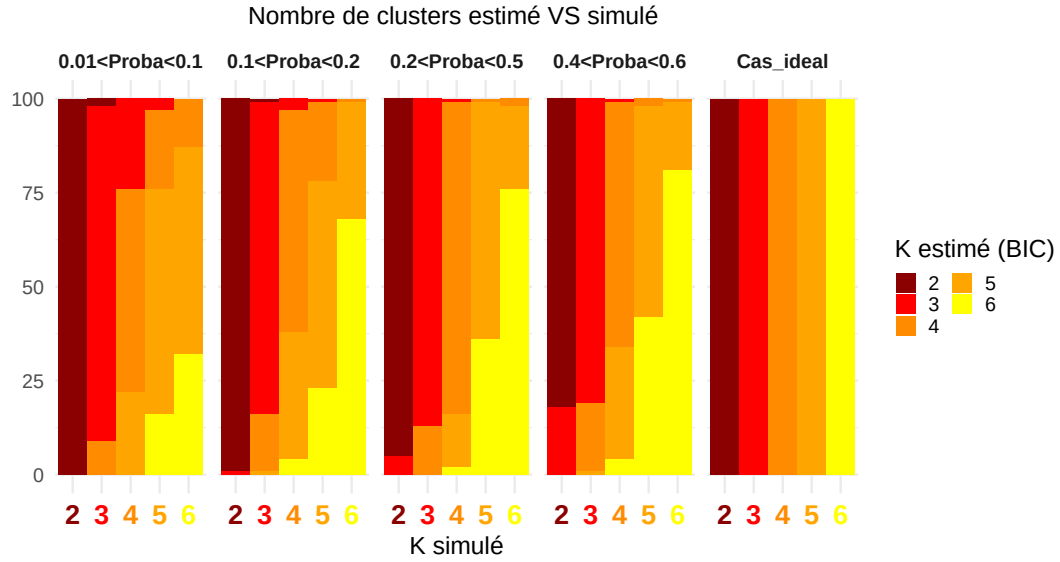


FIGURE 2.6 – Comparaison du nombre de clusters simulés et du nombre de clusters estimés par critère BIC. Chaque graphique correspond à une valeur de probabilité simulée. À droite, le cas idéal correspond à un nombre de clusters estimés égal au nombre de clusters simulés quel que soit le nombre de clusters.

qu'à 4 clusters simulés et pour un taux de clics moyen simulé supérieur à 0.2, la partition estimée est très proche de la partition simulée. La qualité d'estimation de la partition se détériore pour un taux de clics inférieur à 0.2, et cela même pour un petit nombre de clusters.

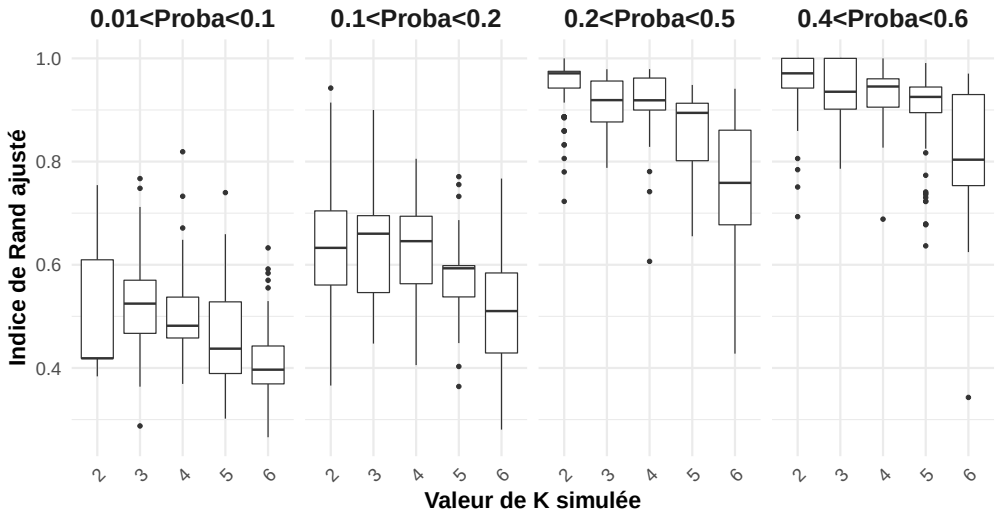


FIGURE 2.7 – Évaluation de la similarité entre la partition simulée et celle estimée à partir de l'indice de Rand ajusté pour un nombre de clusters K égal à celui simulé

2.3.2 Résultats sur données réelles

Sur données réelles, le taux de clics est très faible. On se situe à priori dans une zone d'estimation qui n'est pas idéale pour le modèle de mélange. Nous allons à présent regarder les résultats obtenus dans un cas réel. On considère le jeu de données construit au cours de la section 2.1.1 et les variables décrites dans cette même section. Le jeu de données contient 395 campagnes publicitaires pour l'apprentissage. Le modèle de mélange correspondant peut donc s'écrire sous la forme de l'équation (2.25). Le vecteur des paramètres à estimer β est de dimension $p = 20$.

$$\log\left(\frac{E(Y_{cjh}/n_{cjh})}{1 - E(Y_{cjh}/n_{cjh})}\right) = \beta_0 + \beta_h^H + \beta_{s(c,j)}^S + \beta_{os}^{OS} + \beta_{as}^{AS} + \beta_{ad}^{AD} + \beta_{si}^{SI} \tag{2.25}$$

avec β_0 le coefficient correspondant à la moyenne du modèle, β_h^H le coefficient associé à la variable catégorielle *Plage Horaire*, $\beta_{s(c,j)}^S$ associé au *Jour de la semaine*, β_{os}^{OS} associé au *Type de système d'exploitation*, β_{as}^{AS} associé au *Type de support*, β_{ad}^{AD} associé au *Type de format* et β_{si}^{SI} associé à la *Taille de la publicité*. Par souci de clarté, nous gardons l'indice cjh pour décrire le nombre de clics Y_{cjh} et d'impressions n_{cjh} qui devrait être également indicés par os, as, ad et si.

Choix du nombre optimal de clusters

Le nombre de clusters varie entre $K = 2$ et $K = 6$. Le choix du nombre optimal de clusters est calculé à partir du critère BIC décrit dans la section 1.5.2.3. Sur la figure 2.8, nous avons un tracé de ce critère pour les différentes valeurs de K. D'après l'heuristique de la méthode du coude, le nombre optimal de clusters correspond à la valeur $K = 5$.

En analysant conjointement les critères BIC et ICL (voir figure 2.9), le choix du nombre optimal de clusters est similaire pour les deux critères. Leurs valeurs numériques sont présentées dans la table 2.1. Nous voyons sur cet exemple que la différence entre les valeurs BIC et ICL sont minimes par rapport à leur ordre de grandeur. Le BIC comme l'ICL comportent un terme qui dépend de la logvraisemblance et un terme de pénalité (voir les équations (1.17) et (1.18)). Le critère ICL se base sur le critère BIC auquel on ajoute un terme supplémentaire lié à la matrice d'appartenance de chaque campagne à un cluster. Dans les 2 cas, le terme de pénalité reste largement inférieur au terme qui dépend de la logvraisemblance, et ne pénalise pas réellement la log vraisemblance.

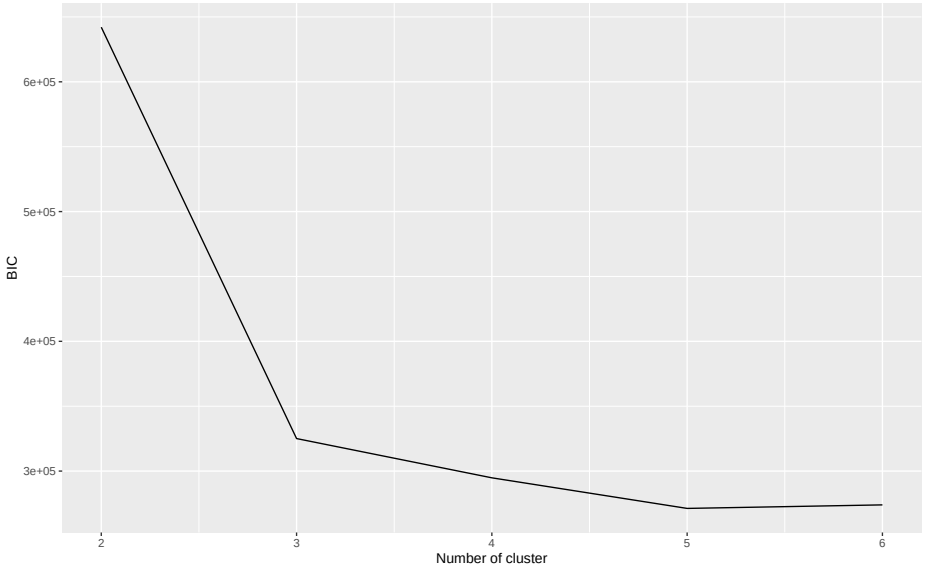


FIGURE 2.8 – Évaluation du critère BIC pour K allant de $K = 2$ à $K = 6$. Avec la méthode du coude, le choix du K optimal se situe au niveau de $K = 5$.

	BIC	ICL	|BIC-ICL|
K=2	642092.11	642221.91	129.80
K=3	325053.18	325233.62	180.44
K=4	294806.35	295044.77	238.42
K=5	271231.81	271545.47	313.66
K=6	273956.28	274340.60	384.32

TABLE 2.1 – Analyse des écarts de valeur entre les critères BIC et ICL sur un exemple donné.

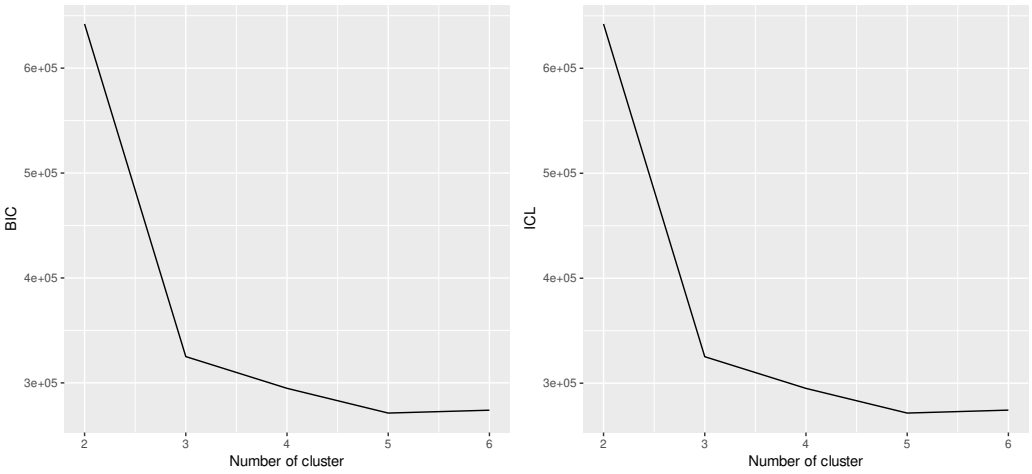


FIGURE 2.9 – Comparatif des résultats obtenus par critère BIC et ICL. Dans les deux cas, le choix du nombre optimal de clusters se porte sur $K = 5$.

Profils inférés

Nous présentons ici le résultat du modèle de mélange pour $K_{optimal} = 5$ comme présenté dans le paragraphe précédent. Les clusters comportent respectivement 39, 217, 29, 37 et 73 campagnes. Sur la figure 2.10, se trouvent les profils inférés pour chaque groupe de campagnes.

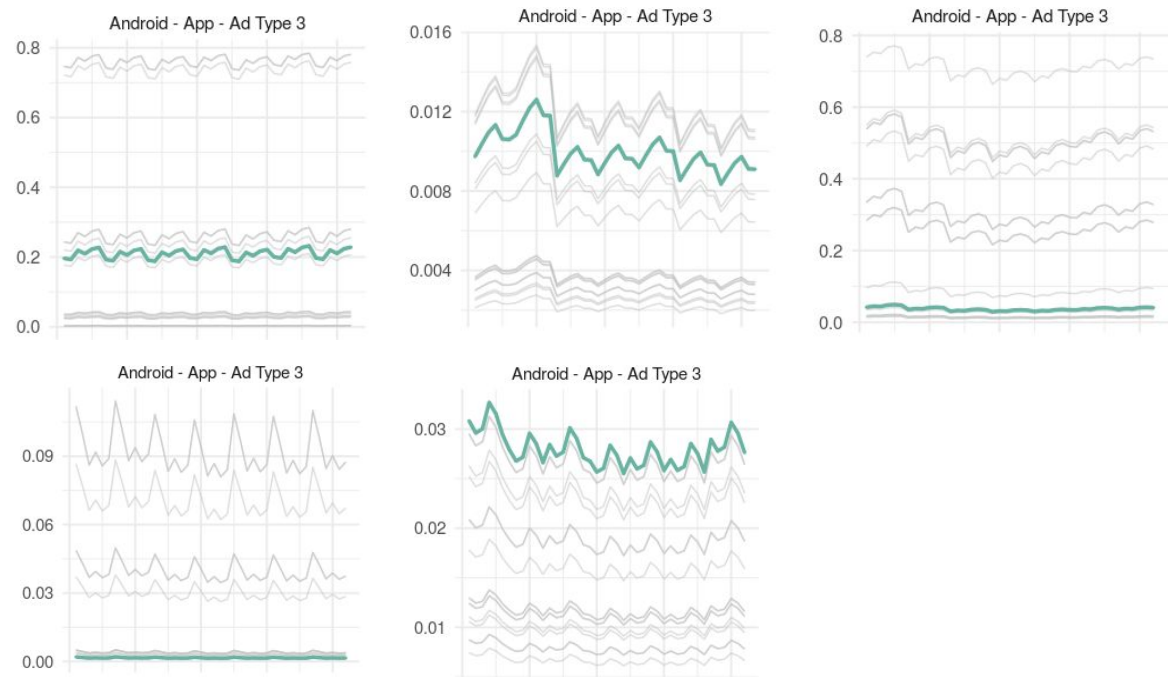


FIGURE 2.10 – Estimation des profils moyens pour chaque cluster lorsque le type de système d'exploitation est Android, le type de support est de type applicatif, le type d'annonce est de type 3 et la taille de l'annonce est de 320 x 480.

L'axe des abscisses correspond à la combinaison des variables *Jour de la semaine* et *Plage horaire* qui possèdent respectivement 7 et 6 modalités. L'axe des abscisses est donc découpé en 42 points de temporalité. L'axe des ordonnées représente le CTR moyen estimé (en pourcentage). Chaque graphique de la figure correspond à un cluster (de gauche à droite et de haut en bas, se trouvent les clusters 1 à 5) et représente la moyenne estimée au cours du temps et pour toutes les combinaisons de variables possibles. Le profil en vert et surligné correspond ainsi, à titre d'exemple, à la combinaison suivante :

Type d'OS : Android

Type de support : Application

Type de publicité : Type 3

Type de format : Bannière (320x480)

L'objectif est de comparer cette configuration dans les différents clusters. L'échelle du CTR sur l'axe des ordonnées diffère d'un groupe à l'autre : pour le 1er cluster, le CTR moyen se situe aux alentours de 0.2 alors que pour les autres clusters, le CTR moyen

estimé est en dessous de 0.1. On voit bien que dans ce contexte donné, l'évolution du CTR moyen est très différente d'un groupe de campagnes à l'autre. Le premier cluster semble avoir une saisonnalité au niveau des valeurs de CTR selon leur plage horaire alors que le deuxième cluster semble quant à lui avoir des différences de CTR bien marquées d'un jour de la semaine à un autre. Ce type de graphique permet de visualiser le résultat du modèle de mélange mis en place dont l'objectif était d'aboutir à des regroupements de campagnes dont les différences d'un cluster à l'autre étaient bien marquées.

Sur la figure 2.11, 5 profils sont représentés pour deux clusters différents (choisis de manière aléatoire). La ligne en pointillé correspond aux profils dont le type de support est de type *Application* tandis que les lignes en trait continu sont pour les types de support *Site*. Le dégradé de rouge représente les profils de type Android et les lignes dégradées de bleu sont pour les profils de type iOS.

Le **cluster 1** regroupe les campagnes ayant un CTR plutôt élevé, surtout pour les *Site web* et les publicités de *Type 3*.

Le **cluster 5** se compose quant à lui de campagnes qui se démarquent principalement sur les supports de type *Application* sans réelles distinctions notables suivant la famille de système d'exploitation ou de publicité.

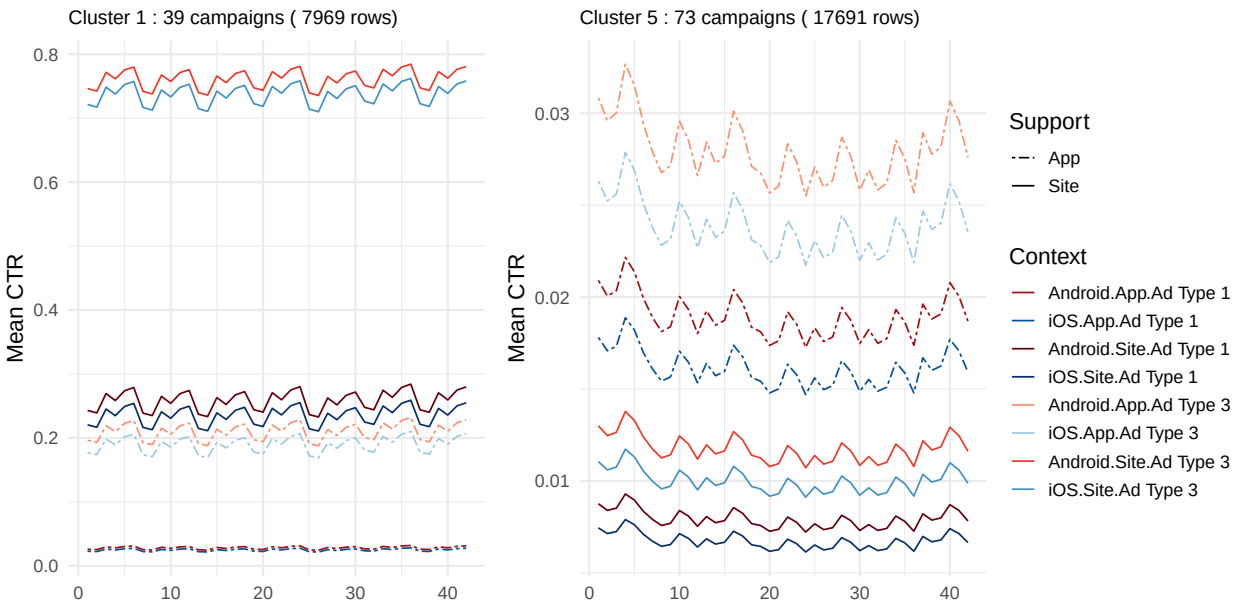


FIGURE 2.11 – Profils inférés pour les clusters 1 et 5. CTR élevé pour les campagnes du cluster 1, en particulier pour les support de type site et des publicités de type 3. Cluster 5 composé de campagnes dont le CTR varie principalement suivant la fonctionnalité App ou Site, quel que soit le type de publicité et la famille de système d'exploitation.

En conclusion, les clusters obtenus ont permis de regrouper des campagnes ayant des valeurs moyennes de taux de clics similaires et caractéristiques selon les profils observés.

Cette étape de clustering sera par la suite utilisée comme socle de départ des modèles prédictifs.

2.4 Perspectives

Au cours de ce chapitre, nous avons mis en place une méthodologie de classification non supervisée à partir d'un modèle binomial. Cette modélisation nous a permis de regrouper des campagnes publicitaires ayant des comportements similaires. Le clustering s'est basé sur un certain nombre de variables catégorielles : calendaires, relatives à l'emplacement publicitaire et enfin à la publicité en elle-même. Plusieurs axes de travail et perspectives d'amélioration sont proposés dans la suite de cette section.

2.4.1 Etape d'initialisation de l'algorithme EM

Revue sur l'étape d'initialisation de l'algorithme EM

Nous avons fait le choix d'un algorithme EM classique. Les extensions de cet algorithme sont nombreuses comme en témoigne la riche littérature à ce sujet. Plusieurs alternatives à l'étape d'initialisation de l'algorithme EM ont été recensées par [Baudry and Celeux (2015)]. En effet, la solution d'un algorithme EM peut s'avérer extrêmement dépendante de son point de départ. Nous pouvons citer ici quelques algorithmes qui portent une importance particulière à l'initialisation :

- l'algorithme **Small EM** décrit par [Biernacki et al. (2003)] et qui consiste à lancer un certain nombre de fois l'EM mais avec peu d'itérations (sans qu'il y ait forcément convergence). Une fois ces itérations effectuées, on choisit celle qui maximise la logvraisemblance comme point de départ pour l'algorithme EM.
- l'algorithme **Classification EM** (CEM) décrit par [Celeux and Govaert (1992)] et qui correspond à un algorithme EM dont l'objectif principal est le clustering.
- l'algorithme **Stochastic EM** (SEM) introduit par [Celeux et al. (1995)] où une étape intermédiaire entre les étapes E et M est ajoutée. À l'issue de l'étape E, on affecte chaque individu à un cluster selon une probabilité calculée à partir d'une loi multinomiale dont les probabilités correspondent aux valeurs de probabilité d'appartenance de chaque individu aux différents clusters.

À travers ces quelques exemples d'extensions possibles connus dans la littérature, nous voyons que les améliorations possibles de l'étape d'initialisation de l'algorithme EM sont très nombreuses. Cette étape n'était pas centrale dans la mise en place du modèle de mélange pour aboutir à un clustering des campagnes publicitaires. Cependant, l'amélioration de l'étape d'initialisation reste un axe de travail assez naturel pour la suite.

2.4.2 Sélection du nombre de clusters pour le mélange

Dans le cadre des expérimentations en temps réel sur la plateforme d'enchère (dont les étapes et résultats sont détaillés au chapitre 4), nous avons dans un premier temps testé le modèle de mélange sur un nombre de clusters allant de $K = 2$ à $K = 6$.

