Rapport de stage de master

Fabrice ZAPFACK

Scoring d'opportunités de covoiturage

Tuteur : M. Tristan croiset
Responsable pédagogique : M. Alexandre gramfort

Stage effectué dans le cardre du
Master M2 Data Sciences
Ecole Poytechnique

Avril 2016 — Août 2016

Résumé

Mon stage de Master s'est déroulé au sein de l'équipe "Science" de la start-up Karos SAS dont le but est de révolutionner la mobilité urbaine à travers le covoiturage domicile-travail.

Dans le cadre de cette mission, l'entreprise développe une application permettant aux utilisateurs d'être mis en relation, de façon automatique, avec d'autres potentiels covoitureurs.

L'objectif de mon stage était alors d'évaluer la pertinence des opportunités de covoiturage ainsi proposées et d'estimer la probabilité que ces dernières soient converties en véritable covoiturage. L'objectif était alors d'utiliser ce score afin de filtrer et de classer les différentes opportunités.

Remerciements

Je tiens tout d'abord à remercier M. Tristan Croiset, mon encadrant chez Karos, pour son accompagnement tout au long de ce stage et pour la confiance, les conseils et l'aide qu'il m'a prodigués et grâce auxquels j'ai pu mener mes travaux dans d'excellentes conditions.

Je souhaite ensuite témoigner toute ma reconnaissance à l'ensemble de l'équipe Science pour son accueil chaleureux et son soutien : M. François Kawala, Louis Dacquet et Assirem Medjer. Merci messieurs pour tout ce que vous m'avez appris et j'espère atteindre votre niveau un jour.

Je remercie également toute la team Karos : Jérôme Calot, Aurelien Guillard, Maria-Isabelle Galbert et Camille Noel. J'adresse un merci particulier à Olivier Binet pour la confiance et les conseils prodigués tout au long de ce stage. Mes chaleureux remerciements s'adressent aussi à mes collègues stagiaires avec lesquels j'ai partagé de nombreux moments de convivialité et qui m'ont été d'une grande aide dans l'avancement de mes travaux : Eva Porée et Sophie Maurel.

Je remercie également le personnel de l'Ecole Polytechnique, pour l'accompagnement qui a été le leur. Je pense en particulier à M. Alexandre Gramfort et M. Erwan Le Pennec, qui ont fait preuve de disponibilité et qui m'ont apporté le soutien administratif dont j'avais besoin tout au long de ce stage.

Enfin, je remercie ma famille pour son soutien, ses encouragements et prières, ainsi que toute personne ayant contribué de près ou de loin à la réalisation de ce stage.

Table des matières

Première partie

Contexte industriel

Chapitre 1

Présentation de l'entreprise

Fondée en Juin 2014, **Karos SAS** est le fruit d'une idée portée par **Olivier Binet**, diplômé d'HEC en 2003, qui travaillait à ce momment en tant qu'investisseur en capital. Ce dernier, impressioné par le succès de la licorne Française Blablacar, se demandait pourquoi le covoiturage sur les trajets courts n'avait pas autant de succès. Il vint alors à la conclusion qu'il faudrait développer une solution qui, à l'aide d'intelligence artificielle, ferait ce travail d'organisation à la place de l'utilisateur.

Il s'est alors rapproché de **Tristan Croiset**, diplomé d'EPITA en 2008 et qui travaillait alors chez Criteo. Tous les deux débutèrent ainsi la merveilleuse aventure Karos.

1.1 Activités

Karos SAS ambitionne de révolutionner la mobilité urbaine à travers le covoiturage domicile-travail. Dans ce cardre, l'entreprise développe une solution innovante de covoiturage spécifiquement dédiée aux trajets quotidiens, de courtes distances, permettant l'émergence d'un nouveau réseau de transport éco-responsable. Le produit alors développé est une application mobile permettant aux utilisateurs d'être automatiquement mis en relation avec de potentiels covoitureurs. Ce produit exploite des technologies de géolocalisation et de gestion des données massives. Ainsi, plusieurs verrous technologiques sont levés pour rendre possible, pour la toute première fois, le covoiturage sur courte distance, en adaptant cette pratique aux trajets quotidiens. Les utilisateurs sont mis automatiquement en relation lorsque leurs trajets correspondent. Le partage des coûts est calculé lui aussi de manière automatique.