La méthode du coude

Le critère choisi pour déterminer le nombre de clusters optimal est la méthode du coude. Il s'agit d'une heuristique qui se base sur l'évolution du critère tel que le BIC ou le ICL en fonction du nombre de classes k. La présence d'une rupture importante dans cette décroissance aide au choix du nombre de classes. Nous avons ainsi voulu automatiser ce critère. On note $X_k = \{2, \ldots, K\}$ l'ensemble des valeurs de k testées et $(v_k)_{k \in X_k}$ le vecteur des valeurs de BIC (ou ICL) obtenues pour les différents nombres de classes k de X_k. La valeur de k optimale est définie par :

$$K_{opt} = \arg\max_{k' \in X_k} \{|v_{k'} - v_{k'+1}|\} + 1 \tag{2.26}$$

Mais il existe d'autres critères pour choisir le nombre de classes optimal, comme le critère silhouette introduit par [Rousseeuw (1987)] qui calcule à la fois la distance inter classe et intra classe pour évaluer si les clusters sont bien séparés et homogènes. Nous gardons cela en tête pour de prochaines améliorations et tests du choix optimal de clusters.

Nombre de clusters à tester

Le vecteur du nombre de classes à tester a d'abord été défini de manière empirique au regard de la quantité de données présentes dans le jeu d'apprentissage. Le choix initial s'est ainsi porté sur l'intervalle $k \in \{2, 3, 4, 5, 6\}$. Après plusieurs jours de test consécutifs sur différents jeux d'apprentissage, nous avons analysé le nombre de clusters optimal choisi par le modèle via le BIC/ICL. Sur la figure 2.12 se trouvent les résultats de 14 jours de tests consécutifs et le K_{opt} oscille entre les valeurs 5 et 6. Pour l'instant, nous avons donc décidé de poursuivre les analyses du modèle de mélange en le testant sur l'intervalle $k \in \{4, 5, 6\}$. Mais cet intervalle fait partie des paramètres à garder en tête pour de futures optimisations de l'étape de clustering.

2.4.3 Optimisation de l'algorithme EM implémenté dans le package R *binomialMix*

Le développement du package R *binomialMix*, qui implémente l'algorithme [1] de mélange pour données binomiales, s'inscrit dans une logique d'amélioration continue. Ce package a vocation à évoluer en fonction des modifications méthodologiques et des

↓ DATE	HOST	SERVICE	CONTENT
Sep 07 09:04:46.351	i-0ede19d607a452e1a	r-implementation	K chosen : 6
Sep 06 08:44:51.592	i-053d768754391d0a3	r-implementation	K chosen : 5
Sep 05 08:57:57.078	i-0e33f5f2cd646ebc3	r-implementation	K chosen : 5
Sep 04 09:00:47.060	i-0ca34fd260781243d	r-implementation	K chosen : 6
Sep 03 08:58:46.387	i-09d2caa6b0a5d4a53	r-implementation	K chosen : 6
Sep 02 08:59:15.556	i-0a875febe5b5dbf2c	r-implementation	K chosen : 6
Sep 01 08:48:48.378	i-067ef5ad346dcf14b	r-implementation	K chosen : 5
Aug 31 09:02:03.035	i-09fa5c2cc45e3fcbe	r-implementation	K chosen : 5
Aug 29 09:03:22.153	i-05447c7cf91bf5d88	r-implementation	K chosen : 5
Aug 28 09:07:18.293	i-0f1392f49c637aca9	r-implementation	K chosen : 5
Aug 27 08:58:00.069	i-09976dc2f79f55de2	r-implementation	K chosen : 6
Aug 26 09:03:19.218	i-06cf6659b8e3b9bab	r-implementation	K chosen : 6
Aug 25 09:04:43.668	i-0cac30e97885ac055	r-implementation	K chosen : 6
Aug 24 08:55:40.423	i-0bbccb26d26966ac9	r-implementation	K chosen : 6

FIGURE 2.12 – Analyse du nombre de clusters optimal K_{opt} sélectionné par critère BIC sur 14 jeux de tests différents.

contraintes de mise en production. Le package est notamment dépendant de packages R (issus du CRAN ou de la communauté *R-users*) qui, pour certains d'entre eux, ne sont plus (ou bientôt plus) maintenus et qui pourraient mettre en péril la pérennité de nos travaux. Par ailleurs, nous allons continuer d'étoffer nos expérimentations sur des cas limites pour voir si les critères d'arrêt de l'algorithme EM et de l'algorithme des scores de Fisher restent pertinents. Il serait peut-être intéressant de voir si le nombre d'itérations pour les scores de Fisher pourrait s'adapter et s'affiner au fil des itérations.

2.4.4 Amélioration du processus de prétraitement des données

Pour les travaux relatifs à ce chapitre et au suivant (chapitre 3), le prétraitement des données suit les étapes qui ont été décrites dans la construction du jeu de données (section 2.1.1) à partir des données brutes (présentées en section 1.3.2). Ce travail de recherche, construction de variables, discrétisation de données fait partie des enjeux à venir pour l'amélioration de la modélisation. Plusieurs pistes se dessinent déjà :

Utilisation de nouvelles variables contextuelles : nous avons décrit au cours de la section 1.3.2.1 la notion de *Bid Request*. Il s'agit d'une requête entrante contenant de l'information liée au contexte de l'emplacement publicitaire disponible. Actuellement, nous utilisons un certain nombre de variables. Mais, il est tout à fait envisageable d'en considérer de nouvelles pour améliorer le modèle. Il y a par exemple des informations concernant le nom de l'application ou du site web, qui en l'état est difficile à exploiter en raison du nombre de modalités possibles. On pourrait aussi travailler sur l'utilisation d'une variable de catégorisation des emplacements publicitaires.

Affinage des variables existantes : la variable concernant l'horaire de la journée est utilisée sous forme de plage horaire dans le modèle actuel. Une journée est ainsi

découpée en 6 périodes de 4h chacune. Ce choix a été établi en concertation avec les experts métiers et après analyse exploratoire des données relatives à l'heure de la journée. De la même manière, les formats publicitaires ont été regroupés en une variable à 4 modalités (voir section 2.1.2.3). Mais il serait intéressant d'analyser l'impact de la modification de ces regroupements sur la qualité de prédiction.

L'objectif de ce chapitre a principalement été motivé par la volonté de regrouper des campagnes publicitaires ayant des caractéristiques similaires afin d'obtenir de l'homogénéité à l'intérieur de chaque partition et de l'hétérogénéité d'une partition à l'autre. Il s'agit d'une étape préliminaire à l'objectif plus global qui concerne la prédiction du taux de clics.

Chapitre 3

Prédiction du CTR à partir du clustering

L'objectif principal de ce travail est de pouvoir améliorer le système d'enchère de la plateforme dédiée de TabMo en essayant de proposer pour chaque enchère disponible, en priorité des campagnes publicitaires qui ont une forte probabilité d'être cliquées.

3.1 Modèles en compétition

Cette section est consacrée à la présentation des différents modèles de prédiction qui ont été mis en place. Dans une première partie, nous présenterons des modèles dits *naïfs* qui serviront de référence. Par la suite, nous développons les modèles mis en place à partir des résultats du clustering obtenus dans le chapitre 2.

3.1.1 Deux modèles naïfs de référence

3.1.1.1 Modèle (A) - Une prédiction naïve : prédire un CTR nul en se basant sur le caractère déséquilibré de cette variable

Avant de définir un modèle de prédiction basé sur l'étape préliminaire du clustering (voir chapitre 2), l'idée est d'obtenir des prédictions en se basant sur les spécificités des données et les statistiques exploratoires qui en découlent. Tout d'abord, les données observées présentent un réel déséquilibre entre les valeurs de CTR égales à 0 et celles différentes de 0. Le CTR repose sur un ratio entre clics et non clics pour une campagne donnée et pour un contexte donné. Dès lors qu'il n'y a pas de clics observés, la valeur du CTR est nulle. La figure 3.1 est un exemple de données observées sur une période de 30 jours d'historique. Il y a 691 campagnes observées. Sur cette figure, 80% d'entre elles ont un $CTR = 0$ dans 50% ou plus des contextes observés. Cette statistique descriptive permet d'obtenir une première référence naïve en terme de prédictions du CTR. Prédire un CTR toujours égal à 0 donnerait des résultats acceptables en terme de qualité de prédiction. On considère l'hypothèse suivante : *le CTR est toujours égal à 0.* Pour un jeu de test donné, ce modèle naïf s'écrit : $\forall c \in 1, ..., C$, $\forall j \in 1, ..., J_c$, $\forall h \in 1, ..., H$

$$\frac{\hat{Y}_{cjh}}{n_{cjh}} = 0 \tag{3.1}$$

où J_c correspond au nombre de jours total où une campagne c a été observée, H correspond au nombre total de plages horaires observées, n_{cjh} le nombre d'impressions observées pour un contexte donné et $\hat{Y}_{cjh}$ le nombre de clics prédits.

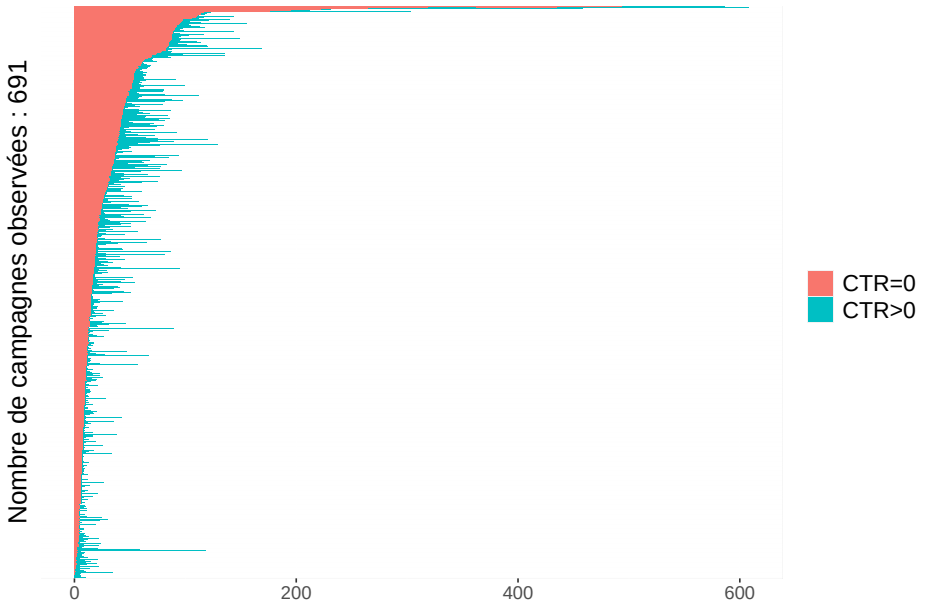


FIGURE 3.1 – Répartition par campagne des observations (contextes) où le CTR vaut 0 et où le CTR est strictement supérieur à 0

3.1.1.2 Modèle (B) - Une prédiction naïve : prédire le CTR par le CTR de la veille

On considère que pour chaque valeur de CTR à prédire, celui-ci est égal au dernier CTR observé pour la campagne et le contexte considéré. On note Y_{cjh} le vecteur correspondant à la variable réponse que l'on cherche à prédire. Le jour d'observation correspondant se note $j \in 1, ..., J_c$ avec J_c le nombre total de jours où la campagne a été observée. On note $h \in 1, ..., H$ la plage horaire observée. Ainsi, ce modèle naïf peut se définir :

$$\frac{\hat{Y}_{cj^{(t)}h}}{n_{cj^{(t)}h}} = \frac{Y_{cj^{(t-1)}h}}{n_{cj^{(t-1)}h}} \tag{3.2}$$

où $\frac{\hat{Y}_{cj^{(t)}h}}{n_{cj^{(t)}h}}$ est la valeur du CTR prédit pour la campagne c, sur la plage horaire h pour le jour de test $j^{(t)}$. Cette valeur prédite du CTR est ainsi égale à la valeur de CTR observée pour la même campagne et la même plage horaire lors du jour d'observation $j^{(t-1)}$. On peut supposer que le jour $j^{(t-1)}$ correspond aux données de la veille. Mais dans le cas où la campagne c ne possède pas de valeur de CTR observé pour cette plage horaire donnée h le jour précédent, la valeur $Y_{cj^{(t-1)}}$ correspond à la dernière observation du CTR pour cette campagne, cette plage horaire et ce même contexte.

3.1.2 Modèle (C) - Modèle linéaire généralisé de distribution binomiale

On considère un modèle linéaire généralisé (GLM) avec une distribution binomiale. Comme décrit dans l'équation (2.1), le nombre de clics Y_{cjh} suit une distribution binomiale :

$$Y_{cjh} \sim \mathcal{B}(n_{cjh}, p_{hs(c,j)})$$

avec n_{cjh} le nombre d'impressions associées pour la campagne c sur la plage horaire (j, h) et $p_{hs(c,j)}$ la probabilité de clic de la campagne c sur cette même plage horaire. En se référant aux notations de la section 1.5.1, on note β le vecteur des paramètres à estimer correspondant aux coefficients associés aux variables explicatives du modèle. L'estimation des paramètres d'un GLM est détaillée dans la section 2.2 et s'appuie sur l'algorithme itératif des scores de Fisher. Finalement, l'estimation du vecteur des paramètres β associés aux variables explicatives du GLM considéré s'écrit :

$$\beta^{(m+1)} = \left(\sum_{c=1}^{C} M_c^t W_{c\beta^{(m)}}^{-1} M_c \right)^{-1} \times \sum_{c=1}^{C} M_c^t W_{c\beta^{(m)}}^{-1} \left[M_c \beta^{(m)} + \frac{\partial \eta_c^{(m)}}{\partial \mu^{(m)}} \left(\frac{Y_c}{n_c} - \mu^{(m)} \right) \right] \tag{3.3}$$

Dans ce modèle décrit par l'équation (3.3), l'estimation du vecteur β est calculée à partir du jeu d'apprentissage et les prédictions se font à partir du vecteur β des coefficients de la manière suivante :

$$\frac{\hat{Y}_{cjh}}{n_{cjh}} = \frac{\exp\left(M_{cjh}^t \beta\right)}{1 + \exp\left(M_{cjh}^t \beta\right)} \tag{3.4}$$

avec M la matrice associée aux variables explicatives du modèle, n_{cjh} le nombre d'impressions observées.

Les deux premiers modèles naïfs (A) et (B) ainsi que ce dernier modèle (C) décrit à partir d'un GLM ont pour objectif de se comparer par la suite aux résultats des modèles prédictifs basés sur les résultats du clustering. Ce seront nos *modèles de référence*.

3.1.3 Des modèles prédictifs basés sur les résultats du clustering

Dans cette partie, on considère les résultats d'une classification obtenue à partir du modèle de mélange développé au cours du chapitre 2. Ainsi, à l'issue de cette étape préliminaire mais néanmoins importante, le modèle de mélange a permis d'obtenir l'estimation de ses différents paramètres :

La matrice des coefficients β associées aux variables du GLM : cette matrice se compose des coefficients β estimés pour chaque cluster du mélange. Elle se note B et est de dimension $p \times K$ où p correspond au nombre de paramètres à estimer et K le nombre de clusters.

Le vecteur λ des proportions du mélange : à l'issue de l'estimation des paramètres par l'algorithme EM, nous obtenons une estimation du vecteur λ de taille K correspondant aux proportions de chaque cluster.

La matrice Π des probabilités d'appartenance : en reprenant les mêmes notations, on a π_{kc} la probabilité que la campagne c appartienne à la composante k du mélange. On note Π la matrice de probabilité d'appartenance des C campagnes aux différents clusters k.

3.1.3.1 Modèle (D) - Prédiction à partir d'un mélange par affectation

On considère $\Pi^{(m)}$ de dimension $K \times C$ la matrice de probabilité d'appartenance des C campagnes aux K différents clusters :

$$\Pi^{(m)} = \begin{array}{c} \\ \mathrm{k}_1 \\ \mathrm{k}_2 \\ \vdots \\ \mathrm{k}_K \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & \dots & c_C \end{array} \\ \begin{pmatrix} \pi_{11}^{(m)} & \pi_{12}^{(m)} & \pi_{13}^{(m)} & \dots & \pi_{1C}^{(m)} \\ \pi_{21}^{(m)} & \pi_{22}^{(m)} & \pi_{23}^{(m)} & \dots & \pi_{2C}^{(m)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \pi_{K1}^{(m)} & \pi_{K2}^{(m)} & \pi_{K3}^{(m)} & \dots & \pi_{KC}^{(m)} \end{pmatrix} \end{array} \tag{3.5}$$

où $\sum_{k=1}^{K} \pi_{kc}^{(m)} = 1$, $\forall c \in 1, \dots, C$. Nous nous concentrons sur la matrice Π^{max} où l'indice max correspond à l'estimation de la matrice Π obtenue lors de la dernière itération de l'EM, i.e une fois que celui-ci a convergé.
On considère pour ce premier modèle de prédiction qu'une campagne ne peut appartenir qu'à un seul cluster à l'issue du mélange. Une campagne c est affectée au cluster k si :

$$\pi_{kc} \geq \max_{j \neq k}(\pi_{jc}) \tag{3.6}$$

La prédiction est calculée à partir de l'estimation des paramètres β du modèle de mélange. Pour toutes les campagnes affectées à un même cluster, l'estimation du vecteur β_k est la même.
On note B la matrice des paramètres β estimés pour les différents clusters $k = 1, \dots, K$, la matrice est de dimension $p \times K$ avec p le nombre total de paramètres à estimer et K le nombre total de clusters :

$$B = \begin{array}{c} \begin{array}{cccc} (k=1) & (k=2) & \dots & (k=K) \end{array} \\ \begin{pmatrix} \beta_0^1 & \beta_0^2 & \dots & \beta_0^K \\ \beta_1^1 & \beta_1^2 & \dots & \beta_1^K \\ \beta_2^1 & \beta_2^2 & \dots & \beta_2^K \\ \vdots & \vdots & \ddots & \vdots \\ \beta_p^1 & \beta_p^2 & \dots & \beta_p^K \end{pmatrix} \end{array}$$

Exemple illustratif : on se base sur les variables calendaires décrites dans la section 2.1.2.1. On considère que l'on a une proposition d'enchère qui arrive sur la plateforme. Cette enchère possède les caractéristiques suivantes :

Date : *12-04-20 13 :48 :54 UTC*

Type d'OS : *iOS*

Type de support : *site web*

Type de publicité autorisé : *banner*

Taille de l'emplacement publicitaire mis aux enchères : 320×50

De ces caractéristiques, on en déduit les modalités correspondantes pour les variables plage horaire h et jour de la semaine $s(c, j)$. Le calcul de la probabilité de clic pour cet espace publicitaire mis aux enchères est le suivant.
On définit C_k l'ensemble des campagnes affectées au cluster k. Ainsi, $\forall c \in C_k$:

$$\frac{\hat{Y}_{cjh}}{n_{cjh}} = \frac{\exp\left(\beta_0^k + \beta_{h=4}^k + \beta_{s(c,j)=0}^k + \beta_{os=iOS}^k + \beta_{as=site}^k + \beta_{ad=banner}^k + \beta_{size=320\times50}^k\right)}{1 + \exp\left(\beta_0^k + \beta_{h=4}^k + \beta_{s(c,j)=0}^k + \beta_{os=iOS}^k + \beta_{as=site}^k + \beta_{ad=banner}^k + \beta_{size=320\times50}^k\right)}$$

Plus généralement, la prédiction du clic s'écrit pour toute campagne $c \in C_k$ (issue du cluster k) :

$$\frac{\hat{Y}_{cjh}}{n_{cjh}} = \frac{\exp\left(M_{cjh}^t \beta_k\right)}{1 + \exp\left(M_{cjh}^t \beta_k\right)} \tag{3.7}$$

où M_{cjh}^t est le vecteur issu de la matrice des variables explicatives pour la campagne c dans un contexte donné. L'objectif est de choisir une des campagnes issues du cluster k, ce cluster devant être celui qui maximise la probabilité de clic.

3.1.3.2 Modèle (E) - Prédiction par une contribution pondérée de chaque cluster

Pour chaque campagne c, on considère le vecteur Π_c des probabilités d'appartenance à chacun des K clusters . Ce vecteur est issu de la matrice Π de dimension $K \times C$ où K est le nombre total de clusters pour la partition obtenue et C le nombre total de campagnes. On note Π la matrice des probabilités d'appartenance des C campagnes aux K différents clusters comme défini dans l'équation (3.5). La probabilité de clic pour toute campagne c est obtenue en pondérant la probabilité de clic associée à chaque cluster k par la probabilité d'appartenance de la campagne c à chaque cluster k. Ce calcul est répété pour tous les clusters k de la partition. La probabilité de clic finale est la somme pondérée des probabilités de clic spécifiques à chaque cluster et se définit de la manière suivante $\forall c \in 1, ..., C$:

$$\frac{\hat{Y}_{cjh}}{n_{cjh}} = \sum_{k=1}^{K} \pi_{kc} \times \frac{\exp\left(M_{cjh}^t \beta_k\right)}{1 + \exp\left(M_{cjh}^t \beta_k\right)} \tag{3.8}$$

où M^t_{cjh} est le vecteur issu de la matrice des variables explicatives pour la campagne c dans un contexte donné.
Contrairement à la méthode d'affectation présentée dans la partie 3.1.3.1 où une probabilité de clic est commune à toutes les campagnes affectées à un même cluster, ici une probabilité de clic est obtenue pour chaque campagne c puisqu'on en prend en compte son vecteur de probabilité d'appartenance π_c.

3.1.3.3 Modèle (F) - Prédiction à partir d'un mélange par affectation et d'effets aléatoires

On se base sur le modèle présenté dans la section 3.1.3.1 où toute campagne c est affectée à un cluster k. On considère à présent que les n_c observations d'une même campagne ne sont pas indépendantes. On définit ainsi un vecteur d'effet aléatoire ξ_c de dimension q afin de modéliser la dépendance entre observations de la campagne c. On note U la matrice des effets aléatoires associés de dimension $n \times q$ où q correspond à la dimension des effets aléatoires à estimer et n le nombre d'observations dans le jeu de données.
On considère un cluster k et l'ensemble des campagnes associées à ce cluster C_k. Pour chaque cluster k, le vecteur ξ se décompose de la manière suivante :

un effet aléatoire ξ_c qui représente l'effet campagne et traduit la dépendance entre les observations d'une même campagne c. ξ_c est de dimension $q = C$ et on a $\xi_c \sim \mathcal{N}(0, \sigma_1^2)$. On ne cherche pas à connaître l'effet de chacune des C campagnes mais plutôt à estimer σ_1^2 la variabilité induite par cet effet.

un effet aléatoire $\xi_{e\times c}$ qui correspond à une des variables explicatives qualitatives suivantes : type d'OS, type d'application, type de publicité, type de format. Ce vecteur $\xi_{e\times c}$ est de dimension q' où $q' = C \times \#e$ où $\#e$ est le nombre de modalités de la variable explicative dont on souhaite étudier l'effet aléatoire conditionnellement à chaque campagne. On note ainsi $\xi_{e\times c} \sim \mathcal{N}(0, \sigma_2^2)$

On a donc pour tout cluster $k \in 1, ..., K$,

$$\forall c \in k,\ \eta_{c\xi} = M_{ck}\beta_k + U_c\xi \tag{3.9}$$

avec ξ l'ensemble des effets aléatoires du modèle (par exemple $(\xi_c, \xi_{e\times c})$) et η_ξ le prédicteur linéaire. Ce dernier est égal à $\xi_\eta = g(\mu_\xi)$ où μ_ξ correspond à l'espérance du vecteur Y des observations conditionnellement à ξ : $\mu_\xi = E(Y|\xi)$.

On compare plusieurs modèles en se basant sur l'équation (3.9) et les matrices des effets aléatoires des différents modèles testés sont présentées à la suite du descriptif de chaque modèle :

Modèle 1 : Effets aléatoires **campagne** dont le vecteur aléatoire associé est de taille

$q_1 = C$

$$\xi = \begin{pmatrix} \xi_{(c=1)} \\ \xi_{(c=2)} \\ \vdots \\ \xi_{(c=C)} \end{pmatrix}_{q_1 \times 1} \tag{3.10}$$

Modèle 2 : effets aléatoires **campagne** et **type d'OS par campagne** dont le vecteur aléatoire associé est de taille $q_2 = C + (3 \times C)$ (3 modalités pour la variable *type d'OS*)

Modèle 3 : effets aléatoires **campagne** et **type de support par campagne** dont le vecteur aléatoire associé est de taille $q_3 = C + (2 \times C)$ (2 modalités pour la variable *type de support*)

Modèle 4 : effets aléatoires **campagne** et **type de publicité par campagne** dont le vecteur aléatoire associé est de taille $q_4 = C + (4 \times C)$ (4 modalités pour la variable *type de publicité*)

Modèle 5 : effets aléatoires **campagne** et **type de format par campagne** dont le vecteur aléatoire associé est de taille $q_5 = C + (3 \times C)$ (3 modalités pour la variable *type de format*)

$$\xi = \begin{pmatrix} \xi_{(c=1)} \\ \xi_{(c=2)} \\ \vdots \\ \xi_{(c=C)} \\ \xi_{(c=1)\times(os=os_1)} \\ \xi_{(c=1)\times(os=os_2)} \\ \vdots \\ \xi_{(c=C)\times(os=os_3)} \end{pmatrix}_{q_2 \times 1} \quad \xi = \begin{pmatrix} \xi_{(c=1)} \\ \xi_{(c=2)} \\ \vdots \\ \xi_{(c=C)} \\ \xi_{(c=1)\times(as=app)} \\ \xi_{(c=1)\times(as=site)} \\ \vdots \\ \xi_{(c=C)\times(as=site)} \end{pmatrix}_{q_3 \times 1} \quad \xi = \begin{pmatrix} \xi_{(c=1)} \\ \xi_{(c=2)} \\ \vdots \\ \xi_{(c=C)} \\ \xi_{(c=1)\times(ad=type_1)} \\ \xi_{(c=1)\times(ad=type_2)} \\ \vdots \\ \xi_{(c=C)\times(ad=type_5)} \end{pmatrix}_{q_4 \times 1} \quad \xi = \begin{pmatrix} \xi_{(c=1)} \\ \xi_{(c=2)} \\ \vdots \\ \xi_{(c=C)} \\ \xi_{(c=1)\times(size=type_1)} \\ \xi_{(c=1)\times(size=type_2)} \\ \vdots \\ \xi_{(c=C)\times(size=type_3)} \end{pmatrix}_{q_5 \times 1}$$

On obtient un modèle linéaire généralisé mixte (GLMM) pour chaque cluster C_k de campagnes publicitaires obtenu par le modèle de mélange présenté dans le chapitre 2. Les effets fixes du modèle associés β_k sont ainsi réestimés. On obtient une prédiction pour chaque campagne appartenant au cluster C_k à partir de la prédiction du vecteur des effets aléatoires ξ (regroupant ξ_c et $\xi_{e\times c}$). La prédiction du taux de clics, pour chaque campagne et pour un contexte donné, est donc calculée de la manière suivante, $\forall c \in C_k$:

$$\frac{\hat{Y}_{cjh}}{n_{c^k jh}} = \frac{\exp\left(M_{cjh}^t \beta_k + U_{cjh}\xi_c\right)}{1 + \exp\left(M_{cjh}^t \beta_k + U_{cjh}\xi_c\right)} \tag{3.11}$$

3.2 Performance prédictive des modèles

On cherche ici à évaluer la qualité de prédiction des différents modèles décrits dans la section précédente.

3.2.1 Évaluation selon la logloss moyenne

La métrique utilisée pour analyser la qualité de prédiction est la logloss, définie par [Murphy (2012)]. Il s'agit d'une métrique populaire employée dans la littérature pour la prédiction du CTR. Cette métrique est une fonction de coût associée à l'erreur de prédiction, qui est utilisée dans le cas où la variable à prédire prend ses valeurs entre 0 et 1. La pénalité est d'autant plus forte que l'erreur est grande (les faibles erreurs impactent en réalité peu la logloss). On considère ici pour chaque contexte de chaque campagne observée le nombre de fois où une publicité a été cliquée (y_{cjh}) parmi toutes les fois où la publicité a été vue (n_{cjh}). On définit ainsi le nombre de *non clics* comme étant égal à $n_{cjh} - y_{cjh}$. La logloss se calcule ainsi pour chaque contexte observé et prédit :

$$LogLoss_{cjh} = -(y_{cjh} \log \hat{p} + (n_{cjh} - y_{cjh}) \log (1 - \hat{p})) \tag{3.12}$$

Pour bien comprendre ce critère, considérons les deux exemples suivants :

Cas 1 : nous cherchons à prédire une variable dont la valeur réelle vaut 1. Le vecteur des prédictions $\hat{p}$ se compose comme des valeurs suivantes : $\hat{p} = (0.7, 0.5, 0.3, 0.1, 0.01)$. Dans ce cas là, le calcul de la logloss (voir équation (3.12)) est équivalent au terme $-\log \hat{p}$

Cas 2 : dans un second temps, nous prédisons une variable dont la valeur réelle vaut 0. Le vecteur des prédictions $\hat{p}$ se compose comme des valeurs suivantes : $\hat{p} = (0.001, 0.01, 0.1, 0.5)$. Dans ce cas là, le calcul de la logloss est équivalent au terme $-\log (1 - \hat{p})$

Nous voulons analyser la valeur de la logloss pour les différentes prédictions obtenues dans les deux cas. Les résultats sont résumés sur la figure 3.2. Il est intéressant de se placer dans l'intervalle des prédictions représentée par la flèche sur l'axe des abscisses. En effet, la valeur de nos prédictions se retrouvent très majoritairement dans cet intervalle là. Tout d'abord, la valeur de la logloss est beaucoup plus élevée si l'on cherche à prédire un clic (valeur réelle vaut 1) car l'ordre de grandeur de nos prédictions reste très proche de 0 et donc loin de la valeur réelle. En revanche, le coût engendré par une prédiction dans cet intervalle fléché lorsqu'on veut prédire un non clic (valeur réelle vaut 0) est bien moindre. Le terme $\log (1 - \hat{p})$ est en réalité très faible par rapport à $\log \hat{p}$ pour des si petites valeurs de prédictions $\hat{p}$. En résumé, la logloss est une fonction de coût à minimiser : plus elle est proche de 0, meilleure sera la prédiction obtenue comparativement à la valeur réelle observée.

De l'équation (3.12), on en déduit l'équation (3.13) la valeur de la logloss moyenne obtenue pour l'ensemble des campagnes $c \in 1, ..., C$ observées dans leurs différents contextes

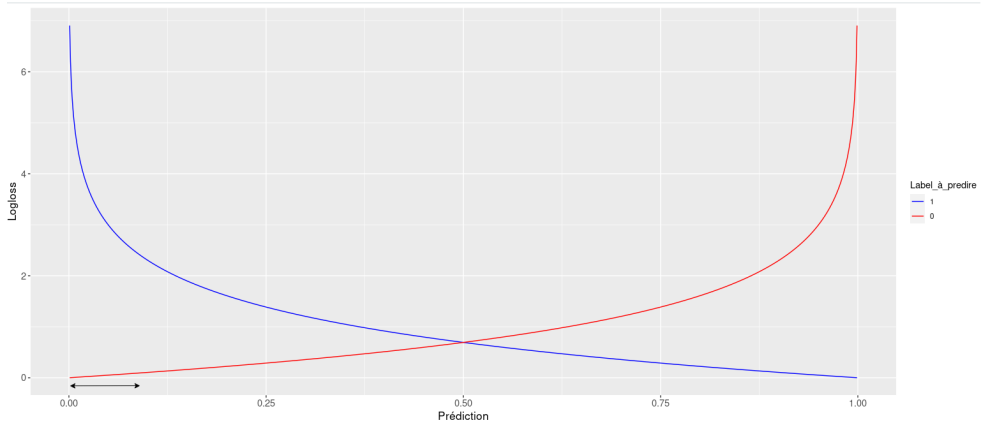


FIGURE 3.2 – Comparatif des valeurs de logloss pour différentes valeurs de prédiction et lorsque la valeur à prédire vaut 1 (en bleu) et 0 (en rouge).

j, h (voir équation (2.1.3)).

$$LogLoss = \frac{\sum_c \sum_j \sum_h LogLoss_{cjh}}{\sum_c \sum_j \sum_h n_{cjh}} \tag{3.13}$$

C'est avec ce critère de logloss moyenne que nous allons analyser la performance prédictive des modèles au cours des différentes expérimentations.

3.2.2 Contexte d'expérimentation

On définit un jeu d'apprentissage d'une durée de 30 jours et un jeu de test associé correspond au jour $j + 1$ faisant suite à la période d'apprentissage observée. On décale la fenêtre du jeu d'apprentissage et de test pour tester le modèle sur plusieurs jours distincts consécutifs. Trois périodes de l'année 2019 ont ainsi été extraites pour faire ces expérimentations : la première correspond au mois de septembre/octobre où le jeu d'apprentissage s'étend par période de 30 jours entre le 1er septembre 2019 et le 29 octobre 2019. Le jeu de test s'étend ainsi du 1er au 30 octobre. Par exemple : le jeu d'apprentissage du 1er septembre au 30 septembre est testé sur le jeu de test issu du 1er octobre. De manière analogue, une étude est également menée sur la période octobre/novembre 2019 ainsi que novembre/décembre 2019. Ainsi, pour les 30 jours de test de ces 3 périodes, nous calculons la logloss moyenne obtenue pour l'ensemble d'une période. Pour la suite, en se basant sur les modèles définis dans les sections 3.1.1 et 3.1.3, on note :

— Modèle A : prédire le CTR en se basant sur les statistiques exploratoires

— Modèle B : prédire le CTR à partir du CTR de la veille

— Modèle C : prédire avec un modèle linéaire généralisé de distribution binomiale

— Modèle D : prédire à partir d'un mélange par affectation

— Modèle E : prédire à partir d'une contribution pondérée de chaque cluster

— Modèle F : prédire à partir d'un mélange par affectation et d'effets aléatoires propres à chaque campagne. On note respectivement *Modèle F1*, *Modèle F2*, *Modèle F3*, *Modèle F4* les modèles avec effet campagne couplé aux effets aléatoires type d'OS, type de support, type de publicité ou type de format.

3.2.3 Résultats pour les modèles en compétition

Modèle (A)	Prédiction du CTR : toujours égal à 0
Modèle (B)	Prédiction à partir du CTR de la veille
Modèle (C)	Prédiction à partir d'un GLM simple
Modèle (D)	Prédiction à partir d'un mélange par affectation
Modèle (E)	Prédiction par une contribution pondérée de chaque cluster
Modèle (F)	Prédiction à partir d'un mélange par affectation et d'effets aléatoires

Nous comparons les résultats pour les modèles (A) à (F). Sur la table 3.1, la logloss moyenne a été calculée sur trois périodes de test distinctes. Prenons l'exemple des résultats obtenus pour le mois de novembre 2019 : les modèles prédictifs se basant sur les résultats du clustering ont des valeurs de logloss moyenne bien inférieures aux modèles plus naïfs que sont les modèles (A), (B) ou (C). Nous analysons ces résultats de manière plus

Logloss moyenne	(A)	(B)	(C)	(D)	(E)	**(F)**
October	0.32176	0.07463	0.07093	0.06394	0.06394	**0.06326**
November	0.43079	0.09758	0.08853	0.08162	0.08160	**0.08124**
December	0.38468	0.09395	0.08190	0.07474	0.07469	**0.07414**

TABLE 3.1 – Logloss moyenne obtenue pour les modèles décrits en section 3.1.1 et 3.1.3. Chaque colonne correspond à un modèle prédictif et nous avons 3 lignes correspondant aux 3 périodes de test.

détaillée via la figure 3.3 : les modèles naïfs (A) et (B) sont représentés en mauve, le modèle basique de distribution binomiale (C) en orange et les modèles se basant sur le clustering en vert. L'objectif est de classer les 6 modèles : le meilleur (*Rank 1*) est le modèle prédictif qui minimise la valeur de la logloss moyenne sur un jour de test et le (*Rank 6*) correspond au modèle présentant l'erreur de prédiction la plus grande quant à la logloss.
Très clairement, le modèle (A) arrive en dernière position sur tous les jours de test. Les modèles (B) et (C) se partagent les rangs 4 et 5. En revanche, les modèles (D), (E) et (F) forment presque systématiquement le top 3 pour ce qui est de la logloss sur les 30 jours de test. On peut même noter que le modèle (F), modèle avec effet aléatoire *campaign*, est celui qui minimise la logloss dans près de 80% des jours testés. Les modèles (D) et (E) occupent

les rangs 2 et 3 en majorité, avec le modèle (E) qui donne de meilleurs résultats. Pour rappel, ces deux modélisations, toutes deux basées sur les résultats de clustering, diffèrent par leur utilisation des résultats du clustering : la première affecte chaque campagne à un seul cluster tandis que la seconde se base sur l'appartenance pondérée de chaque campagne aux différents clusters. Les figures des résultats obtenus des modèles (A) à (F) pour les mois d'octobre et de décembre 2019 sont disponibles en Annexe (C). Pour ces deux périodes, les conclusions sont les mêmes. En octobre, le modèle (F) termine premier du classement sur les 30 jours de test tandis que le modèle (E) est deuxième meilleur modèle dans près de 90% des jours testés. Sur le mois de décembre, les modèles (E) et (F) se partagent également les deux premières places avec une très large proportion du modèle (F) en première position.

Sur l'ensemble de ces 6 modèles mis en compétition, cette première expérimentation a

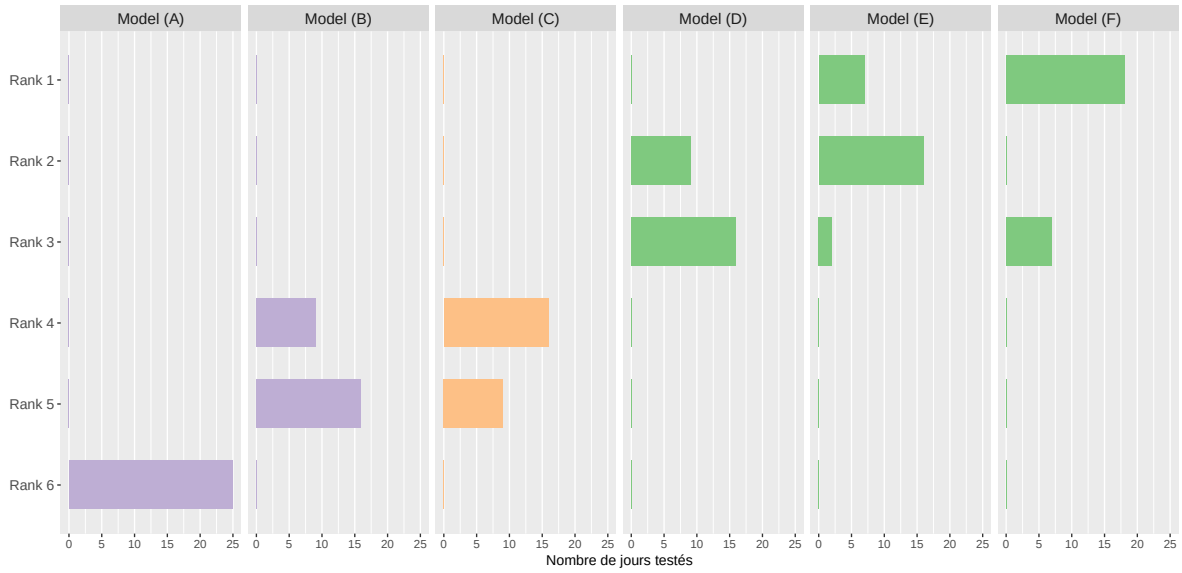


FIGURE 3.3 – Classement des 6 modèles (A),(B),(C),(D),(E) et (F) en compétition pour les 30 jours de tests du mois de Novembre 2019. Le *Rank 1* correspond au modèle dont la logloss est minimale tandis que le *Rank 6* correspond au modèle qui a la plus mauvaise logloss pour un jeu de test donné.

permis aux modèles (E) et (F) d'obtenir les résultats les plus intéressants avec une erreur de prédiction en terme de logloss parmi les plus faibles.

3.2.4 Comparatif pour les différents modèles avec effets aléatoires

Modèle (E)	Prédiction par une contribution pondérée de chaque cluster
Modèle (F)	Prédiction à partir d'un mélange par affectation et d'effets aléatoires
	(F) : *effet aléatoire campagne*
	(F1) : *effet aléatoire campagne et type d'OS par campagne*
	(F2) : *effet aléatoire campagne et type de support par campagne*
	(F3) : *effet aléatoire campagne et type de publicité par campagne*
	(F4) : *effet aléatoire campagne et type de format par campagne*

Pour affiner notre analyse nous comparons maintenant les modèles (E) et (F) avec les différentes déclinaisons du modèle (F) en termes d'effets aléatoires. En regardant les résultats obtenus quant à la logloss moyenne sur le mois de novembre (table 3.2), le modèle (F1) est celui qui obtient le meilleur résultat avec une logloss de 0.08081. Pour rappel, ce modèle correspond à une affectation de chaque campagne à un unique cluster. Au sein de chaque cluster, nous calculons un modèle mixte avec un effet aléatoire campagne ainsi qu'un autre lié au système d'exploitation (variable *os*) par campagne. Sur la figure 3.4, les

Logloss moyenne	(E)	(F)	**(F1)**	(F2)	(F3)	(F4)
October	0.06394	0.06326	**0.06269**	0.06321	0.06321	0.06322
November	0.08160	0.08124	**0.08076**	0.08119	0.08122	0.08121
December	0.07469	0.07414	**0.07301**	0.07396	0.07414	0.07414

TABLE 3.2 – Logloss moyenne obtenue pour les modèles décrits en section 3.1.1 et 3.1.3. Chaque colonne correspond à un modèle testé et les 3 lignes correspondent aux différentes périodes de test.

modèles sont classés du meilleur (*Rank 1*) au moins performant (*Rank 6*) pour les 30 jours de test du mois de novembre 2019. Le modèle (F1) est celui qui arrive en tête dans près de 90% des jours testés. L'ajout d'un effet aléatoire de type de système d'exploitation pour chaque campagne semble affiner et améliorer la prédiction du CTR. En revanche les modèles (F2) à (F4) ne semblent pas apporter une meilleure précision en matière de prédiction.

À l'issue de ces différentes expérimentations, le modèle (F1) est celui que nous allons utiliser pour la suite de notre travail. Pour rappel, ce modèle se base sur l'affectation des campagnes à un cluster et sur la prédiction d'effets aléatoires propres à chaque campagne (combinaison linéaire d'un effet campagne et d'un effet type de système d'exploitation par campagne).