L'entreprise développe également des partenariats avec des entreprises et des collectivités qui souhaitent augmenter l'accessibilité de leurs sites, parfois éloignés des réseaux de transport en commun, et réduire leur empreinte carbonne. L'impact direct de Karos sur les déplacements quotidiens permet d'envisager une réduction significative des émissions de CO2 et des coûts liés à la congestion automobile.

La solution apportée par Karos est d'autant plus pertinente qu'elle vise à résoudre un réel problème de mobilité urbaine : 82% des ménages français possèdent au moins une

voiture et 93% d'entre eux déclarent l'utiliser faute d'alternatives. Chère, la voiture est souvent considérée comme un mal nécessaire. En effet ; les offres de transports publics ne peuvent répondre à tous les besoins de mobilités individuelles de manière optimale. Le projet de Karos est donc de faire de la voiture individuelle un transport en commun à part entière.

FIGURE 1.1 – Réseau de transport vu par Karos

Trois éléments principaux (ADEME, 2015) permettent de corroborer la vison de Karos sur le marché du court-covoiturage en France :

— Dans 90% des déplacements quotidiens en voiture, le conducteur est seul dans son véhicule. Ceci représente une gigantesque réserve latente de mobilité, extrêmement capillaire. Ainsi, rien qu'en Ile-de-France, 40 millions de sièges libres réalisent un trajet tous les jours.
— Le covoiturage est une pratique en pleine explosion sur les trajets longs (330 km en moyenne par jour chez Blablacar). De plus, 3% des déplacements domicile-travail se font déjà en covoiturage "informel".
— Il existe des technologies permettant d'adapter le covoiturage aux enjeux et contraintes des trajets quotidiens. Notamment, les algorithmes d'apprentissage statistique et les technologies de gestion des données massives permettent d'appréhender efficacement la complexité des déplacements. De plus, les dernières technologies mobiles offrent des fonctionnalités critiques en termes de captation de données, de géolocalisation et de communication en temps réel entre utilisateurs.

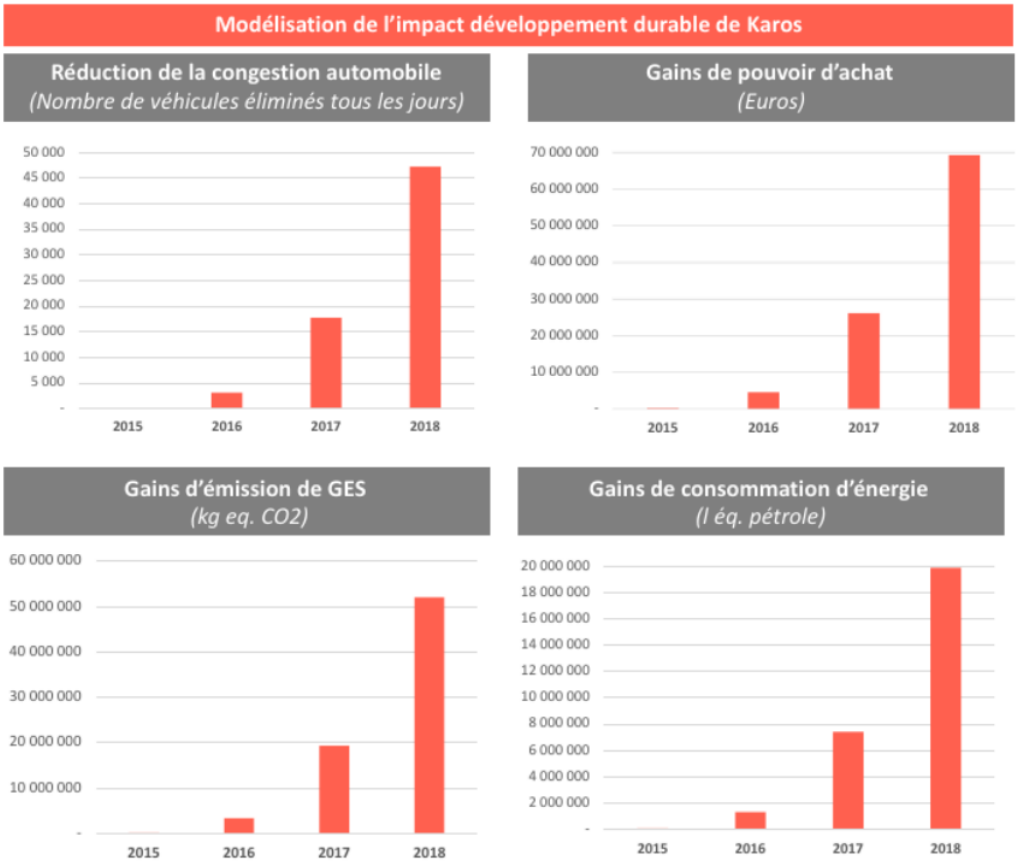

FIGURE 1.2 – Impact sur le développement durable de Karos

1.2 Equipe

A mon arrivée chez Karos, j'ai pu constater que les **neuf** employés de l'entreprise étaient regroupés en **deux** équipes à savoir la team **"Science"** et la team **"Produit"**. Ces deux équipes fontionnent en parfaite synergie avec pour objectif principal de **"rendre l'expérience des utilisateurs la plus simple possible"**.

Team "Produit"

Compte tenu du stade d'avancement de l'entreprise, cette équipe a pour principale mission de faire grandir la communauté des utilisateurs Karos. Elle est alors chargée de tous les aspects marketing, sales, business development, community management ou encore customers success de la compagnie. Elle est constiuée de 3 personnes à savoir :

- **Olivier Binet** : CEO et responsable de tous les aspects business development
- **Jérôme Calot** : Responsable Marketing
- **Camille Noel** : Responsable de la Satisfaction Clientèle

Team "Science"

Afin d'apporter un service aussi irréprochable à ses clients, Karos s'appuie sur une architecture technique exceptionnelle. Cette dernière est le fruit du travail acharné d'une armée de "génies du code" à savoir :

— **Tristan Croiset** : CTO et responsable du développent de la technologie de l'entreprise
— **Aurelien Guillard** : Développeur Android
— **Maria-Isabelle Galbert** : Développeur IOS
— **François Kawala** : Data Scientist & Full-Stack
— **Louis Dacquet** : Data Scientist
— **Assirem Medjer** : Data Scientist

Chapitre 2

Zoom sur la technologie Karos

2.1 Vue globale du service Karos

Karos est une application mobile agissant comme un assistant personnalisé de mobilité quotidienne. Cette application est une interface, reliée à un moteur d'apprentissage statistique. Les étapes d'utilisation sont les suivantes :

1. L'utilisateur télécharge Karos gratuitement et s'inscrit.
2. Karos apprend alors automatiquement ses habitudes de déplacements quotidiens, tout en consommant peu de batterie du smartphone. En effet, Karos n'active pas le GPS mais tourne " en tâche de fond " en agrégant l'information provenant de différents capteurs (GSM, wifi, accéléromètre, ...)
3. Les algorithmes d'apprentissage statistique permettent ensuite d'anticiper l'ensemble des déplacements et des besoins de mobilité sur les 48 prochaines heures.
4. Karos exploite toutes les opportunités de mise en relation entre les potentiels conducteurs et les potentiels passagers.
5. Karos gère ensuite de bout en bout le covoiturage :
   — un module de géolocalisation se met automatiquement en route peu avant le début du covoiturage, afin que passager et conducteur puissent se retrouver facilement
   — La prise en charge et la dépose du passager sont automatiquement prises en compte par Karos, qui calcule ainsi la distance parcourue en covoiturage.
   — La participation aux frais payée par le passager au conducteur est donc calculée. Puis les paiements électroniques se font automatiquement.