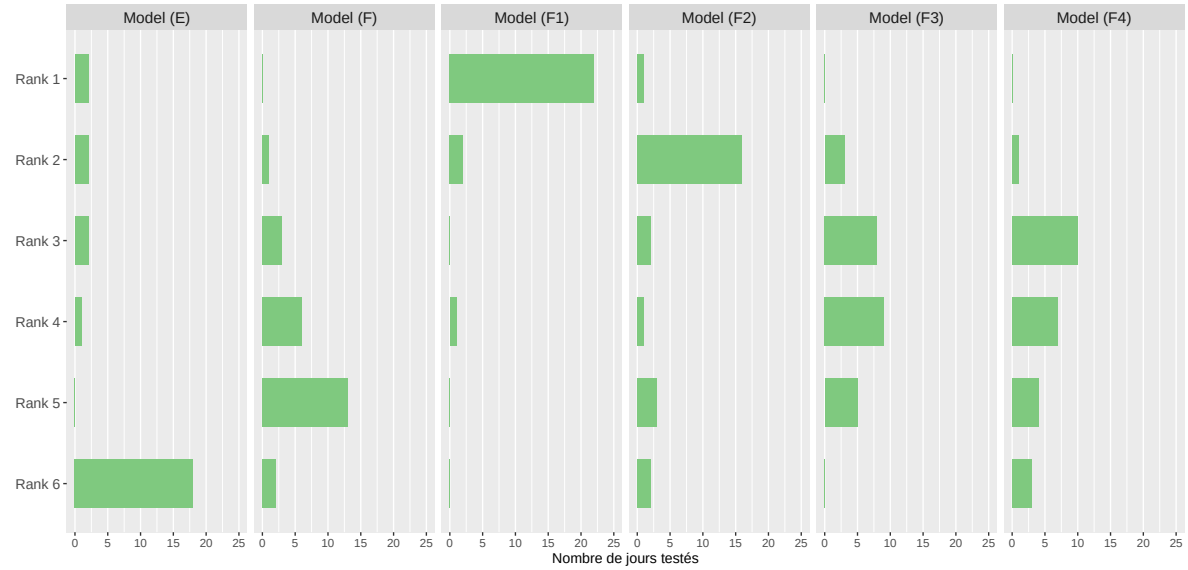


FIGURE 3.4 – Classement des 6 modèles (E),(F),(F1),(F2),(F3) et (F4) en compétition pour les 30 jours de test du mois de Novembre 2019. Le *Rank 1* correspond au modèle dont la logloss est minimale tandis que le *Rank 6* correspond au modèle qui a la plus mauvaise logloss pour un jeu de test donné.

3.2.5 Impact de l'étape d'initialisation de l'EM sur la prédiction

Nous faisons une étude rapide de l'étape d'initialisation de l'algorithme EM développé au cours du chapitre 2 quant à son impact sur la prédiction du taux de clics. Sur la figure 3.5, sont représentées les courbes d'évolution de la logvraisemblance pour 8 initialisations différentes d'un même jeu d'apprentissage et pour $K = 5$ clusters. Pour chaque initialisation, une valeur proche du maximum local est atteinte plus ou moins rapidement en termes d'itérations.

La table 3.3 permet d'analyser la valeur de l'indice de Rand ajusté entre les différentes partitions obtenues au cours des différentes itérations dont la définition est donnée dans l'équation (1.19). Plus la valeur de l'ARI est proche de 1, plus les partitions se ressemblent. Sur cette table, les partitions obtenues d'une initialisation à l'autre sont très différentes et ne regroupent pas les campagnes du jeu d'apprentissage de la même manière. Ce résultat peut paraître surprenant, nous poursuivons l'analyse.

On regarde maintenant les résultats de logloss obtenus pour un modèle de prédiction. Il s'agit du modèle (F1) basé sur l'estimation des paramètres du mélange et les clusters obtenus. Ce modèle est défini en détail dans la section 3.1.3.3. La logloss moyenne est une métrique permettant d'évaluer la performance prédictive d'un modèle. Elle est définie en détail dans la section (3.2.1) et prend ses valeurs entre 0 et 1. Ainsi, plus la logloss est proche de 0, meilleure sera la prédiction. Ces calculs sont effectués sur un même jeu de test pour toutes les initialisations (J+1 par rapport au jeu d'apprentissage) et résumés sur la table 3.4.

S'agissant des performances prédictives, nous obtenons des valeurs de logloss moyenne

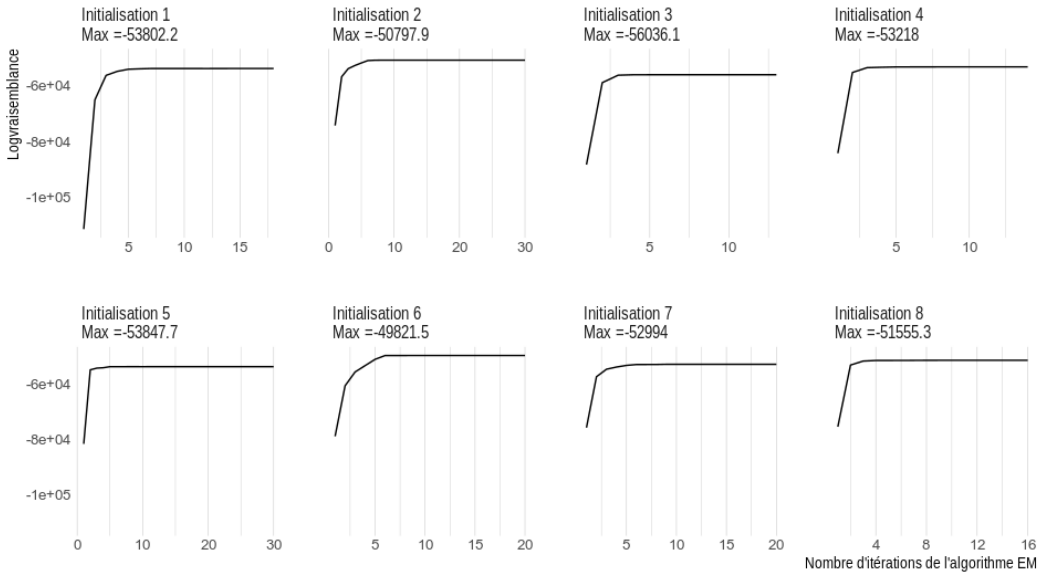


FIGURE 3.5 – Evolution de la logvraisemblance au cours des itérations de l'algorithme EM. 8 initialisations ont été lancées sur un même jeu de données afin de voir le maximum de vraisemblance obtenu pour chacune d'elle.

plutôt stables en dépit des différentes classifications. Même si pour certaines initialisations, nous sommes un peu en dessous du maximum local, les performances de prédiction ne semblent pas affectées. Contrairement à ce qu'on aurait pu penser, si l'initialisation de l'algorithme EM impacte fortement la classification (voir ARI table 3.3), l'impact reste minime sur la performance prédictive. Notre interprétation à posteriori est la suivante : l'étape de clustering joue le rôle d'étape préliminaire et permet d'améliorer la prédiction du CTR. Il n'y a pas un unique bon résultat de clustering du point de vue de la prédiction, mais un grand nombre de façons différentes de regrouper les campagnes publicitaires entre elles.

Faisons une analogie avec l'exemple d'une population de 30 individus que l'on souhaite classer en k groupes distincts à partir des informations suivantes : couleur des yeux, couleur des cheveux et longueur des cheveux. Ici, le clustering se fait sur des informations décrivant l'un des critères énoncés à l'exclusion des deux autres. Chacune des variables utilisées successivement va ainsi aboutir un découpage qui lui sera propre et qui sera probablement différent des autres découpages avec les autres variables. Pour autant, on obtiendra à chaque fois k groupes d'individus qui partagent une ou des caractéristiques communes. Ici, l'idée est la même. Nous ne souhaitons pas nécessairement savoir pourquoi ni dans quelles mesures les campagnes publicitaires sont regroupées mais plutôt s'assurer que le partitionnement obtenu permet d'améliorer l'étape de prédiction que l'on a étudiée au cours du chapitre 3. Il serait compliqué d'obtenir un modèle par campagne puisque nous n'aurions pas assez de données pour estimer correctement les paramètres de ces modèles. Le choix d'un modèle de mélange correspond ainsi à une forme de réduction

	Init 1	Init 2	Init 3	Init 4	Init 5	Init 6	Init 7	Init 8
Init 1	1.00	0.13	0.13	0.22	0.26	0.04	0.29	0.15
Init 2	-	1.00	0.13	0.06	0.13	0.22	0.22	0.15
Init 3	-	-	1.00	0.22	0.15	0.07	0.06	0.27
Init 4	-	-	-	1.00	0.15	0.09	0.05	0.16
Init 5	-	-	-	-	1.00	0.13	0.17	0.17
Init 6	-	-	-	-	-	1.00	0.15	0.24
Init 7	-	-	-	-	-	-	1.00	0.14
Init 8	-	-	-	-	-	-	-	1.00

TABLE 3.3 – Analyse des partitions obtenues pour les 8 initialisations grâce à l'indice de Rand ajusté (ARI) : les partitions semblent très différentes d'une initialisation à l'autre puisque les différentes valeurs d'ARI sont inférieures à 0.5.

	Init 1	Init 2	Init 3	Init 4	Init 5	Init 6	Init 7	Init 8
Logloss moyenne	0.06412	0.06482	0.06462	0.06465	0.06436	0.06485	0.06466	0.06499

TABLE 3.4 – Logloss obtenue pour chacune des 8 initialisations de l'EM sur le même jeu de test à partir d'un modèle prédictif basé sur l'estimation des paramètres du mélange et des clusters obtenus.

de dimension et permet d'avoir un compromis entre un modèle par campagne et un seul modèle commun à toutes les campagnes comme décrit dans le modèle (C) (voir la Section 3.1.1).

3.3 Perspectives

3.3.1 Recherche de l'historique optimal

Sur toutes les expérimentations effectuées jusqu'alors, l'historique du jeu d'apprentissage est d'un mois. Cependant, nous avons initié quelques expérimentations pour comparer l'évolution des résultats prédictifs en fonction de la durée de l'historique d'apprentissage. C'est une piste de travail sur laquelle nous n'avons pas encore toutes les conclusions mais en voici les prémices. La table 3.5 récapitule la logloss moyenne obtenue pour les modèles (C), (D), (E) et (F1) sur 20 jeux de tests consécutifs en faisant varier la durée de l'historique d'apprentissage. Nous testons ainsi 5 fenêtres de temps : 7 jours, 14 jours, 28 jours, 56 jours et 84 jours.
Quel que soit l'historique, le modèle qui minimise le critère de logloss moyenne est le modèle (F1). L'ordre établi jusqu'à présent n'est donc pas impacté par la durée de l'historique. La figure 3.6 illustre aussi les mêmes observations : le modèle (F1) est quasiment

toujours le meilleur modèle parmi les 4 modèles analysés, et le modèle (C) qui est un simple GLM et qui n'utilise pas l'étape préliminaire de clustering possède les moins bons résultats prédictifs. Les modèles (D) et (E) se partagent le milieu du classement. En conclusion, la fenêtre d'historique choisie ne semble pas impacter la performance du modèle (F1) vis-à-vis des autres modèles.
En regardant plus en détail la table 3.5, on observe de légères variations de la valeur de la logloss pour le modèle (F1) suivant l'historique choisi. À partir des résultats obtenus sur cette table, ce travail sur la recherche de l'historique optimal se résume de la manière suivante :

1) Nous avons convenu d'une fenêtre standard de 28 jours avec les experts pour l'ensemble des modèles testés jusqu'à présent.

2) Sur ces résultats de test (20 jours), il semblerait que sur ce jeu de données, 14 jours d'historique soient suffisants. Mais la différence de logloss moyenne entre 14 et 28 jours reste très faible.

3) Cependant, la période de test choisie correspond au mois de Décembre, qui est le mois de l'année où se succèdent le plus de campagnes publicitaires en raison des fêtes de fin d'année. Il est donc tout à fait possible d'imaginer que selon la période de l'année, la fenêtre d'historique puisse s'adapter en fonction de la durée moyenne des campagnes à chacune des périodes. Si les performances du modèle sont préservées, réduire la fenêtre d'historique permettrait de réduire les temps de calcul, notamment pour l'étape M de l'algorithme [1].

	Model (C)	Model (D)	Model (E)	Model (F1)
7 jours	0.08182	0.07437	0.07424	0.07312
14 jours	0.08198	0.07410	0.07409	0.07264
28 jours	0.08249	0.07474	0.07465	0.07295
56 jours	0.08300	0.07568	0.07558	0.07302
84 jours	0.08304	0.07497	0.07483	0.07270

TABLE 3.5 – Logloss moyenne obtenue pour les modèles (C), (D), (E) et (F1) suivant la fenêtre d'historique du jeu d'apprentissage. Chaque logloss moyenne a été calculée sur 20 jours de test.

3.3.2 Vers un choix de modèle prédictif dépendant de chaque campagne

Nous avons constaté qu'en moyenne le modèle (F1) offre de meilleures performances prédictives, suivi du modèle (E). Cependant, plusieurs perspectives émergent naturellement de ces premières conclusions à propos du choix d'un modèle prédictif plutôt qu'un

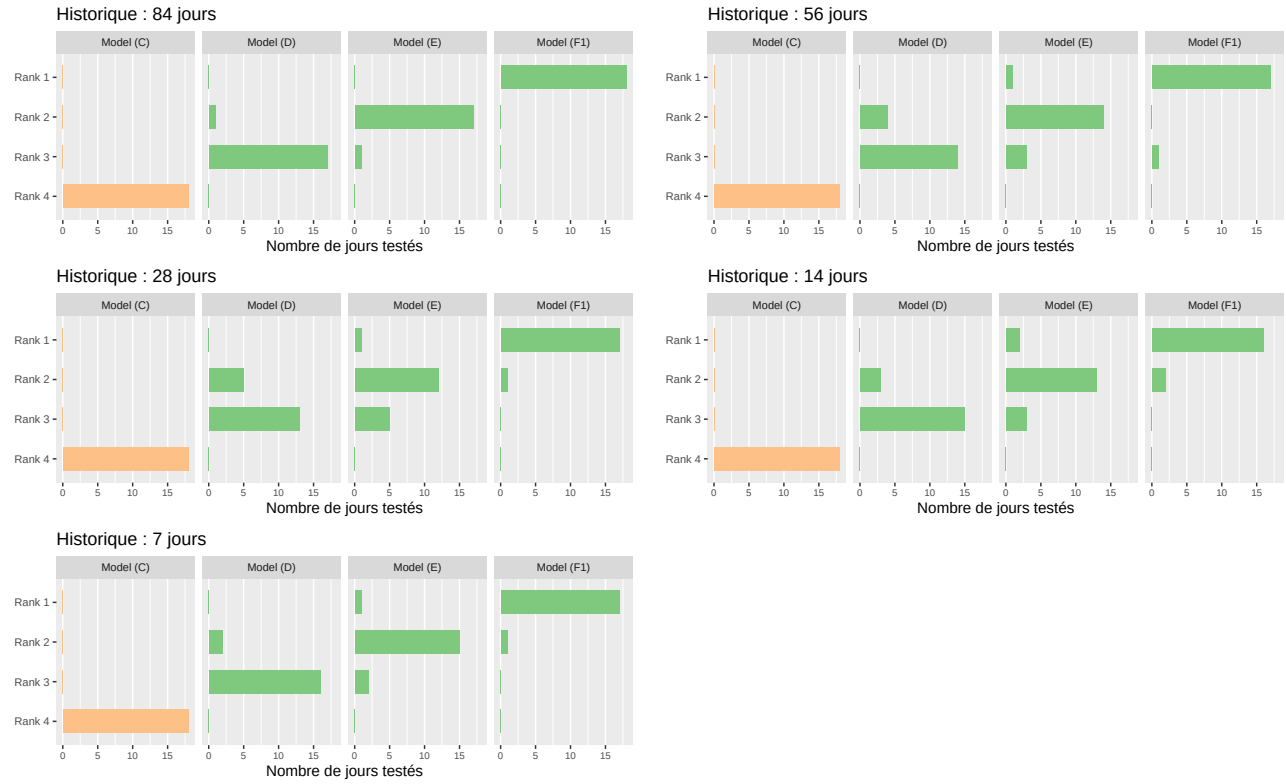


FIGURE 3.6 – Évaluation de la performance prédictive des modèles (C), (D), (E), (F1) sur 20 jours de test consécutifs du mois de Décembre 2019. Analyse des résultats en faisant varier la fenêtre d'historique du jeu d'apprentissage : 7, 14, 28, 56 et 84 jours.

autre en fonction des campagnes. Comme vu au cours de la section 3.1.3.3, le modèle (F1) affecte chaque campagne à un unique cluster.
Prenons l'exemple d'une matrice des probabilités d'appartenance π_{ex} de 3 campagnes à 2 clusters (k_1 et k_2) et définie de la manière suivante :

$$\begin{array}{c} \\ k_1 \\ k_2 \end{array} \begin{array}{c} \begin{array}{ccc} c_1 & c_2 & c_3 \end{array} \\ \begin{pmatrix} 0.54 & 0.92 & 0.27 \\ 0.46 & 0.08 & 0.73 \end{pmatrix} \end{array}$$

Les campagnes c_2 et c_3 sont affectées respectivement aux clusters 1 et 2 avec probabilité 0.92 et 0.73. En revanche, pour la campagne c_1, l'affectation à un unique cluster semble plus difficile à définir, le modèle (F1) affectera c_1 au cluster k_1 mais la contribution de cette campagne est quasiment équivalente pour les deux groupes. Dans ce cas, on peut se poser la question de l'utilisation du modèle (F1) de manière systématique : pourrions-nous plutôt utiliser le modèle (E), qui permet de calculer la prédiction à partir d'une contribution pondérée des différents clusters ? C'est toute la question qui se pose ici.

On peut aussi s'interroger sur la pertinence de l'utilisation du modèle (F1) dans le cas où une campagne a peu d'historique, comme le modèle (F1) définit un effet aléatoire

propre à chaque campagne afin de traduire la dépendance entre les observations d'une même campagne. Il est donc intéressant de se questionner sur l'historique nécessaire d'une campagne donnée afin d'évaluer si l'effet campagne est pertinent dans le calcul de la prédiction. On pourrait tout à fait imaginer un seuil minimal de durée de diffusion pour lequel on exclut systématiquement l'utilisation du modèle (F1) au profit du modèle (E).

Chapitre 4

De la modélisation statistique à la mise en production

4.1 Modèle probabiliste d'affectation d'une campagne à une enchère à partir de la prédiction de clics

4.1.1 Présélection des campagnes compatibles

L'exploitation de la probabilité de clic doit s'adapter à la structure du moteur d'enchère déjà en place. L'objectif est de pouvoir proposer une probabilité de clic pour chaque requête entrante compatible avec l'une des publicités présentes dans l'inventaire TabMo. Le processus de choix d'une campagne publicitaire pour laquelle la plateforme va proposer une enchère est décrit dans la figure 4.1 : chaque requête entrante fournit des caractéristiques sur l'emplacement publicitaire mis à l'enchère. En parallèle, un certain nombre d'annonceurs programment des campagnes publicitaires pour une durée de diffusion sur la plateforme et avec un budget donné. Chaque campagne publicitaire est caractérisée par un certain nombre d'éléments tels que sa taille ou son format. L'annonceur choisit en plus certains ciblages pour sa campagne comme la volonté de diffuser seulement à certaines heures de la journée ou encore sur un système d'exploitation spécifique, par exemple iOS ou Android. Ces critères spécifiques à chaque campagne publicitaire permettent de faire un premier tri des campagnes compatibles dès lors qu'une proposition d'espace publicitaire est reçue. On peut prendre l'exemple présenté dans la figure 4.1 :

Le *Filtre 1* correspond à ce premier filtrage lié à la compatibilité des critères de ciblages et format/taille. Dans le schéma, la publicité nommée *Ad 4* est directement écartée puisque sa dimension diffère de celle requise d'après les caractéristiques de la requête entrante. Les *Filtres* $2, ..., F$ permettent de trier selon d'autres critères que nous ne détaillons pas ici.

À l'issue de cette étape de sélection et tri, se trouve un ensemble de campagnes ayant passé tous les filtres. On les appelle les candidates et c'est parmi elles que se fait le choix final. Un poids lié à différents critères tels que le budget ou le retard de diffusion est associé à chacune des candidates. Un choix aléatoire est donc effectué à partir de la valeur des poids. Ainsi dans notre exemple, les campagnes *Ad 2* et *Ad 3* ont respectivement des poids de 0.7 et 0.3. Il y a donc 70% de chance (contre

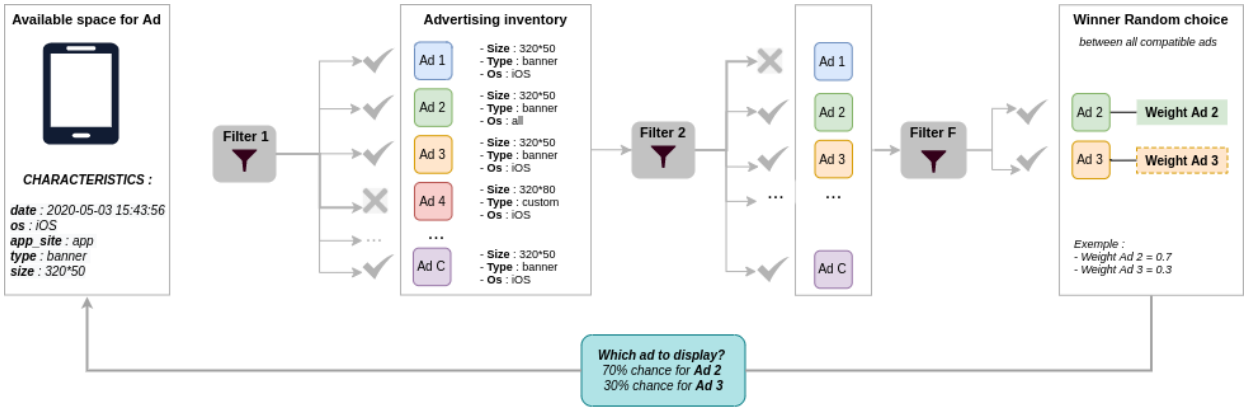


FIGURE 4.1 – Processus de choix d'une campagne publicitaire à afficher parmi toutes celles disponibles dans l'inventaire à partir des filtres sur les caractéristiques de l'enchère et l'espace publicitaire associé

30% pour la campagne *Ad 3*) que la campagne *Ad 2* soit celle qui sera proposée à l'enchère pour cet emplacement publicitaire.