Pour assurer ces différents services, un nombre de choix ont été faits lors du développement de la stack de Karos. Ainsi, l'archichecture est basée sur une application REST conçue sur **Django**, un framework open-source de développement web. Une des particularités de ce framework est la simplicité de routing à travers un "remapping" d'URL. Une autre raison à mon avis qui justifie le choix de ce framework est le fait qu'il soit codé en python, ce qui rend plus simple l'ajout de services d'analyse de données. Aussi, Karos a fait le choix de déployer son application sur **Google Cloud Platform**, la plateforme de cloud computing

fournie par Google. Je pense qu'en plus du coût relativement faible, l'une des raisons du choix de cette plateforme est la facilité de déploiement d'applications grâce à la technologie container et leurs communications via des microservices Web.

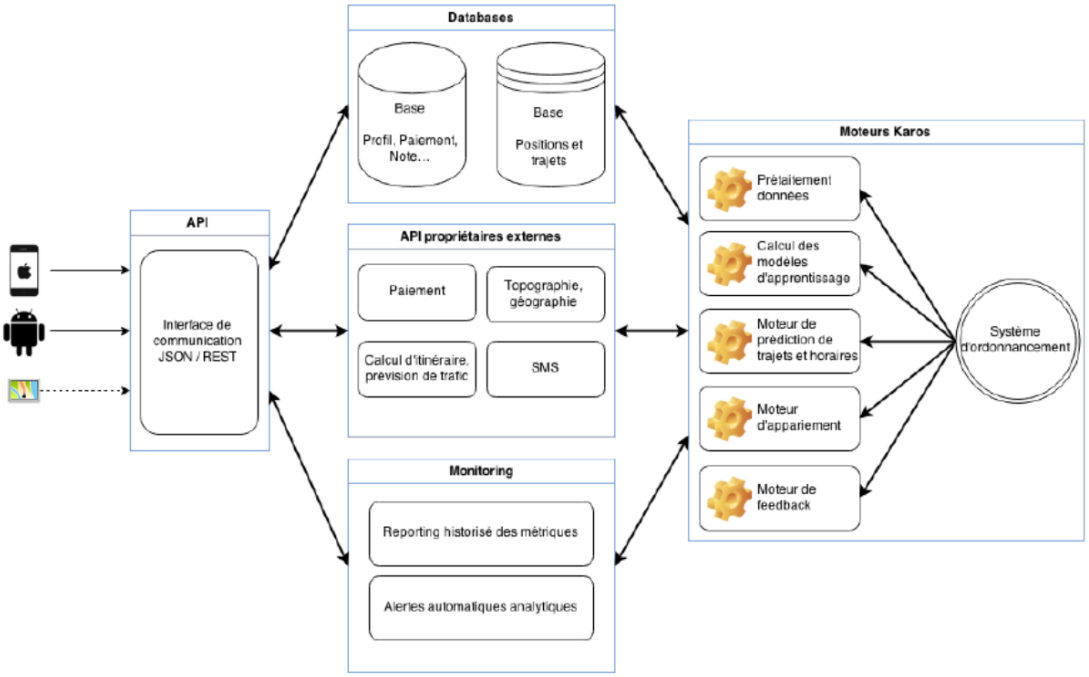

FIGURE 2.1 – Architecture technique de Karos

2.2 Description du service Matching

Dénommé **Fastmatch**, ce bijou de technologie développé par l'entreprise a pour but de permettre aux utilisateurs de trouver des covoitureurs en un temps record. L'entreprise a ainsi développé une nouvelle architecture technique découpée en micro-services spécialisés. Chaque module est chargé d'une tâche bien précise. Par exemple : prédire la durée du covoiturage ou bien estimer si le détour pour l'utilisateur et son covoitureur est acceptable.