En se basant sur ce processus de choix d'une candidate pour chaque requête entrante, les probabilités de clic sont pré-calculées à partir du modèle prédictif détaillé dans le chapitre 3. On définit un contexte $(\iota)_{1,\dots,L}$ défini par la combinaison des modalités possibles de chacune des variables catégorielles suivantes : un jour de la semaine (7 modalités), une plage horaire (6 modalités), un type de support (2 modalités), un type de système d'exploitation (3 modalités), un type de format (4 modalités) et une taille spécifique (3 modalités). On en déduit qu'il y a $L = 3024$ contextes distincts possibles par la combinatoire des variables présentées. Le modèle prédictif défini dans le chapitre 3 permet de calculer une prédiction pour chaque campagne c et pour chaque contexte L. Ainsi, une fois que le modèle s'est exécuté sur les données d'apprentissage, il est très facile de récupérer l'estimation des effets fixes associées aux variables contextuelles ainsi que l'estimation des effets aléatoires propres à chaque campagne. Cela nous permet alors de construire la matrice de prédictions suivante :

$$\begin{array}{c} \\ \iota_1 \\ \iota_2 \\ \vdots \\ \iota_L \end{array} \begin{array}{c} \begin{array}{ccccc} id_1 & id_2 & id_3 & \dots & id_C \end{array} \\ \begin{pmatrix} \hat{p}_{11} & \hat{p}_{12} & \hat{p}_{13} & \dots & \hat{p}_{1C} \\ \hat{p}_{21} & \hat{p}_{22} & \hat{p}_{23} & \dots & \hat{p}_{2C} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \hat{p}_{L1} & \hat{p}_{L2} & \hat{p}_{L3} & \dots & \hat{p}_{LC} \end{pmatrix} \end{array}$$

où $\hat{p_{\iota c}}$ correspond à la probabilité de clic obtenue par le modèle $(F1)$ pour la campagne c et pour le contexte ι.

4.1.2 Utilisation des matrices de quantile

L'exploitation des prédictions peut s'avérer compliquée lorsqu'il s'agit de comparer les valeurs de CTR d'une campagne à une autre.
Prenons l'exemple de trois campagnes c_1, c_2 et c_3 ayant les mêmes caractéristiques. On représente la valeur du CTR prédit pour chacune d'elles au cours du temps et pour un contexte ι fixé et défini par les variables *application, OS, type et format*. Chaque point de la figure 4.2 représente ainsi un créneau horaire pour un jour de la semaine donné. Comme nous avons 7 jours possibles et 6 plages horaires, la courbe de prédiction du CTR est construite sur 42 points de temporalité. En moyenne, la campagne c_1 a un CTR plus élevé que les deux autres campagnes concurrentes. Si on utilise la matrice des prédictions par contexte et par campagne que nous avons construite précédemment, la campagne c_1 sera choisie quel que soit le moment de la semaine observé (sauf pour la dernière temporalité) dans ce contexte donné puisque la probabilité $\hat{p}_{\iota c_1}$ est supérieure à $\hat{p}_{\iota c_2}$ et $\hat{p}_{\iota c_3}$. Si on regarde la prédiction pour le temps 5 représenté par la ligne en pointillé, on remarque que choisir la campagne c_1 ne serait pas le choix le plus judicieux puisqu'en comparaison avec les autres valeurs prédites de la campagne, il s'agit d'une probabilité basse. En revanche, regardons maintenant la campagne c_3. Il semble que dans le cadre de cet exemple et pour ce jour et cette plage horaire donnés, la prédiction obtenue soit meilleure à ce moment précis que la prédiction moyenne obtenue au cours des autres moments de la semaine. Il serait donc intéressant de privilégier cette campagne c_3 qui se trouve dans un contexte favorable bien qu'ayant une probabilité de clic inférieure à celle de c_1. De manière analogue, compte tenu de l'évolution du CTR au cours de la semaine pour ce contexte donné, il est plus judicieux de choisir la campagne c_2 pour la temporalité d'abscisse égale à 21 (deuxième ligne en pointillée). Pour ce contexte-ci, la campagne c_2 a un CTR légèrement inférieur à celui de la campagne c_1. Pour autant, en regardant l'évolution du CTR au global sur l'ensemble de la semaine, la campagne c_2 est un contexte très favorable comparativement à son CTR moyen.

Cet exemple montre qu'il n'est pas intéressant de comparer les prédictions comme des valeurs brutes. On va alors considérer la transformation du vecteur de prédictions d'une campagne $\hat{p}_{\iota c}$ en un vecteur de quantile $\hat{q}_{\iota c}$.

$$\begin{array}{c} \\ \iota_1 \\ \iota_2 \\ \vdots \\ \iota_L \end{array} \begin{array}{c} \begin{array}{ccccc} id_1 & id_2 & id_3 & \dots & id_C \end{array} \\ \begin{pmatrix} \hat{q}_{11} & \hat{q}_{12} & \hat{q}_{13} & \cdots & \hat{q}_{1C} \\ \hat{q}_{21} & \hat{q}_{22} & \hat{q}_{23} & \cdots & \hat{q}_{2C} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \hat{q}_{L1} & \hat{q}_{L2} & \hat{q}_{L3} & \cdots & \hat{q}_{LC} \end{pmatrix} \end{array}$$

Ainsi pour chaque exécution du pipeline global présenté sur la figure 4.3, une matrice des

FIGURE 4.2 – Analyse de l'évolution du CTR prédit au cours du temps pour 3 campagnes c_1, c_2, c_3 choisies aléatoirement. En pointillé (à gauche), le CTR est maximisé pour la campagne c_1 alors qu'en regardant son CTR au cours du temps, il s'agit d'un CTR bas pour cette campagne. La campagne c_3 semble quant à elle être dans un contexte qui lui est favorable avec un CTR plus haut que son CTR moyen.

quantiles est pré-calculée avec autant de colonnes C que de campagnes dans l'apprentissage et autant de lignes que de contextes possibles avec la combinaison des variables explicatives. Ce processus est programmé pour se lancer chaque jour à une heure prédéfinie. Toutes les 24h les paramètres du modèle de mélange et la matrice des quantiles liée aux prédictions sont mis à jour. L'objectif de cette actualisation quotidienne est principalement de pouvoir inclure de potentielles nouvelles campagnes publicitaires.

4.2 Intégration de la prédiction dans le calcul d'enchère

4.2.1 État des lieux de l'actuel fonctionnement

Le fonctionnement actuel de la plateforme d'enchère se base sur un vecteur de poids associé au budget alloué à chaque campagne pour choisir la campagne finale :

Étape 1 Une fois tous les filtres de compatibilité établis entre la requête entrante et les ciblages définis par le publicitaire, il reste un nombre C' de campagnes potentiellement candidates pour la requête.

Étape 2 Le système d'enchère se base ensuite sur deux informations principales pour choisir parmi les campagnes en compétition : le temps et le budget. Le temps permet de situer à quel niveau se situe la campagne vis-à-vis de sa durée de diffusion prévue et ainsi ajuster au mieux les dépenses jusqu'à la fin de sa période de diffusion. Le budget quant à lui correspond au budget restant de la campagne. Ce dernier est

recalculé toutes les 5 minutes afin d'éviter des dépenses trop conséquentes en peu de temps. Un calcul est ensuite effectué en utilisant la combinaison de ces deux paramètres. Il permet d'avoir un vecteur de poids pour les campagnes C' que l'on note $(b_i)_{i=1,...,C'}$ pour la suite. Ce dernier prend ses valeurs dans l'intervalle $[0; 50]$. Les valeurs correspondent au budget que l'on souhaite diffuser dans les 5 prochaines minutes. Un budget proche de 0 correspond à une campagne à court de budget et/ou de temps de diffusion. Et, plus la campagne a du retard, plus elle aura un budget journalier proche de 50. Cela permet de privilégier au maximum la diffusion de cette campagne pour les enchères où elle est compatible.

Étape 3 Une fois que le vecteur de poids est défini, le moteur d'enchère choisit aléatoirement parmi les C' campagnes en compétition en se basant sur le vecteur de poids normalisé comme probabilité pour chaque campagne $c \in C'$ d'être sélectionnée.

4.2.2 Utilisation des quantiles de prédictions

Prenons un exemple illustratif pour commencer. On considère trois campagnes publicitaires c_1, c_2 et c_3. On note (b_1, b_2, b_3) le vecteur des poids associé au budget pour chacune des trois campagnes. Ce vecteur prend ses valeurs dans l'intervalle $[0; 50]$. On note (q_1, q_2, q_3) le vecteur des quantiles de prédiction. Ces valeurs de quantile sont naturellement comprises entre 0 et 1. Les deux vecteurs sont normalisés par la somme respective de chacun d'eux. Dans cette première itération, nous accordons autant d'importance au budget qu'à la prédiction. De ce fait, le vecteur de poids associé à la probabilité finale de choisir l'une des trois campagnes de l'exemple s'écrit de la manière suivante :

$$(p_1, p_2, p_3) = 0.5 \times \frac{(b_1, b_2, b_3)}{\sum_{i=1}^{3} b_i} + 0.5 \times \frac{(q_1, q_2, q_3)}{\sum_{i=1}^{3} q_i}$$

Finalement, le choix de la campagne publicitaire à sélectionner pour l'enchère est calculé selon le dé à trois faces de poids (p_1, p_2, p_3). En considérant les valeurs suivantes : $(b_1, b_2, b_3) = (0.05, 5, 0.52)$ pour le budget et $(q_1, q_2, q_3) = (0.98, 0.25, 0.63)$ pour les quantiles de prédiction du CTR. On obtient ainsi le vecteur de poids final $(p_1, p_2, p_3) = (0.270, 0.49, 0.24)$. Un tirage aléatoire choisirait ainsi la campagne c_1, c_2 ou c_3 avec une probabilité respective de $0,27$, 0.49 ou 0.24.

À travers cet exemple illustratif, nous constatons que le choix de campagnes pour l'enchère en cours peut considérablement varier selon l'utilisation (ou pas) des quantiles de prédiction. Le vecteur des poids pour le budget indique clairement une volonté de diffuser la campagne c_2 tandis que le vecteur des quantiles de prédiction privilégie la campagne c_1 dont le contexte de la requête entrante semble plus favorable pour cette campagne-là. Il est donc intéressant de voir que le calcul du vecteur de poids final pour choisir la campagne gagnante est issue d'une combinaison linéaire entre les deux critères distincts que

sont le budget et le contexte de l'enchère pour l'obtention d'un clic.

4.2.3 Une matrice des quantiles de prédiction hybride

Comme expliqué à la fin du chapitre 3, le modèle retenu pour la mise en production est le modèle (F1), basé sur l'affectation des campagnes à un cluster et la prédiction d'effets aléatoires propres à chaque campagne (combinaison linéaire d'un effet campagne et d'un effet type de système d'exploitation par campagne). À l'issue de l'étape de clustering développée au cours du chapitre 2, chaque campagne est affectée à un unique cluster (voir équations (3.9)). Au sein de chaque cluster, l'ensemble des effets fixes liés aux variables explicatives du modèle sont ré-estimées et les effets aléatoires prédits. Cependant, selon la répartition des campagnes publicitaires présentes dans le jeu d'apprentissage, il est tout à fait possible qu'une modalité issue d'une des variables explicatives ne soit pas présente dans les observations associées à ce cluster.

Exemple illustratif : prenons le cas d'un clustering réparti de la manière suivante (table 4.1) : Dans cet exemple, le cluster 5 est composé de 26 campagnes. Il se trouve que

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
158	111	75	124	26

TABLE 4.1 – Exemple d'un résultat de clustering : répartition du nombre de campagnes par groupe dans le cas où $K = 5$

parmi l'ensemble des observations de ce groupe de campagnes, il n'y a ni impression ni clic pour la modalité *Type 2* issue de la variable explicative **Type de publicité**. Seules les modalités *Type 1*, *Type 3* et *Type 4* sont présentes dans ce sous-groupe. En conséquence, lorsque les prédictions sont calculées sur le jeu de test, les contextes dont le **Type de publicité** correspond au *Type 2* ne peuvent pas être prédits comme on peut le voir dans la table 4.2 où 3 contextes ne sont pas prédits pour cette campagne. Le GLMM calculé à partir des observations des campagnes issues du groupe 5 n'est pas en mesure d'estimer l'effet fixe lié à la modalité *Type 2*.

En reprenant les notations du l'équation (3.11) et à titre d'exemple, on détaille le calcul pour certaines lignes de la table 4.2, extrait du jeu de test. On considère la campagne c_1 au jour $j = 1$ et pour la plage horaire $h = 1$ où $c_1 \in C_{k=5}$. S'agissant de la première ligne de la table, la prédiction se calcule à partir de l'estimation des coefficients β où les variables explicatives décrivent le contexte suivant : $os = iOS$, $as = app$, $type = 1$ et $format = 1$.

$$\frac{\hat{Y}_{c_1jh}}{n_{c_1jh}} = \frac{\exp\left(\beta_0^5 + \beta_{h=1}^5 + \beta_{s(c_1,j)=1}^5 + \beta_{ios}^5 + \beta_{app}^5 + \beta_{type_1}^5 + \beta_{form_1}^5 + \xi_{c_1}^5 + \xi_{(c_1)\times(ios)}^5\right)}{1 + \exp\left(\beta_0^5 + \beta_{h=1}^5 + \beta_{s(c_1,j)=1}^5 + \beta_{ios}^5 + \beta_{app}^5 + \beta_{type_1}^5 + \beta_{form_1}^5 + \xi_{c_1}^5 + \xi_{(c_1)\times(ios)}^5\right)}$$

	ID	Jour	Horaire	OS	App/Site	Type	Format	*Prédiction*
1	c_1	1	1	iOS	app	Type 1	Format 1	*0.00364*
2	c_1	1	1	iOS	app	Type 1	Format 2	*0.00012*
3	c_1	1	1	iOS	app	Type 1	Format 3	*0.00107*
4	c_1	1	1	iOS	app	Type 2	Format 1	?
5	c_1	1	1	iOS	app	Type 2	Format 2	?
6	c_1	1	1	iOS	app	Type 2	Format 3	?
7	c_1	1	1	iOS	app	Type 3	Format 1	*0.00018*
8	c_1	1	1	iOS	app	Type 3	Format 2	*0.00159*
9	c_1	1	1	iOS	app	Type 3	Format 3	*0.00314*

TABLE 4.2 – Exemple de prédictions obtenues avec le modèle (F1) pour une campagne c_1 et pour différents contextes au sein d'un cluster. Le coefficient β associé au format de campagne de Type 2 n'est pas estimable dans ce sous-ensemble de campagne (cluster 5).

avec β^5 le vecteur des coefficients associés aux variables explicatives estimés dans le cluster 5, $\xi^5_{c_1}$ l'effet campagne pour la campagne c_1 et $\xi^5_{(c_1)\times(iOS)}$ l'effet aléatoire campagne associé à l'effet iOS.
De manière analogue, nous voulons calculer la prédiction pour la ligne 4 du jeu de test à partir de l'expression qui suit :

$$\frac{\hat{Y}_{c_1jh}}{n_{c_1jh}} = \frac{\exp\left(\beta^5_0 + \beta^5_{h=1} + \beta^5_{s(c_1,j)=1} + \beta^5_{ios} + \beta^5_{app} + \beta^5_{type_2} + \beta^5_{form_1} + \xi^5_{c_1} + \xi^5_{(c_1)\times(ios)}\right)}{1 + \exp\left(\beta^5_0 + \beta^5_{h=1} + \beta^5_{s(c_1,j)=1} + \beta^5_{ios} + \beta^5_{app} + \beta^5_{type_2} + \beta^5_{form_1} + \xi^5_{c_1} + \xi^5_{(c_1)\times(ios)}\right)}$$

Mais comme nous n'avons pas d'estimation du coefficient $\beta^5_{type_2}$, le calcul de la prédiction de ce contexte-là pour l'ensemble des campagnes de ce cluster n'est pas possible.

Cet exemple illustratif nous permet donc d'appréhender les limites potentielles du modèle (F1) en production. Les clusters établis sont dépendants du jeu d'apprentissage qui est actualisé toutes les 24h, i.e des campagnes en cours de diffusion au jour j sur la fenêtre d'historique choisie. Pour palier ce problème, nous mettons en place une matrice des quantiles hybride composée des quantiles de prédiction issus du modèle (F1) et de ceux issus du modèle (E) en cas de paramètres non estimables avec (F1) pour certains clusters. On considère que les campagnes c_1 (issue de l'exemple de la table 4.2) et c_3 appartiennent au cluster 5 tandis que les autres campagnes appartiennent aux 4 autres clusters. On obtient alors la matrice des quantiles de prédiction suivante :

$$
\begin{array}{c}
 \\
\iota_1 \\
\iota_2 \\
\vdots \\
\iota_L
\end{array}
\begin{array}{c}
\begin{array}{ccccc}
c_1 \in C_5 & c_2 \in C_1 & c_3 \in C_5 & \dots & c_c \in C_2
\end{array} \\
\begin{pmatrix}
\hat{q}_{11} & \hat{q}_{21} & \hat{q}_{31} & \dots & \hat{q}_{c1} \\
\hat{q}_{12} & \hat{q}_{22} & \hat{q}_{32} & \dots & \hat{q}_{c2} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\hat{q}_{1L} & \hat{q}_{2L} & \hat{q}_{3L} & \dots & \hat{q}_{cL}
\end{pmatrix}
\end{array}
$$

avec C le nombre total de campagnes du jeu d'apprentissage et L le nombre de contextes possibles ι à prédire pour chaque campagne. Les quantiles prédictifs des campagnes c_1 et c_3 (ainsi que de toutes les campagnes appartenant au cluster 5) sont calculés via le modèle (E) représenté en orange dans la matrice ci-dessus (et présenté dans la section 3.1.3.2). Tous les autres vecteurs de quantiles de chaque campagne appartenant aux clusters 1, 2, 3 ou 4 sont quant à eux calculés normalement avec le modèle (F1).

À travers cet exemple illustratif, nous avons pu présenter un cas de matrice de quantiles hybride qui contient à la fois des prédictions issues du modèle (F1) mais aussi du modèle (E). Cela permet d'avoir à disposition de manière quotidienne une estimation des quantiles prédictifs pour chacune des campagnes et des contextes possibles.

4.2.4 Première expérimentation en production

Dans cette première expérimentation, le modèle est testé en temps réel sur la plateforme d'enchère. Afin de ne pas bouleverser tout le processus d'enchère et analyser la qualité prédictive du modèle, plusieurs itérations sont prévues et dès lors qu'une itération sera mise en production et validée en terme de performance, l'itération suivante pourra démarrer.

Cette mise en production se base un principe d'A/B testing. Cette méthodologie permet de mettre en concurrence deux versions d'une fonctionnalité (souvent l'actuelle et la nouvelle) afin de mesurer leur efficacité selon un KPI donné. Ici, nous procédons de manière analogue puisque nous voulons comparer le taux de clics moyen observé selon que l'on utilise le système d'enchère actuel (sans prise en compte de la prédiction de clics) ou le nouveau système (prenant en compte la prédiction de clics). Ainsi, le modèle prédictif est utilisé sur un certain pourcentage de requêtes entrantes (de manière aléatoire) pour définir la campagne publicitaire qui va être proposée à l'enchère (décrit dans la section 4.2.2). Le reste du temps, c'est le fonctionnement du choix de la campagne sans modèle prédictif qui est utilisé (décrit dans la section 4.2.1). De fait, nous pouvons comparer à posteriori le CTR obtenu sur une durée de test définie lorsque le modèle est utilisé et lorsqu'il ne l'est pas. L'objectif est d'analyser si le CTR a significativement augmenté lorsque le modèle de prédiction est utilisé.

Pour rappel, parmi toutes les requêtes arrivant sur la plateforme d'enchère, seulement

une partie d'entre elles sont compatibles avec le modèle. Par exemple, nous ne traitons pas dans ces travaux les formats publicitaires de type vidéo ou audio. Pour autant, ces types de formats publicitaires sont présents dans certaines requêtes qui nous parviennent. Dans la suite, nous nous concentrons exclusivement sur le sous-ensemble de requêtes qui nous intéressent afin d'avoir deux populations comparables à tester. Lors de cette première itération en production, nous avons décidé de tester et d'utiliser le modèle sur un pourcentage restreint du trafic entrant (et compatible). Ainsi, le modèle de prédiction du CTR est utilisé pour 16% des requêtes compatibles.
Au cours de ces premières expérimentations en production, l'impact du budget et celui du modèle de prédiction du CTR sont pondérés de façon à compter de manière équivalente dans le choix de la campagne à diffuser :

$$(p_i)_{i=1,\dots,C'} = 0.5 \times \frac{(b_i)_{i=1,\dots,C'}}{\sum_{j=1}^{C'} b_j} + 0.5 \times \frac{(q_i)_{i=1,\dots,C'}}{\sum_{j=1}^{C'} q_j} \tag{4.1}$$

avec (p_i) le poids associé à la campagne publicitaire i, b_i le poids relatif au budget variant entre $[0; 50]$ et q_i le quantile de la prédiction entre $[0; 1]$ pour le contexte de l'enchère. La campagne proposée à l'enchère est choisie de manière aléatoire selon la probabilité calculée via le vecteur de poids.

Jour de la semaine	Modèle	Impressions	Clicks	**CTR**	IC_inf	IC_sup
Lundi	No	2753811	22911	**0.8320**	0.8212	0.8427
	Yes	520125	4227	**0.8127**	0.7883	0.8371
Mardi	No	2825820	26280	**0.9300**	0.9188	0.9412
	Yes	532114	5251	**0.9868**	0.9603	1.0134
Mercredi	No	3022994	29033	**0.9604**	0.9494	0.9714
	Yes	575033	5939	**1.0328**	1.0067	1.0589
Jeudi	No	2015751	22808	**1.1315**	1.1169	1.1461
	Yes	383703	4411	**1.1496**	1.1159	1.1833
Vendredi	No	2558380	38273	**1.4960**	1.4811	1.5109
	Yes	486553	7311	**1.5026**	1.4684	1.5368
Samedi	No	1825740	29915	**1.6385**	1.6201	1.6569
	Yes	349183	5647	**1.6172**	1.5754	1.6590
Dimanche	No	1405435	24486	**1.7422**	1.7206	1.7639
	Yes	265711	4818	**1.8132**	1.7625	1.8640

TABLE 4.3 – Résultats obtenus sur 15 jours de test en production : valeur du CTR obtenue par jour de la semaine avec (*Modele=Yes*) et sans (*Modele=No*) utilisation du modèle (proportion d'enchères utilisant le modèle : 16%).

OS	Modèle	Impressions	Clicks	**CTR**	IC_inf	IC_sup
Android	No	11619521	143573	**1.2356**	1.2293	1.2420
	Yes	2207281	27813	**1.2601**	1.2453	1.2748
iOS	No	4742143	50249	**1.0596**	1.0504	1.0688
	Yes	896481	9835	**1.0971**	1.0755	1.1186
Other	No	62113	106	**0.1707**	0.1382	0.2031
	Yes	11760	7	**0.0595**	0.0154	0.1036

TABLE 4.4 – Résultats obtenus sur 15 jours de test en production : valeur du CTR obtenue par système d'exploitation avec (*Modele=Yes*) et sans (*Modele=No*) utilisation du modèle (proportion d'enchères utilisant le modèle : 16%).

Les tables 4.3 à 4.8 récapitulent les résultats obtenus en production sur une période de 13 jours (3 juillet - 16 juillet 2020). Les tables sont à lire et analyser de la manière suivante :

- Chaque table correspond à une variable explicative découpée selon ses modalités. Ainsi nous observons des valeurs de CTR moyen pour chaque modalité.
- La colonne **Modèle** correspond à la valeur moyenne de CTR calculée avec le modèle prédictif (Yes) ou sans (No). Il est important de noter que toutes les analyses qui vont suivre sont effectuées dans un cadre où les valeurs de CTR sont comparables. Cela signifie que pour les cas où le modèle n'est pas utilisé (Modèle=No), seuls les cas où il y a au moins deux campagnes en compétition sont analysés.
- Pour chaque valeur de CTR obtenue, un intervalle de confiance est calculé de la manière suivante :

$$IC_{Y/n} = \left[\frac{Y}{n} - 1.96 \times \sqrt{\frac{\frac{Y}{n}(1-\frac{Y}{n})}{n}} \; ; \; \frac{Y}{n} + 1.96 \times \sqrt{\frac{\frac{Y}{n}(1-\frac{Y}{n})}{n}} \right] \tag{4.2}$$

  avec n le nombre d'impressions et $\frac{Y}{n}$ le CTR observé.
- Lorsque la modalité issue d'une variable explicative est colorée en vert dans les tables, cela signifie que l'utilisation du modèle prédictif induit une amélioration significative du CTR moyen observé. En d'autre termes, les intervalles de confiance des cas où le modèle est utilisé et des cas où le modèle ne l'est pas, sont distincts. Cela rend la différence de valeur de CTR significative.
- De manière analogue, lorsque la coloration d'une modalité est rouge dans les tables 4.3 à 4.8, cela signifie qu'il y a une différence significative de la valeur du CTR moyen entre l'utilisation ou pas du modèle. Mais que cet écart n'est pas à l'avantage du modèle.

Plage horaire	Modèle	Impressions	Clicks	**CTR**	IC_inf	IC_sup
$[0h; 4h[$	No	748488	9448	**1.2623**	1.2370	1.2876
	Yes	143950	1826	**1.2685**	1.2107	1.3263
$[4h; 8h[$	No	2044027	35415	**1.7326**	1.7147	1.7505
	Yes	388623	6890	**1.7729**	1.7314	1.8144
$[8h; 12h[$	No	3523370	47060	**1.3357**	1.3237	1.3476
	Yes	668656	9302	**1.3911**	1.3631	1.4192
$[12h; 16h[$	No	3029435	40343	**1.3317**	1.3188	1.3446
	Yes	576436	7993	**1.3866**	1.3564	1.4168
$[16h; 20h[$	No	2749886	29753	**1.0820**	1.0697	1.0942
	Yes	522167	5654	**1.0828**	1.0547	1.1109
$[20h; 24h[$	No	4313051	31691	**0.7348**	0.7267	0.7428
	Yes	812646	5939	**0.7308**	0.7123	0.7493

TABLE 4.5 – Résultats obtenus sur 15 jours de test en production : valeur du CTR obtenue par plage horaire avec (*Modele=Yes*) et sans (*Modele=No*) utilisation du modèle (proportion d'enchères utilisant le modèle : 16%).

Application/Site	Modèle	Impressions	Clicks	**CTR**	IC_inf	IC_sup
Application	No	12682884	156206	**1.2316**	1.2256	1.2377
	Yes	2414029	30253	**1.2532**	1.2392	1.2672
Site	No	4166066	42768	**1.0266**	1.0169	1.0363
	Yes	782872	8303	**1.0606**	1.0379	1.0833

TABLE 4.6 – Résultats obtenus sur 15 jours de test en production : valeur du CTR obtenue par type de support (application ou site web) avec (*Modele=Yes*) et sans (*Modele=No*) utilisation du modèle (proportion d'enchères utilisant le modèle : 16%).

Type	Modèle	Impressions	Clicks	**CTR**	IC_inf	IC_sup
Type 1	No	7840108	89871	**1.1463**	1.1388	1.1537
	Yes	1477438	17101	**1.1575**	1.1402	1.1747
Type 2	No	3136111	41406	**1.3203**	1.3077	1.3329
	Yes	602213	8330	**1.3832**	1.3537	1.4127
Type 3	No	371819	14414	**3.8766**	3.8146	3.9387
	Yes	71381	2698	**3.7797**	3.6398	3.9196
Type 4	No	5076234	48245	**0.9504**	0.9420	0.9588
	Yes	964598	9527	**0.9877**	0.9679	1.0074

TABLE 4.7 – Résultats obtenus sur 15 jours de test en production : valeur du CTR obtenue par type de publicité avec (*Modele=Yes*) et sans (*Modele=No*) utilisation du modèle (proportion d'enchères utilisant le modèle : 16%).

Format	Modèle	Impressions	Clicks	**CTR**	IC_inf	IC_sup
Pavé	No	6415907	23803	**0.3710**	0.3663	0.3757
	Yes	1213694	4471	**0.3684**	0.3576	0.3792
Interstitiel	No	6331110	154090	**2.4339**	2.4219	2.4459
	Yes	1202461	30056	**2.4995**	2.4716	2.5274
Bannière	No	3677936	16059	**0.4366**	0.4299	0.4434
	Yes	699636	3129	**0.4472**	0.4316	0.4629

TABLE 4.8 – Résultats obtenus sur 15 jours de test en production : valeur du CTR obtenue par format publicitaire avec (*Modele=Yes*) et sans (*Modele=No*) utilisation du modèle (proportion d'enchères utilisant le modèle : 16%).

L'analyse de ces tables nous permet d'aboutir à de premières conclusions. Plusieurs perspectives, qui découlent directement de cette première expérimentation menées sur 13 jours, sont détaillées en section 4.4. Globalement, lorsqu'il y a une différence significative de la valeur du CTR pour une modalité donnée (d'une variable explicative), il s'agit d'une amélioration du taux de clics avec l'utilisation du modèle. Par exemple, en analysant la table 4.6, nous remarquons que pour la modalité *application* le CTR est passé de 1.2316 à 1.2532 avec des intervalles de confiance qui ne se croisent pas. Il en est de même de la modalité *site* où le CTR moyen calculé augmente d'environ 3%.
Cette toute première expérimentation concerne uniquement les analyses du CTR moyen au sein d'une variable donnée. Elle n'en demeure pas moins encourageante. Elle a par exemple permis de constater que lorsque la valeur du CTR moyen obtenue sur la proportion de requêtes utilisant le modèle de prédiction est significativement supérieure au CTR moyen obtenu sur les autres requêtes, cela correspond à des contextes où le modèle a été beaucoup utilisé. Cette observation confirme notre souhait d'augmenter la proportion de requêtes entrantes utilisant les quantiles de prédiction serait bénéfique à la plateforme dans les prochaines itérations de la mise en production.

4.3 Développement du modèle prédictif pour le moteur d'enchère

4.3.1 Architecture du projet

L'industrialisation du modèle de prédiction de clics a nécessité un découpage en plusieurs étapes qui sont résumées dans la figure 4.3.

4.3.1.1 Création de l'historique

La création de l'historique des données utilisées est décrite en bleu dans la figure 4.3. Les données sont directement récupérées de la base de données Big Query.

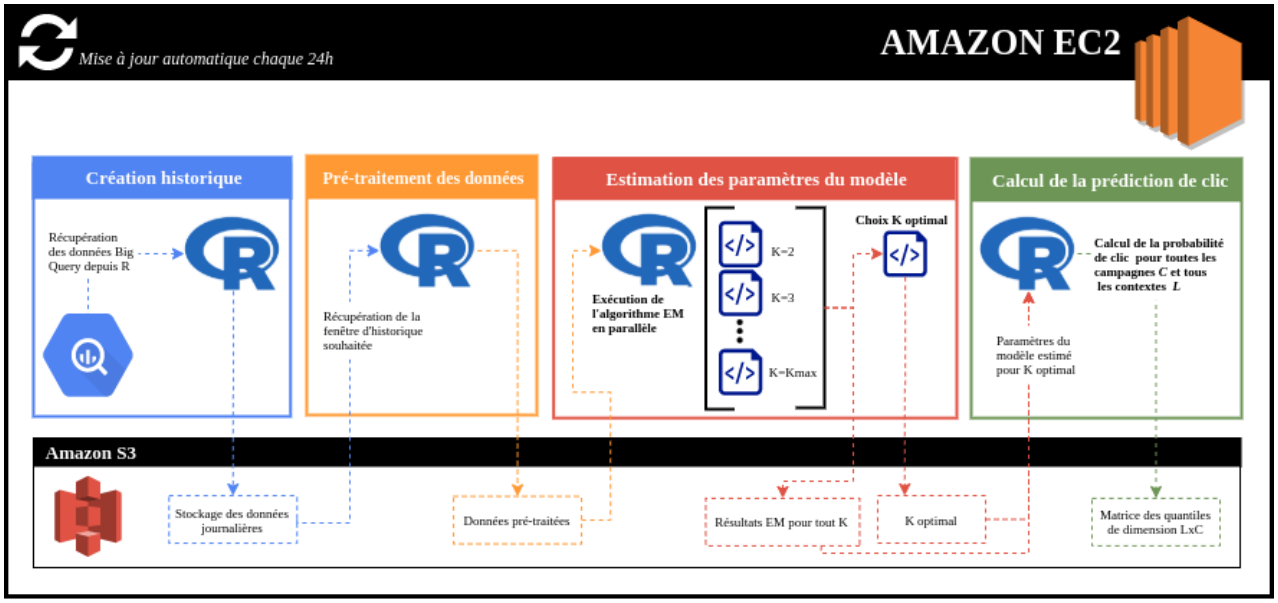


FIGURE 4.3 – Architecture du projet de prédiction de clics : les chapitres 2 et 3 de ce manuscrit correspondent respectivement aux étapes *Estimation des paramètres du modèle* et *Calcul de la prédiction de clics*. Le déploiement en production a permis de lier la création de l'historique, le prétraitement des données, l'estimation des paramètres, la prédiction du clic et l'utilisation de la prédiction en temps réel dans le système d'enchère.

Big Query en quelques mots : les données utilisées pour le processus de prédiction de clics sont stockées sur Big Query, outil proposé par Google Cloud Platform (GCP). Big Query est un entrepôt de données (*data warehouse* en anglais). Il s'agit d'une base de données sur laquelle on a la possibilité de centraliser nos différentes sources à travers des processus nommés Extract Load Transform (ELT). Un entrepôt de données est souvent couplé avec un lac de données (*data lake* en anglais). Dans un lac de données, on charge des données brutes sur un système de fichiers difficilement requêtable tandis qu'avec l'entrepôt de données, on charge la donnée pour la rendre directement exploitable pour des analyses SQL à travers des vues spécifiques appelées "magasins de données". En tant qu'entrepôt de données, BigQuery contient principalement :

1) Des données de type "événements" (stockées en brut sur Amazon S3) : il s'agit de données qui recensent l'ensemble de tous les événements obtenus sur une publicité : impressions, clics, ou tout autre type de conversion/d'action.

2) Des données de type "métier" (stockées sur une base de données) : il s'agit des méta-données propres à chaque campagne créée. On y retrouve la date de création, la dernière modification, le format de la publicité, le ciblage souhaité par l'annonceur et tout ce qui est relatif aux caractéristiques de la campagne.

Ces **deux types de données**, issues de deux lacs de données distincts, sont **réconciliées** via une jointure SQL et disponible sur ce que l'on appelle une vue dans BigQuery.

Ce sont ces données qui sont utilisées pour créer l'historique de données. La fenêtre d'his-

torique choisie est de 28 jours (nous prenons ici la durée standard, mais le nombre de jours peut évoluer). Considérons que le modèle prédictif est exécuté à une date t donnée. Il est nécessaire de requêter les $t-28$ derniers jours d'historique sur Big Query pour construire le jeu de données. Cette requête est effectuée de manière indépendante pour chaque jour nécessaire dans l'historique. Une fois les données récupérées, elles sont directement stockées sur Amazon S3 avec autant de fichiers stockés que de jours d'historique souhaités. Ainsi, lorsque le modèle prédictif est exécuté à la date $t+1$, seul le jour t d'historique est considéré comme manquant sur S3 et doit être requêté. En d'autres termes, ce stockage par jour présente l'avantage d'optimiser le nombre de requêtes à faire quotidiennement lorsque l'algorithme prédictif est exécuté.

Amazon en quelques mots : chez TabMo, Amazon S3 est le principal lac de données. Il s'agit d'un service payant de stockage proposé par Amazon Web Services. Nos applications métiers y envoient leurs messages. Ce service permet de stocker n'importe quelle quantité de données tout en assurant une très haute disponibilité de celles-ci tout en limitant les coûts de stockage (0,023 USD par Go).

4.3.1.2 Prétraitement des données

Le prétraitement des données concerne la partie jaune de la figure 4.3. Un script R se charge en premier lieu de récupérer les différents fichiers de données journaliers stockés sur S3. Le nombre de fichiers à récupérer est dépendant de l'historique souhaité. De manière standard, il s'agirait de récupérer 28 fichiers correspondant aux 28 jours d'historique. À ce stade, nous avons donc T jeux de données distincts correspondant à l'historique choisi. Les données sont donc concaténées en une seule table. C'est sur cette table jointe contenant tout l'historique brut que des prétraitements sont appliqués sur les données. Le détail de ces derniers est décrit dans la section 2.1.1 du chapitre 2. La dernière étape de cette partie consiste à agréger les données puis à sauvegarder dans S3 ce jeu de données prétraité, prêt à être utilisé pour l'apprentissage du modèle. Cette étape de prétraitements est présentée

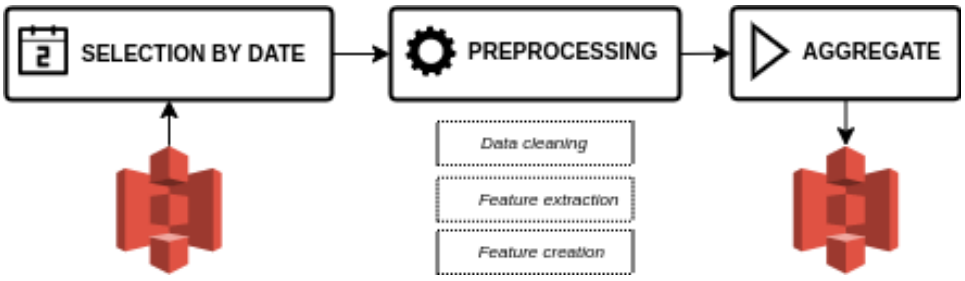


FIGURE 4.4 – Etape de prétraitement des données : les données sont extraites de S3 pour être prétraitées puis elles sont ensuite stockées sous ce nouveau format dans Amazon S3.

et schématisée sur la figure 4.4 avec ses 3 étapes principales : la sélection de l'historique,

le nettoyage de données couplé à la création de variables explicatives puis l'agrégation des données par contextes observés.

4.3.1.3 Estimation des paramètres du modèle

Toute la partie du processus représenté en rouge dans la figure 4.3 concerne l'estimation des paramètres du modèle de mélange, étape préliminaire indispensable pour la prédiction. Les détails de l'estimation des paramètres de ce modèle sont disponibles dans le chapitre 2. La chronologie de cette étape se découpe en plusieurs phases :

1) Les données prétraitées et stockées dans S3 sont récupérées et utilisées dans l'estimation des paramètres du modèle de mélange.

2) Le modèle de mélange est exécuté pour plusieurs valeurs de cluster K différentes afin de pouvoir ensuite choisir le nombre de clusters optimal K_{opt} pour ce clustering. L'exécution de l'algorithme EM pour les différentes valeurs de K s'effectue en parallèle afin d'optimiser le temps de calcul et les capacités de l'instance Amazon sur laquelle les calculs sont exécutés. L'estimation des paramètres du mélange se base sur l'algorithme EM développé dans le package *binomialMix* (décrit dans l'annexe (A)). Pour chaque exécution de l'algorithme EM avec un nombre de clusters K fixé, les résultats associés sont sauvegardés sur S3.

3) La dernière étape consiste à comparer les différents résultats obtenus pour différents nombres de clusters. La comparaison se fait au sens de critères statistiques tels que le BIC ou l'ICL et permet de choisir un nombre de clusters optimal K_{opt} pour le mélange.

4.3.1.4 Calcul de la prédiction de clics

Cette dernière étape correspond à la partie verte de la figure 4.3. À l'issue de l'étape précédente, les paramètres du mélange ont été estimés et stockés dans S3 pour un nombre de clusters K_{opt}. Ainsi, à partir des paramètres estimés, la matrice des quantiles de prédiction est calculée (voir section 4.2.2 pour plus de détails sur cette matrice des quantiles). Il s'agit alors de calculer les probabilités de clic pour l'ensemble des campagnes présentes dans le jeu de données et pour tous les contextes possibles. Cette matrice est ainsi précalculée à chaque nouvelle exécution du pipeline de prédiction de clics. Cela permet au moteur d'enchère d'être plus performant au moment où une requête arrive. Le système peut alors lire la matrice des quantiles et procéder au choix de la campagne publicitaire la plus pertinente à sélectionner. Le modèle prédictif n'a pas besoin d'être sollicité pour chaque requête entrante.

4.3.2 Processus d'industrialisation du développement

L'architecture de ce projet, présentée dans la section 4.3.1, repose sur un certain nombre de *bonnes pratiques* et concepts de développement. Tout le processus de mise en production du modèle de prédiction se base sur l'utilisation de la pratique devOps via l'outil Git. Ce dernier est un système de contrôle de source et de version. Il se compose d'un ensemble d'outils permettant le versionnage d'un projet et facilite le travail collaboratif. Ses principales caractéristiques sont les suivantes :

- Git permet de suivre l'évolution d'un code source et possède la capacité de revenir en arrière en cas de problème grâce au suivi de modifications des fichiers.
- Git permet de travailler à plusieurs et à distance sans risque sur un même projet, voire un même fichier. Si deux collaborateurs modifient en même temps le même fichier, Git permet une fusion de leurs modifications respectives et cela sans perte d'information.
- Il permet l'application d'un modèle de workflow de travail efficace. Les branches sont l'une des plus puissantes fonctionnalités proposées par Git. Il est possible de créer plusieurs branches totalement indépendantes au sein d'un projet. Chaque branche sert alors à développer une (ou des) fonctionnalité(s) tout en maintenant chacune d'elles isolée des autres le temps du développement. La branche par défaut s'appelle la branche master [1] et c'est sur celle-ci que toutes les branches créées doivent fusionner. Lorsqu'on effectue des modifications dans son dossier de travail local, on peut pousser les changements dans le dépôt distant. Il est également très facile d'accéder aux branches de quelqu'un d'autre ou de mettre sa copie de travail locale à jour grâce à des commandes git spécifiques.
- **L'utilisation de Git via Gitlab** : Gitlab est la plateforme de gestion de projet collaboratif utilisée chez TabMo pour gérer les dépôts Git. Il s'agit d'un outil permettant la gestion de tout le processus de développement. Il assure aussi une collaboration simplifiée entre les différents contributeurs d'un même projet. Une fois qu'un projet est poussé ("PUSH") sur Gitlab, il est alors possible de le partager à d'autres utilisateurs. Pour chaque nouvelle fonctionnalité développée sur une branche, l'objectif est de proposer aux autres collaborateurs une *revue de code* avant de fusionner et déployer ladite branche sur la branche principale.

Le processus de développement se divise en plusieurs étapes décrites sur la Figure 4.5 :

Planification : cette partie concerne le séquençage des actions ainsi que la mise en place de la collaboration entre les différents acteurs du projet. La planification du projet s'effectue selon la méthodologie Agile qui a pour objectif de découper le projet en petites tâches à réaliser. Ce découpage se fait de manière collégiale et itérative tout

1. Cependant, de plus en plus souvent la branche principale est appelée "main" (pour éviter la référence à l'esclavagisme).