Des algorithmes et des heuristiques se chargent de déterminer les calculs à réaliser en priorité et de paralléliser les tâches. Par exemple lorsque l'utilisateur ajoute un trajet, il faut le comparer avec des centaines d'autres pour trouver les meilleures opportunités. Karos a développé un service de ranking qui trie en quelques millisecondes tous les trajets à examiner. Il est très efficace et dans 90% des cas, le meilleur covoiturage se trouve parmi les 3 premiers. L'application affichera donc un premier résultat bien avant d'avoir analysé tout le réseau Karos !

Chapitre 3

Objectifs du stage et plan du rapport

3.1 Objectifs du stage

Grâce aux différentes briques technologiques développées par Karos, plus de 80% des utilisateurs Karos sont capables de trouver au moins un covoitureur dans les heures qui suivent leur "on-boarding" dans l'application. Cependant une très faible partie de ces utilisateurs réagissent à ces matchs et une partie encore plus petite seulement convertissent ces opportunités de covoiturage en demande de covoiturage. Ainsi moins de **2%** des opportunités de covoiturage calculées par Karos aboutissent à une demande de covoiturage et moins de **10%** des demandes sont converties en covoiturage.

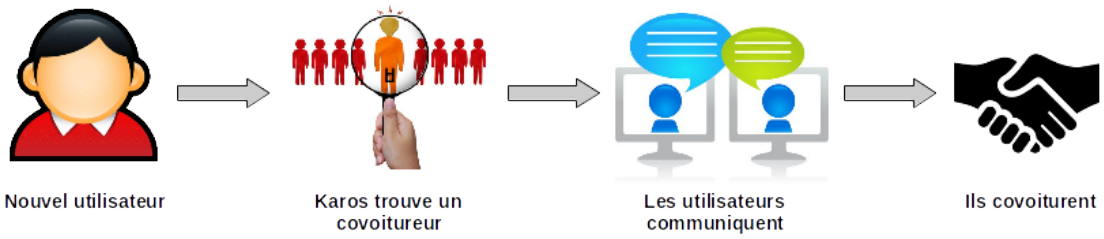

FIGURE 3.1 – Processus de covoiturage

A mon arrivée, l'une des priorités de l'entreprise était alors de trouver des raisons qui rendraient certaines opportunités plus intéressantes et surtout d'être certain que les utilisateurs aient accès aux opportunités les plus intéressantes (celles qui ont le plus de chance d'être converties en demande de covoiturage ou voire en covoiturage).

Dans ce contexte l'objectif principal de mon stage était alors de proposer une méthodologie permettant de *classer les opportunités de covoituarage proposées à un utilisateur en fonction de la probabilité que ces dernières soient converties en demande de covoiturage*. Le problème ainsi défini peut se formaliser en problème de *Learning to rank* (wikipedia, 2016b). Avant d'attaquer ce type de poblème, nous avons décidé qu'il serait plus approprié, dans un premier temps, de résourdre un autre problème moins compliqué à savoir **la**

classification d'opportunités de covoiturage. Les raisons ayant motivé ce choix vous seront présentées dans la première partie de ce rapport.

La démarche que j'ai utilisée pour atteindre cet objectif peut se résumer en 3 phases :

1. La constitution d'une base de données réelles : obtenue à partir des données réelles issues d'informations récupérées par l'entreprise. Le jeu de données ainsi constitué permettant la classification des opportunités de covoiturage calculées par Karos.
2. Le développement de modèles statistiques : En m'appuyant sur les différents travaux existants dans la littérature, cette phase permet de proposer et d'implémenter des modèles de classification.
3. La validation des modèles : Les modèles proposés seront par la suite testés puis comparés. Une étude sera par la suite faite pour analyser les résultats obtenus et juger de leur pertinence.