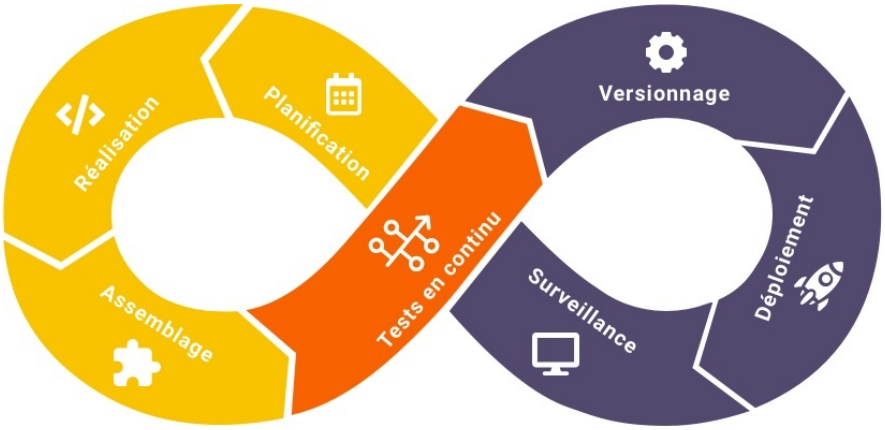


FIGURE 4.5 – Processus d'industrialisation du développement en 7 étapes selon la pratique *devOps*

au long du développement. Pour cela, les outils utilisés sont principalement Gitlab et Jira. Ce dernier est un outil de gestion de projets, où chaque tâche est décrite et assignée, permettant un suivi et une collaboration à travers toutes les équipes de l'entreprise. L'organisation du travail s'articule aussi à travers des rituels (meeting journalier).

Réalisation et Assemblage : une fois les tâches définies, l'objectif est de développer les fonctionnalités attendues dans le temps imparti (sprint), et les intégrer dans la solution logicielle.

Tests en continu : à chaque modification du code fonctionnel, des tests automatisés sont exécutés sur un environnement d'intégration continue (comme Jenkins, Travis ou encore GitlabCI,...) pour s'assurer de la qualité du projet et limiter le nombre de régressions lors des prochains déploiements.

Versionnage : chaque élément modifié dans le code, ainsi que chaque nouvelle livraison, sont versionnés. Ceci est important à des fins de traçabilité, de collaboration, et de retour arrière si un problème est détecté.

Déploiement :

— Cette partie s'effectue sur la plateforme Cloud Amazon Web Services, via le service EC2. Il s'agit d'un service proposé par Amazon permettant la location de serveurs sur lesquels les applications sont exécutées. Amazon est leader dans ce que l'on appelle le *Cloud* et possède des millions de serveurs informatiques répartis sur différents sites dans le monde. L'intérêt du Cloud par rapport à des serveurs gérés par soi-même repose sur le dynamisme de l'infrastructure et sa capacité à faire payer l'utilisateur à la minute pour les ressources qu'il utilise au lieu d'un engagement sur plusieurs mois. EC2 possède plusieurs fonctionnalités très utiles pour le déploiement et le monitoring des applications en production.

— Afin d'automatiser le déploiement de nos services, on utilise un service d'in-

tégration continue (par exemple : Jenkins, GitlabCI,...) qui a pour mission de compiler le projet, d'exécuter les tests, et en cas de succès de le déployer sur notre infrastructure. Pour automatiser la création de machines sur nos différentes infrastructures (on parle d'environnement de développement ou de production) et réduire au maximum les tâches manuelles, l'outil Terraform est utilisé. Cette pratique se nomme l'Infrastructure As Code (IAC) et permet de décrire nos types d'instances souhaités ainsi que toutes les configurations réseaux.

Surveillance : une fois le projet déployé et mis à disposition des utilisateurs il est crucial de contrôler et surveiller l'utilisation de ses ressources : CPU, mémoire, disques, logs applicatifs. Il est également important de mettre en place des alertes automatisées en cas d'anomalie pour être le plus réactif possible dans leur résolution.

— Datadog est un service de monitoring d'application à une échelle Cloud. Il peut ainsi surveiller des bases de données ou des applications en affichant les traces (logs) et les métriques (metrics) de celles-ci. Datadog répond au problème de la distribution des applications où les logs sont éparpillés partout sur des instances allouées à la demande. Avec Datadog, tous les logs sont ainsi envoyés dans un endroit centralisé.

— En alternative à une solution propriétaire et payante, on peut utiliser Grafana. Il s'agit d'un outil de métrologie permettant la visualisation de données et génère des graphiques, tableaux de bord à partir de bases de données de séries temporelles. Ce service est principalement utilisé pour surveiller et alerter sur des métriques que l'on souhaite monitorer. Par exemple, il est possible de programmer un seuil à partir duquel une certaine métrique n'est plus dans la norme et ainsi envoyer une notification à l'utilisateur via des alertes mails.

Ainsi, tout le processus d'industrialisation se base sur ces bonnes pratiques de développement informatique où toutes les étapes décrites ci-dessus sont essentielles pour assurer la qualité des prédictions et résultats obtenus.
Cette philosophie de travail s'applique également au du package R *binomialMix* développé au cours de cette thèse. Lorsque nous souhaitons mettre à jour le package (qu'il s'agisse d'une correction de bug ou d'ajout d'une nouvelle fonctionnalité), les différentes étapes de la méthodologie de développement sont menées à bien. La figure 4.6 représente les étapes de réalisation, tests en continu, versionnage et déploiement. Cela signifie que, pour chaque modification de code, le package est recompilé sur un environnement dédié et une série de tests unitaires doit être (re)passée en revue (voir figure 4.7) Ce processus, un peu lourd de prime abord, permet d'éviter des régressions ou des bugs dans le package après l'ajout ou la modification d'une ligne de code.

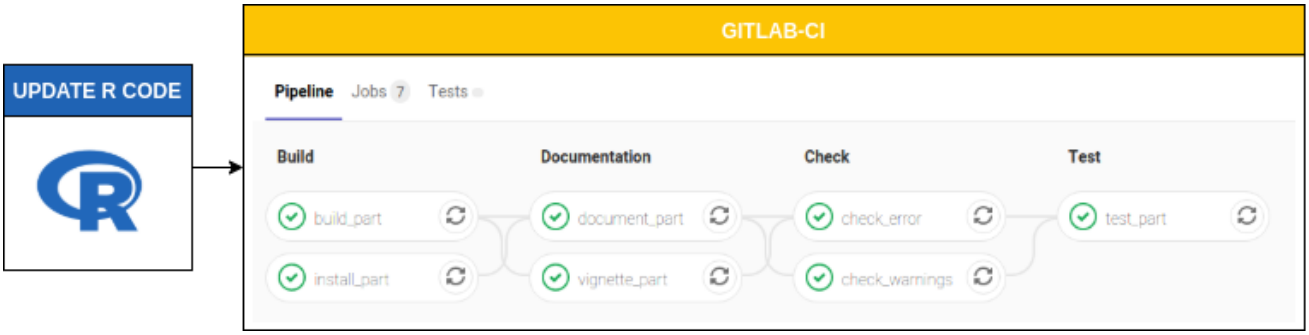


FIGURE 4.6 – Processus de développement du package R construit sur le principe d'intégration continue : pour chaque mise à jour du code, le package doit être en mesure de repasser toutes les étapes de déploiement, documentation et tests.

```
Testing binomialMix
✔ |  OK F W S | Context
✔ |   2       | extract_id [0.1 s]
✔ |   3       | extract_target
✔ |   3       | extract_variables [0.1 s]
✔ |   6       | init_design_matrices [0.1 s]
✔ |   2       | init_lambda
✔ |   5       | init_tau
✔ |   5       | log_density_binom
══ Results ═══════════════════════
Duration: 0.5 s
OK:       26
Failed:   0
Warnings: 0
Skipped:  0
```

FIGURE 4.7 – Affichage des résultats à l'issue des tests unitaires : tous les tests doivent être au niveau du statut *OK* pour que le package puisse compiler.

4.4 Perspectives

La mise en production du modèle prédictif a permis d'aboutir à de premières expérimentations en temps réel. Pour cela, nous avons utilisé le modèle prédictif sur une petite partie du trafic entrant (16%) et laissé le reste des requêtes dans leur comportement classique. Cette méthode nous a permis à posteriori d'analyser si, pour les requêtes utilisant le modèle prédictif, le CTR moyen observé avait augmenté. Après quelques analyses, cette première itération a fourni des premiers résultats prometteurs. Pour certains contextes observés (i.e pour certaines modalités des variables explicatives du modèle) le CTR moyen des requêtes utilisant le pipeline de prédiction a significativement augmenté par rapport à celles passant via le processus classique. Plusieurs axes d'amélioration nous permettraient cependant d'aboutir à des résultats encore meilleurs :

Augmenter la proportion de requêtes utilisant le modèle : la proportion de requêtes entrantes qui utilisent le modèle prédictif se situe aux alentours de 16% de l'ensemble du trafic entrant. L'idée était de commencer les tests sur une petite proportion des requêtes afin de ne pas trop perturber la diffusion des campagnes publicitaires. Après plusieurs semaines de test, nous avons constaté que le modèle prédictif ne dégradait en rien le processus de diffusion des campagnes. Au vue des

premiers résultats obtenus dans la section 4.2.4, il semble pertinent d'augmenter la proportion de requêtes entrantes utilisant le modèle prédictif pour de prochaines expérimentations.

Donner plus de poids à la prédiction : jusqu'à présent, le choix final de la campagne publicitaire se base sur une répartition équitable entre le poids associé au budget et celui associé au quantile de prédiction obtenu. Il serait intéressant de faire varier ces pourcentages associés à chacun des deux critères pour voir l'impact que cela peut avoir sur la performance prédictive.

Automatiser l'analyse des résultats en production : il serait intéressant de réfléchir à une méthodologie pour analyser de manière automatique et itérative les résultats obtenus en production. Des outils de surveillance avec des graphiques en temps réel permettraient d'avoir un aperçu de l'évolution du taux de clics et cela dans les deux groupes de test distincts : celui qui regroupe les requêtes qui utilisent le modèle prédictif et celui regroupant toutes les autres.

Chapitre 5

Conclusion

Ces travaux de thèse ont eu pour objectif l'amélioration du taux de clics pour les annonceurs qui diffusent leurs campagnes publicitaires sur la plateforme de TabMo. Pour répondre à cette problématique, plusieurs étapes ont été nécessaires :

- La caractérisation des données métiers a été le premier travail. Ce travail exploratoire nous a permis de mettre en exergue les principales caractéristiques des données (volumineuses, hétérogènes, complexes et clairsemées). Nous avons ainsi construit un jeu de données basé sur l'observation du taux de clics au cours du temps et pour chaque campagne en cours de diffusion sur une période donnée.
- À partir de ces données, l'objectif préliminaire était de classer les publicités en plusieurs groupes distincts. Pour mettre en œuvre cette classification non supervisée, nous avons utilisé un modèle de mélange pour données longitudinales et de distribution binomiale (puisque la variable d'intérêt est un taux de clics). Le package R *binomialMix* a été développé dans cet objectif de clustering.
- Cette étape de classification nous a servi de point d'entrée dans la mise en place d'un modèle prédictif. L'objectif du modèle de prédiction est de fournir, pour chaque requête entrante qui transite sur la plateforme, une probabilité de clic pour chaque publicité présente dans notre inventaire. Plusieurs itérations ont été faites sur le modèle de prédiction afin de comparer différentes approches. Deux modèles se sont clairement démarqués en terme de qualité de prédiction au cours de nos expérimentations. Le premier modèle se base sur l'affectation de chaque campagne à un cluster. Il estime au sein de chaque groupe de campagne des effets fixes liés aux variables explicatives du modèle (caractéristiques de la campagne, du contexte de l'encart publicitaire,...) ainsi qu'un effet aléatoire propre à chaque campagne qui permet de traduire une dépendance entre les observations d'une même campagne. Le second modèle se base sur une contribution pondérée des campagnes au sein des différents clusters. Il n'y a pas d'affectation à proprement parler sur un unique cluster. Le calcul de la prédiction pour chaque campagne se base ainsi sur le vecteur π_c d'appartenance de chaque campagne c aux différents clusters. Pour la suite, nous avons fait le choix de nous concentrer sur le modèle offrant les meilleurs résultats en terme de prédiction (modèle avec affectation à un cluster et effet aléatoire associé à la campagne).
- La dernière étape de ce travail a permis la mise en production de l'ensemble du processus : récupération et prétraitements des données, construction de groupes de

campagnes à partir d'un mélange de distribution binomiale et développement d'un modèle prédictif basé sur les groupes obtenus. Cette phase a été une étape d'industrialisation du modèle prédictif, afin de l'intégrer directement à la plateforme d'enchère en temps réel. Cette dernière brique nous a permis d'estimer une probabilité de clic pour toutes les campagnes présentes dans l'inventaire à un instant t dès lors qu'un emplacement publicitaire est disponible et nous parvient.

Pour chacune des étapes de ce travail, plusieurs perspectives se sont naturellement dessinées. Le prétraitement des données et le choix des variables explicatives du modèle sont un levier évident pour la suite de ces travaux. De nouvelles variables pourraient être considérées au regard des informations disponibles dans les requêtes. De la même manière, des regroupements de modalités ont été faits sur certaines variables explicatives mais ces choix pourraient être réévalués à posteriori, maintenant que les résultats en production sont disponibles. S'agissant de la classification non supervisée, nous avons développé un algorithme de type EM dans le cadre d'un mélange de distributions binomiales. Comme décrit en détails dans la section 2.4, l'initialisation de l'EM a été l'objet de beaucoup de travaux dans la littérature. Dans le cadre de cette thèse, l'objectif principal était prédictif. De fait, nous n'avons pas porté une attention particulière sur cette étape de l'algorithme mais cela fait partie des perspectives d'amélioration du modèle de mélange. Les évolutions possibles des performances du modèle prédictif sont détaillées dans la section 3.3. Elles s'articulent principalement autour de la fenêtre d'historique optimale du jeu d'apprentissage afin de le rendre adaptatif selon les périodes de l'année et autour du choix de modèle prédictif propre à chaque campagne. Pour l'instant, nous avons principalement analysé l'ensemble des résultats de prédiction à partir d'un modèle par affectation à un cluster et effets aléatoires liés à chaque campagne de l'apprentissage. La mise en production du modèle a montré quelques cas limites à l'utilisation exclusive de ce modèle et a permis le développement d'un modèle "back-up" se basant sur une contribution pondérée de chaque cluster (plus de détails sur les modèles en section 3.1.3). Cela nous a ainsi permis d'ouvrir une perspective à propos de l'utilisation d'un modèle plutôt qu'un autre selon les caractéristiques de la campagne (depuis quand diffuse-t-elle ? par exemple). Enfin, et il s'agit ici des perspectives qui seront réalisés à court terme, plusieurs itérations sont possibles concernant le déploiement du modèle prédictif en production. Pour l'instant, seule une petite partie des requêtes entrantes est affectée par les quantiles prédictifs, l'objectif serait d'étendre l'utilisation du modèle à la totalité du trafic entrant. Jusqu'à présent, les quantiles de prédictions propres à chaque campagne publicitaire sont considérés de manière équitable avec d'autres paramètres tels que le budget dans le choix de la campagne la plus pertinente à diffuser. Un autre perspective serait de faire varier ces poids pour analyser l'impact de l'un ou l'autre des paramètres.

Bibliographie

Baudry, J.-P. and Celeux, G. (2015). Em for mixtures. *Statistics and computing*, 25(4) :713–726.

Biernacki, C., Celeux, G., and Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE transactions on pattern analysis and machine intelligence*, 22(7) :719–725.

Biernacki, C., Celeux, G., and Govaert, G. (2003). Choosing starting values for the em algorithm for getting the highest likelihood in multivariate gaussian mixture models. *Computational Statistics & Data Analysis*, 41(3-4) :561–575.

Celeux, G., Chauveau, D., and Diebolt, J. (1995). On stochastic versions of the em algorithm.

Celeux, G. and Govaert, G. (1992). A classification em algorithm for clustering and two stochastic versions. *Computational statistics & Data analysis*, 14(3) :315–332.

Chan, P. P., Hu, X., Zhao, L., Yeung, D. S., Liu, D., and Xiao, L. (2018). Convolutional neural networks based click-through rate prediction with multiple feature sequences. In *IJCAI*, pages 2007–2013.

Chapelle, O., Manavoglu, E., and Rosales, R. (2015). Simple and scalable response prediction for display advertising. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(4) :61.

Cheng, H.-T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., et al. (2016). Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10. ACM.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38.

Evans, D. S. (2009). The online advertising industry : Economics, evolution, and privacy. *Journal of economic perspectives*, 23(3) :37–60.

Fisher, R. A. (1919). Xv.—the correlation between relatives on the supposition of mendelian inheritance. *Earth and Environmental Science Transactions of the Royal Society of Edinburgh*, 52(2) :399–433.

Friedman, J. H. (2001). Greedy function approximation : a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Graepel, T., Candela, J. Q., Borchert, T., and Herbrich, R. (2010). Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. Omnipress.

Grewal, D., Bart, Y., Spann, M., and Zubcsek, P. P. (2016). Mobile advertising : a framework and research agenda. *Journal of Interactive Marketing*, 34 :3–14.

Guo, H., Tang, R., Ye, Y., Li, Z., and He, X. (2017). Deepfm : a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv :1703.04247*.

He, X., Pan, J., Jin, O., Xu, T., Liu, B., Xu, T., Shi, Y., Atallah, A., Herbrich, R., Bowers, S., et al. (2014). Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, pages 1–9. ACM.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2(1) :193–218.

Juan, Y., Lefortier, D., and Chapelle, O. (2017). Field-aware factorization machines in a real-world online advertising system. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 680–688. International World Wide Web Conferences Steering Committee.

Juan, Y., Zhuang, Y., Chin, W.-S., and Lin, C.-J. (2016). Field-aware factorization machines for ctr prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 43–50. ACM.

Kaye, B. K. and Medoff, N. J. (2001). *World Wide Web : a mass communication perspective*. McGraw-Hill Higher Education.

Kondakindi, G., Rana, S., Rajkumar, A., Ponnekanti, S. K., and Parakh, V. (2014). A logistic regression approach to ad click prediction. *Mach Learn Class Project*.

Kumar, R., Naik, S. M., Naik, V. D., Shiralli, S., Sunil, V., and Husain, M. (2015). Predicting clicks : Ctr estimation of advertisements using logistic regression classifier. In *2015 IEEE International Advance Computing Conference (IACC)*, pages 1134–1138. IEEE.

Li, C., Lu, Y., Mei, Q., Wang, D., and Pandey, S. (2015). Click-through prediction for advertising in twitter timeline. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1959–1968. ACM.

Liu, W., Tang, R., Li, J., Yu, J., Guo, H., He, X., and Zhang, S. (2018). Field-aware probabilistic embedding neural network for ctr prediction. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 412–416. ACM.

McCullagh, P. and Nelder, J. A. (1989). *Generalized Linear Models*, volume 37. CRC Press.

McCulloch, C. E. and Neuhaus, J. M. (2005). Generalized linear mixed models. *Encyclopedia of biostatistics*, 4.

McLachlan, G. and Peel, D. (2004). *Finite mixture models*. John Wiley & Sons.

McLachlan, G. J. and Krishnan, T. (2007). *The EM algorithm and extensions*, volume 382. John Wiley & Sons.

Murphy, K. P. (2012). *Machine learning : a probabilistic perspective*. MIT press.

Nelder, J. A. and Wedderburn, R. W. (1972). Generalized linear models. *Journal of the Royal Statistical Society : Series A (General)*, 135(3) :370–384.

Oentaryo, R. J., Lim, E.-P., Low, J.-W., Lo, D., and Finegold, M. (2014). Predicting response in mobile advertising with hierarchical importance-aware factorization machine. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 123–132. ACM.

Pan, Z., Chen, E., Liu, Q., Xu, T., Ma, H., and Lin, H. (2016). Sparse factorization machines for click-through rate prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 400–409. IEEE.

Pearson, K. (1894). Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. A*, 185 :71–110.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336) :846–850.

Ren, K., Zhang, W., Rong, Y., Zhang, H., Yu, Y., and Wang, J. (2016). User response learning for directly optimizing campaign performance in display advertising. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 679–688. ACM.

Rendle, S. (2010). Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000. IEEE.

Rendle, S. (2012). Social network and click-through prediction with factorization machines. In *KDD-Cup Workshop*, page 113.

Richardson, M., Dominowska, E., and Ragno, R. (2007). Predicting clicks : estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web*, pages 521–530. ACM.

Rousseeuw, P. J. (1987). Silhouettes : a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20 :53–65.

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The annals of statistics*, 6(2) :461–464.

Searle, S. R., Casella, G., and McCulloch, C. E. (2009). *Variance components*, volume 391. John Wiley & Sons.

Titterington, D. M., Smith, A. F., and Makov, U. E. (1985). *Statistical analysis of finite mixture distributions*. Wiley,.

Wang, R., Fu, B., Fu, G., and Wang, M. (2017). Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*, page 12. ACM.

Yan, L., Li, W.-J., Xue, G.-R., and Han, D. (2014). Coupled group lasso for web-scale ctr prediction in display advertising. In *International Conference on Machine Learning*, pages 802–810.

Yuan, S., Abidin, A. Z., Sloan, M., and Wang, J. (2012). Internet advertising : An interplay among advertisers, online publishers, ad exchanges and web users. *arXiv preprint arXiv :1206.1754*.

Yuan, S., Wang, J., and Zhao, X. (2013). Real-time bidding for online advertising : measurement and analysis. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*, page 3. ACM.

Yuan, Y., Wang, F., Li, J., and Qin, R. (2014). A survey on real time bidding advertising. In *Proceedings of 2014 IEEE International Conference on Service Operations and Logistics, and Informatics*, pages 418–423. IEEE.

Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H., and Gai, K. (2018). Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1059–1068. ACM.

Annexe A

binomialMix : la création d'un package R

R-package binomialMix tutorial

Description

The **binomialMix** package provides a clustering method for longitudinal and non gaussian data. It uses an EM algorithm for GLM. For now, a model-based clustering for mixture of binomial data is available.

STEP 1: Installation

You can install the `binomialMix` R package with the following R command:

```
# install.packages("devtools")
devtools::install_git("https://gitlab.com/tabmo/binomialmix")
devtools::install_gitlab("tabmo/binomialMix")
```

You can also directly use the git repository :

```
git clone https://gitlab.com/tabmo/binomialMix
```

Once you cloned the git repository, you can run to install the `binomialMix` package:

```
devtools::install("/path/to/binomialMix/pkg") # edit the path
```

STEP 2: Use-case tutorial

Imagine that you are working for an advertising company. You need to make groups of campaigns with similar profiles.

1. First, you need to import the following library:

```
# our library for mixture modelling:
library(binomialMix)
# if not installed :
#install.packages("pander", repos="http://cran.us.r-project.org")
#install.packages("ggplot2", repos="http://cran.us.r-project.org")
#library(pander)
library(qpdf)
library(futile.logger)
```

2. Let's have a look at the dataset:

```
data(adcampaign)
```

```
##   id timestamp_ymd yearDay day timeSlot app_or_site impressions click
## 1 14    2019-01-01       1   3        1         app        2675   117
## 2 14    2019-01-01       1   3        1         app         729    16
## 3 14    2019-01-01       1   3        2         app        1016    33
## 4 14    2019-01-01       1   3        2         app         342     6
## 5 14    2019-01-01       1   3        3         app        3431    92
## 6 14    2019-01-01       1   3        3         app         864     9
##          ctr
## 1 0.04373832
## 2 0.02194787
## 3 0.03248031
## 4 0.01754386
## 5 0.02681434
## 6 0.01041667
```

NB : Of course, you can use your own data. The format you need to have is the following:

- a *dataframe* type is needed (ex: **adcampaign** from binomialMix)
- a column with *factor* id representing the objects you want to cluster (ex: **id** from adcampaign)
- a target value (ex: **ctr** from adcampaign)
- a weighted value variable as we are in case of binomial data (ex: **impressions** from adcampaign)
- at least, one column as *explicative variable* (ex: **day** from adcampaign)

3. Let's make some clusters!

The objective of the study is to group advertising campaigns into clusters. We observe by campaign, time slot, day of week and ad slot campaign (like app or site) the observed number of clicks and impressions. CTR corresponds to the number of click on the number of impressions. CTR value differs a lot from one observation to another, as well as the total length of a campaign. Some last fews days and others broadcast for months. Then, each campaigns (**column "id"**) is composed of n_c observations from the whole dataset and we have repeated mesure for a same id level. The available explicative variables are:

- day
- timeSlot
- app_or_site

Let's now try to cluster our dataset into K groups.

```
# The dataframe to cluster:
df_tocluster<-adcampaign
# We choose two explainable variables:
model_formula<-"ctr~timeSlot+day"
# As we are in a case of binomial mixture model, we define the weighted variable
weighted_variable<-"impressions"
# We want to analyse results for K=3.
K<-3
# We define the individual to cluster:
col_id<-"id"
set.seed(1992)
# We run our EM algorithm developped for mixture of binomial and longitudinal dataset:
result_K3<-runEM(model_formula,
                 weighted_variable,
                 K,
```

```
                df_tocluster,
                col_id)
```

4. Analysis of clustering results:

The output of the runEM function provides the following values:

1. Loglikelihood for each EM iteration
2. Estimation of model parameters (β, λ, π)
3. BIC and ICL values
4. Number of fisher iteration needed for each M-Step

Plotting evolution of Loglikelihood over iteration

```
library(ggplot2)
qplot(seq_along(result_K3[[1]]), result_K3[[1]],
    xlab="Number of EM iterations",
    ylab="Loglikelihood")
```

Estimated β parameters

Let's have a look at the estimated parameters for each cluster k. We only show the estimation from the last EM iteration in the following.

```
result_K3[[3]][[length(result_K3[[3]])]]
```

```
##              k=1         k=2        k=3
## [1,] -3.27524617 -6.02001952 -5.0421272
## [2,]  0.31581767  0.12798712  0.4729844
## [3,]  0.20178096  0.26338824  0.5358001
## [4,]  0.26372976  0.43174257  0.7074703
## [5,]  0.09812297  0.41739277  0.8413444
## [6,]  0.08388539  0.09588596  0.7098054
```

Estimated proportion of campaigns λ for each cluster

We want to have a look at the repartition of our campaigns for adcampaign dataset to analyze the size of each cluster. We only display value for the last iteration of EM algorithm.

```
result_K3[[3]][[length(result_K3[[3]])]]
```

```
## [1] 0.114300 0.498075 0.387625
```

Matrix of proability for each campaign to belong to the different clusters

We analyze the contribution of each campaign to the K clusters. The columns define the campaigns and the rows the different cluster k.

```
# We only display the results for the first 10 campaigns (10 columns)
set.seed(1992)
result_K3[[4]][[length(result_K3[[4]])]][,1:10]
```

```
##     ID_1 ID_2 ID_3 ID_4 ID_5 ID_6 ID_7  ID_8  ID_9 ID_10
## k=1    0    0    0    0    0    0    0 0.000 0.000     0
## k=2    0    0    1    0    0    0    1 0.999 0.096     1
## k=3    1    1    0    1    1    1    0 0.001 0.904     0
```

Analyze of BIC and ICL values

The analyze of BIC and ICL values is essential when we want to choose the right number of clusters. We can compare BIC/ICL values and choose the K that minimize one or both of these criteria.

```
result_K3[[5]][[length(result_K3[[5]])]] # BIC value
result_K3[[6]][[length(result_K3[[6]])]] # ICL value
```

```
## [1] "BIC=372360.14"
```

```
## [1] "ICL=372367.72"
```

Analyze of Fisher scoring number of iterations for each M step

If we want to know the number of Fisher scoring iterations at each M step, we can display the following matrix.

```
matrix(unlist(result_K3[[7]]),ncol=length(result_K3[[7]])-1)
```

```
##     iter_1 iter_2 iter_3 iter_4 iter_5 iter_6 iter_7 iter_8
## k=1      4      3      3      2      1      1      1      1
## k=2      4      3      1      3      2      1      1      1
## k=3      4      3      3      2      2      1      1      1
```

Annexes du Chapitre 2

B.1 Détails de l'estimation des proportions du mélange

$$\sum_{k=1}^{K} -\alpha\lambda_k = \sum_{k=1}^{K}\left(\sum_{c=1}^{C}\pi_{kc}\right) \tag{B.1}$$

$$\Leftrightarrow -\alpha \times 1 = \sum_{c=1}^{C}\left(\sum_{k=1}^{K}\pi_{kc}\right) \tag{B.2}$$

$$\Leftrightarrow -\alpha = \sum_{c=1}^{C} \tag{B.3}$$

$$\Leftrightarrow \alpha = -C \tag{B.4}$$

B.2 Détails du calcul de l'espérance et la variance d'un GLM

$$\frac{\partial l}{\partial \theta_{cjht}} = \frac{\partial}{\partial \theta_{cjht}}\left(\frac{y_{cjht}\theta_{cjht} - b(\theta_{cjht})}{a_{cjht}(\psi)} + c(y_{cjht}, \psi)\right) \Leftrightarrow \frac{\partial l}{\partial \theta_{cjht}} = \frac{y_{cjht} - b'(\theta_{cjht})}{a_{cjht}(\psi)} \tag{B.5}$$

$$\frac{\partial^2 l}{\partial^2 \theta_{cjht}} = \frac{\partial}{\partial \theta_{cjht}}\left(\frac{y_{cjht} - b'(\theta_{cjht})}{a_{cjht}(\psi)}\right) \Leftrightarrow \frac{\partial^2 l}{\partial^2 \theta_{cjht}} = \frac{-b''(\theta_{cjht})}{a_{cjht}(\psi)} \tag{B.6}$$

De l'équation (B.5), on en déduit que :

$$\begin{aligned} E\left(\frac{\partial l}{\partial \theta_{cjht}}\right) = 0 &\Leftrightarrow \frac{y_{cjht} - b'(\theta_{cjht})}{a_{cjht}(\psi)} = 0 \\ &\Leftrightarrow \frac{1}{a_{cjht}(\psi)}\left(E(y_{cjht}) - b'(\theta_{cjht})\right) = 0 \\ &\Leftrightarrow E(y_{cjht}) = b'(\theta_{cjht}) \end{aligned}$$

De l'équation (B.6), on en déduit que :

$$\begin{aligned} E\left(\frac{\partial l^2}{\partial \theta_{cjht}{}^2}\right) = -E\left(\frac{\partial l}{\partial \theta_{cjht}}\right)^2 &\Leftrightarrow E\left(\frac{-b''(\theta_{cjht})}{a_{cjht}(\psi)}\right) = -E\left(\frac{y_{cjht}-b'(\theta_{cjht})}{a_{cjht}(\psi)}\right)^2 \\ &\Leftrightarrow E\left(\frac{-b''(\theta_{cjht})}{a_{cjht}(\psi)}\right) = -\frac{1}{a_{cjht}(\psi)^2}E\left(y_{cjht}-E(y_{cjht})\right)^2 \\ &\Leftrightarrow E\left(\frac{-b''(\theta_{cjht})}{a_{cjht}(\psi)}\right) = -\frac{Var(y_{cjht})}{a_{cjht}(\psi)^2} \\ &\Leftrightarrow Var(y_{cjht}) = b''(\theta_{cjht})a_{cjht}(\psi) \end{aligned}$$

B.3 Détails de l'estimation des β du mélange binomial

Calcul de W : Pour rappel, on a la matrice diagonale définie de la manière suivante :

$$W_{c\beta_k} = diag\Big(a_{cjht}(\psi_k)b''(\theta_{cjht_k})g'(\mu_{cjht_k})^2\Big)\ \forall c,j,h,t \tag{B.7}$$

— Si on regarde la dérivée de la fonction de lien g :

$$\begin{aligned} g'(\mu_{cjht}) &= \left((\log\left(\frac{\mu_{cjht}}{1-\mu_{cjht}}\right)\right)' \\ &= \frac{\frac{1(1-\mu_{cjht})+\mu_{cjht}}{(1-\mu_{cjht})^2}}{\frac{\mu_{cjht}}{1-\mu_{cjht}}} \\ &= \frac{1}{(1-\mu_{cjht})^2}\frac{(1-\mu_{jcht})}{\mu_{cjht}} \\ &= \frac{1}{(1-\mu_{cjht})\mu_{cjht}} \\ &= \frac{1}{(1-\frac{\exp\theta_{cjht}}{1+\exp\theta_{cjht}})\frac{\exp\theta_{cjht}}{1+\exp\theta_{cjht}}} \\ &= \frac{(1+\exp\theta_{cjht})^2}{\exp\theta_{cjht}} \\ &= \frac{(1+\exp M_{cjht}\beta_k)^2}{\exp M_{cjht}\beta_k} \end{aligned}$$

comme $\mu_{cjht} = b'(\theta) = \dfrac{\exp\theta_{cjht}}{1+\exp\theta_{cjht}}$ on a :

— Si on regarde maintenant la dérivée seconde de la fonction b :

$$\begin{aligned} b''(\theta_{cjht})) &= \frac{\exp\theta_{cjht}(1+\exp\theta_{cjht}) - (\exp\theta_{cjht})^2}{(1+\exp\theta_{cjht})^2} \\ &= \frac{\exp\theta_{cjht}}{(1+\exp\theta_{cjht})^2} \\ &= \frac{\exp M_{cjht}\beta_k}{(1+\exp M_{cjht}\beta_k)^2} \end{aligned}$$

On obtient donc :

$$W_{c\beta_k} = diag\left(\frac{1}{n_{cjht}}\frac{(1+\exp M_{cjht}\beta_k)^2}{\exp M_{cjht}\beta_k}\right) \forall c,j,h,t \tag{B.8}$$

Calcul de $\frac{\partial(\eta_k)}{\partial(\mu_k)}$: Comme on a par défintiion que $g(\mu) = \eta$, on en déduit que $\frac{\partial(\eta_k)}{\partial(\mu_k)} = g'(\mu)$ (voir calcul plus haut pour avoir les détails de $g'(\mu)$). Au final,

$$\frac{\partial \eta_k}{\partial \mu_k} = diag\left(\frac{(1+\exp M_{cjht}\beta_k)^2}{\exp M_{cjht}\beta_k}\right) \forall c,j,h,t \tag{B.9}$$

Annexes du Chapitre 3

C.1 Compléments des résultats expérimentaux pour la prédiction

Résultats obtenus pour les jeux de test du mois d'Octobre 2019 des différents modèles prédictifs

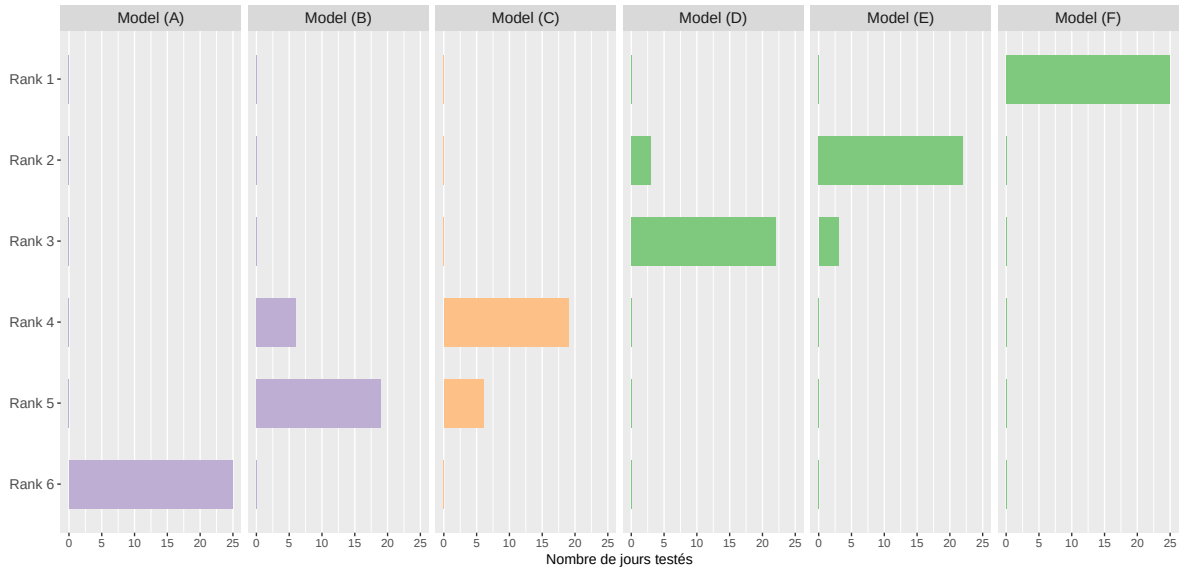


Figure C.1 – Classement des 6 modèles (A),(B),(C),(D),(E) et (F) en compétition pour les 30 jours de test du mois d'Octobre 2019. Le *Rank 1* correspond au modèle dont la logloss est minimale tandis que le *Rank 6* correspond au modèle qui a la plus mauvaise logloss pour un jeu de test donné.

Résultats obtenus pour les jeux de test du mois de Décembre 2019 des différents modèles prédictifs

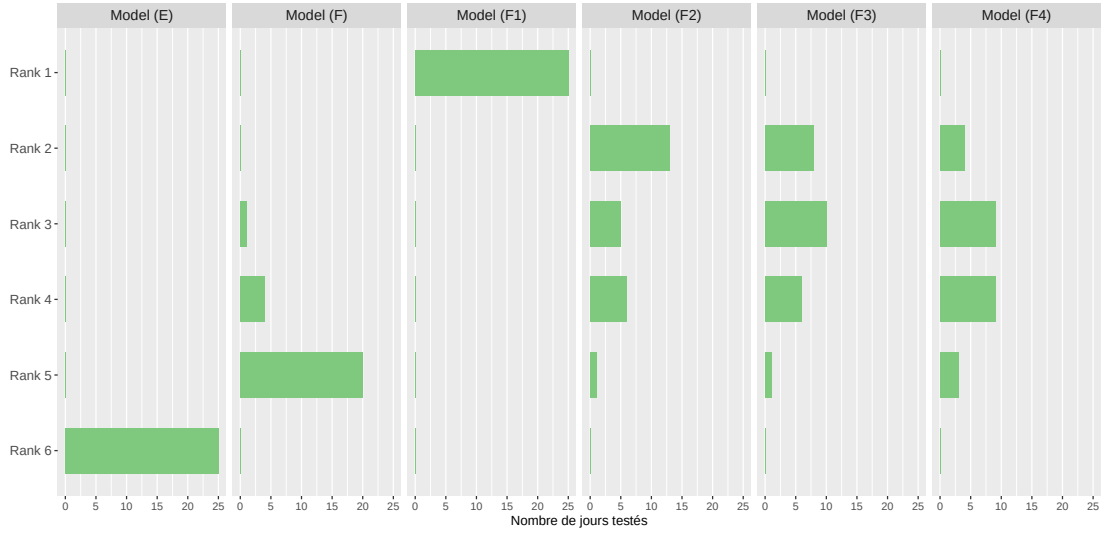


FIGURE C.2 – Classement des 6 modèles (D),(F),(F1),(F2),(F3) et (F4) en compétition pour les 30 jours de test du mois d'Octobre 2019. Le *Rank 1* correspond au modèle donc la logloss est minimale tandis que le *Rank 6* correspond au modèle qui a la pire logloss pour un jeu de test donné.

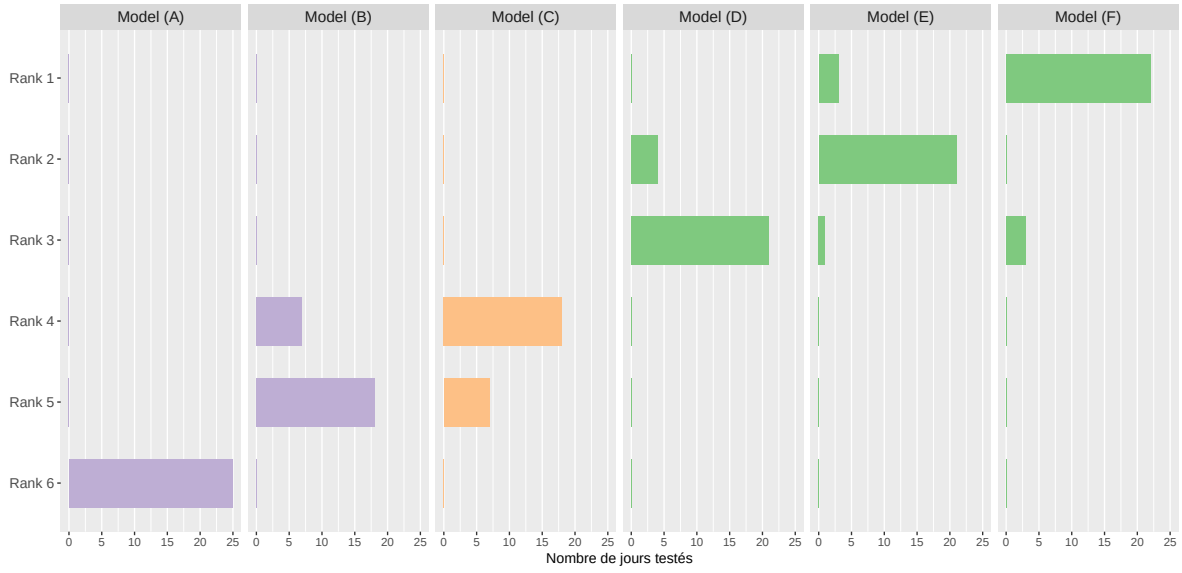


FIGURE C.3 – Classement des 6 modèles (A),(B),(C),(D),(E) et (F) en compétition pour les 30 jours de test du mois de Décembre 2019. Le *Rank 1* correspond au modèle donc la logloss est minimale tandis que le *Rank 6* correspond au modèle qui a la pire logloss pour un jeu de test donné.

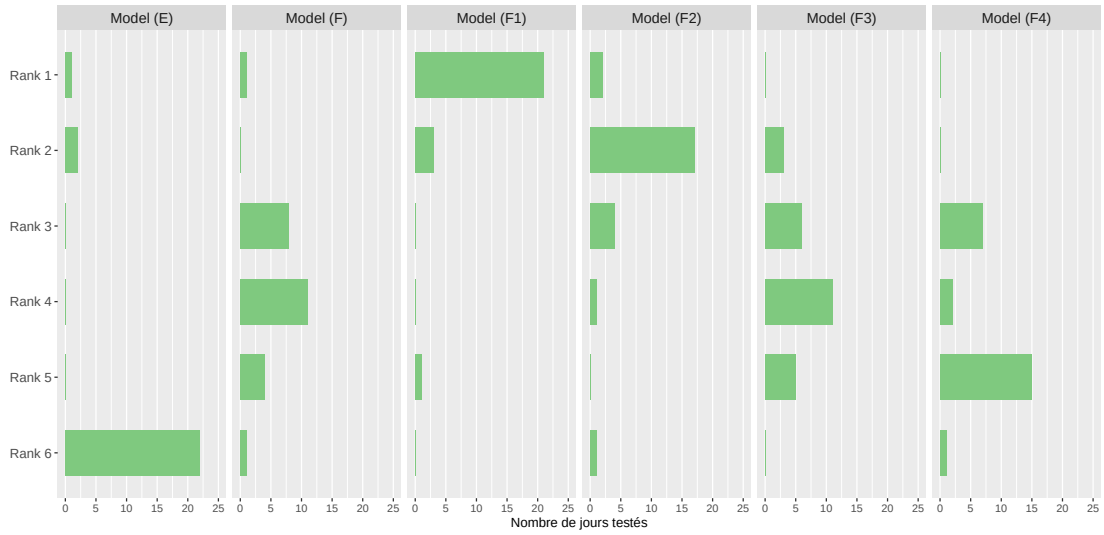


FIGURE C.4 – Classement des 6 modèles (D),(F),(F1),(F2),(F3) et (F4) en compétition pour les 30 jours de test du mois de Décembre 2019. Le *Rank 1* correspond au modèle donc la logloss est minimale tandis que le *Rank 6* correspond au modèle qui a la pire logloss pour un jeu de test donné.