3.2 Plan du rapport

La démarche présentée ci-dessus est expliquée en détails dans la suite de ce rapport.

La prochaine section du rapport présente la méthodologie utilisée dans le processus de classification des opportunités de covoiurage. Cette partie permettra également de décrire les données utilisées durant toute notre analyse.

La troisième partie sera l'occasion de présenter les options que nous avons envisagées et de justifier les choix qui ont été faits.

La quatrième partie détaillera le déroulement et les contraintes liées à la mise en production de ces modèles.

Je finirai enfin par une synthèse et une analyse critique des travaux effectués. J'y présenterai également les enseignements tirés de cette expérience.

Deuxième partie

Travail préliminaire

Chapitre 1

Formalisation du problème

1.1 Ranking des opportunités de covoiturage

Nous avons rappelé dans la première partie de ce rapport que l'un des objectifs principaux de Karos au début de mon stage était de proposer une méthodologie permettant de classer les opportunités de covoiturage, en d'autres termes, à effectuer un "ranking" de ces opportunités de covoiturage.

Definition 1.1 (Ranking)**.** Le **"Ranking"** peut se définir comme une classe de problèmes dont l'objectif est d'apprendre une fonction à valeurs réelles permettant de classer des exemples de l'ensemble de départ (Qian et al., 2013).

Definition 1.2 (Learning to rank)**.** Le **"Learning to rank"** ou **"machine-learned ranking"** (MLR) pourra alors se définir comme l'application du machine learning afin de construire les modèles de ranking (wikipedia, 2016b).

Definition 1.3 (Bipartite ranking)**.** Dans ce cadre le **"bipartite ranking"** pourra être défini comme une catégorie de ranking dans lequel les exemples sont issus de deux classes : une classe positive et une classe négative. Le but est alors d'apprendre, grâce à ces exemples, une fonction de ranking qui permette de classer de futurs échantillons tel que les échantillons positifs aient un meilleur rang que les échantillons négatifs (Agarwal, 2005).

En d'autres termes, ce problème peut s'exprimer sur la forme suivante :

- (X, Y) random pair, valued in $\mathbb{R}^d \times \{-1, +1\}$ with $d >> 1$
- **Observation:** sample $\mathcal{D}_n$ of i.i.d. copies of (X, Y)

$$(X_1, Y_1), \ldots, (X_n, Y_n)$$

- **Goal:** from labeled data $\mathcal{D}_n$, learn to **order** new data $X'_1, \ldots, X'_{n'}$

$$\begin{array}{cccccc} X'_7 & X'_{n'-2} & X'_3 & X'_6 & \ldots \\ + & + & - & + & \ldots \end{array}$$

in order to recover **positive instances on top of the list** with large probability

1.2 Liens entre le "bipartite ranking et la classification binaire"

Les définitions ci-dessus permettent d'identifier le lien entre les problèmes de "Bipartite ranking" et ceux de classification. Il apparait par exemple que les problèmes de "learning to rank" se formulent de la même façon que des problèmes de "classification binaire" à la différence près que des problèmes de ranking ont une nature "globale" alors que ceux de classification ont une nature "locale" (Clémençon, 2016).

Comme expliqué dans la première partie de ce rapport, nous avons décidé d'attaquer le problème de la classification d'opportunités de covoiturage comme sujet principal de ce stage. L'objectif étant alors de **proposer une démarche, par analyse statistique de données, permettant de distinguer les opportunités qui seront converties en opportunités de covoiturage de celles qui ne le seront pas**. Deux raisons principales ont motivé ce choix :

— Il était crucial pour l'entreprise de comprendre les critères qui rendent les opportunités de covoiturage plus "intéressantes" pour l'utilisateur.

— D'après les définitions ci-dessus, on peut aisément se rendre compte que les modèles de classifcation qui seraient développés pourraient facilement être appliqués à du ranking par la suite.

Chapitre 2

Collecte des données

Un travail important durant ce stage fut de prendre connaissance de l'ensemble des données stockées par l'entreprise ainsi que celles qui pourraient être récupérées afin de résourdre le problème posé. Les données utilisées tout au long de ce stage provenaient essentiellement de deux sources : Une **base de données relationnelle"** et un outil de "Mobile Analytics" dénommé **"Mixpanel"**.

2.1 Base de données

La majorité des informations que nous avons utilisées étaient stockées dans deux bases de données relationnelles : une base de donnée de **"production"** contenant diverses informations liées à l'utilisation quotidienne de l'application et une base de données de **"pré-production"** qui est un replicat "light" de la première ne contenant qu'une partie de ces données. La politique de copie de ces données est interne à l'entreprise.

La technologie utilisée pour ces bases de données est **"MySQL"** et ces dernières sont déployées dans le cloud (Google Cloud Platform), ce qui rend leur accès plus rapide de la part des différents micro-services qu'utilise l'application. Les données y sont stockées dans de multiples tables dont :

- Une table **"Users"**, contenant des données relatives aux utilisateurs Karos telles que le nom, la ville ou la date d'inscription.
- Une table **"Matchs"**, contenant des données relatives aux opportunités de covoiturage calculées par Karos tels que les identifiants du conducteur et du passager ou encore les dates et horaires de départ et arrivée de ce trajet.
- Une table **"Matchs-steps"**, qui permet de suivre l'évolution de ces matchs. Elle permet notamment d'enregistrer le moment où un match a été converti en demande de covoiturage ou encore les dates et horaires de départ et d'arrivée effective des covoitureurs.

Ces bases de données étaient accessibles à la fois en local (afin d'analyser les données localement) et à distance (en passant par des clés stockées dans "Buckets").

2.2 Mixpanel

"Mixpanel" est un outil de d'analyse statistique qui permet de "tracer" l'ensemble des intéractions entre les utiliseurs et une application (qu'elle soit Android ou iOS) (wikipedia, 2016a).
Karos a ainsi choisi cette solution afin d'enregistrer en temps réel la façon dont chacun de ses utilisateurs utilise l'application. L'objectif étant alors de récupérer des données permettant de faire de l'A/B testing et ainsi de comprendre comment amméliorer le service proposé aux clients.
Les données ainsi stockées sont accessibles à travers des api mises à la disposition des clients. L'accès à ces api se fait via des requêtes écrites en language **"JQL"**.
Le **"JQL"** (JavaScript Query Language) est une façon d'utiliser le langage Javascript afin d'analyser des données brutes. Il permet ainsi de bénéficier de la puissance et la flexibilité du JavaScript afin de produire des requêtes allant des plus simples aux plus complexes. Le code ainsi produit se révèle être plus simple et compréhensible que son équivalent en **"SQL"**. De plus le JQL à la particularité d'être un langage "fonctionnel", contrairement au SQL qui lui est "déclaratif" (Mixpanel, 2016).

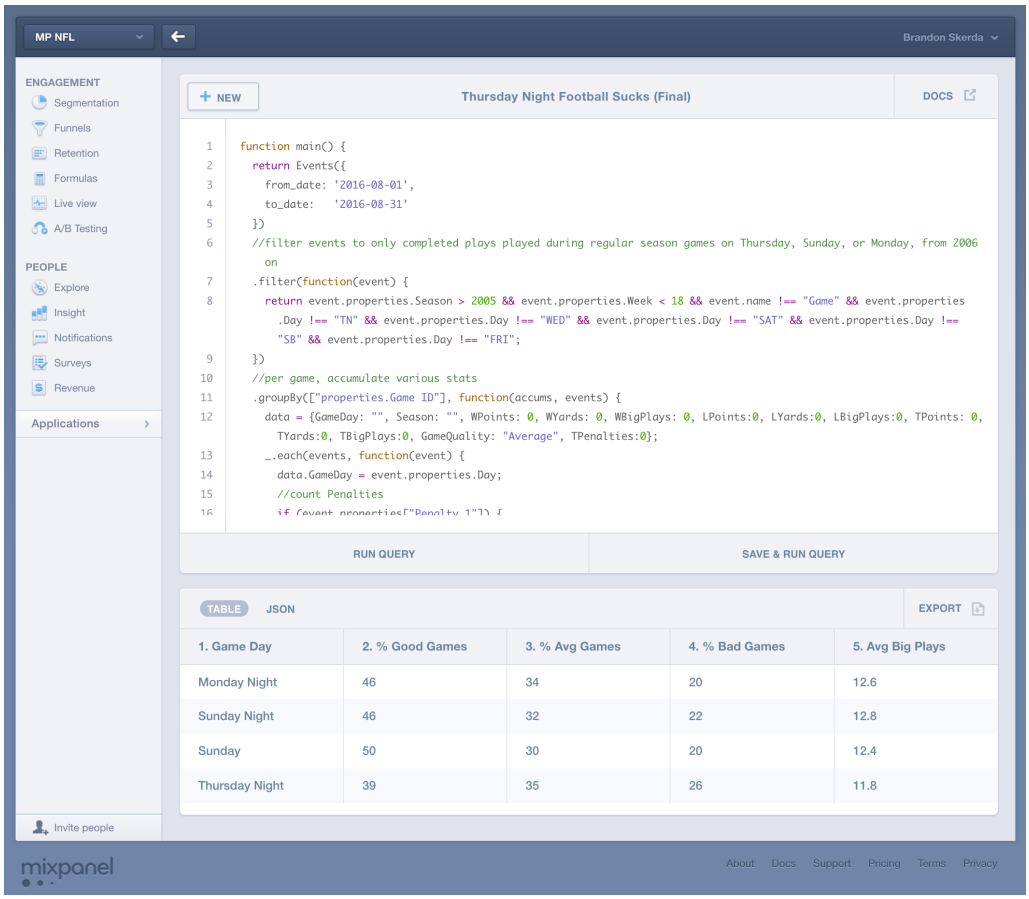

FIGURE 2.1 – Exemple de code sur mixpanel

Chapitre 3

Extraction des variables

Les données présentées ci-dessus furent par la suite utilisées afin de créer des variables pertinentes pour résoudre notre problème. En effet, il est important de rappeler que l'un des objectifs de l'entreprise au cours de ce stage était d'identifier les facteurs qui rendent certaines oppotunités de covoiturage plus "intéressantes" que d'autres pour les utilisateurs. Une grande partie de ce stage fut donc consacrée à proposer des variables qui permettraient de résourdre ce problème, à les calculer à partir des données disponibles et ensuite à vérifier statistiquement leur pertinence.

3.1 Description des variables

Durant ce stage, notre travail s'est porté sur l'analyse de **matchs** (opportunités de covoiturage) proposés par l'entreprise. Le jeu de données présenté dans ce rapport contenait **127324** matchs extraits entre le **06/05/2015** et le **07/06/2016**.
La première étape après avoir extrait ces données consistait à affecter un "Label" à chacun de ces matchs. 4 labels étaient ainsi envisagés :

- **"Accepted"** : Il s'agit là d'indiquer si une opportunité de covoiturage a été convertie en covoiturage effectif, c'est-à-dire si le conducteur et le passager ont accepté de réaliser le trajet. Malheureusement seuls **502** des **127324** matchs analysés ont été "accepted" (0.39%). Pour cette raison, il était impossible d'utiliser ce label pour notre analyse.
- **"Driver-interact"** : Il s'agit ici d'indiquer si, pour un match donné, le conducteur à manifesté un intérêt. Cet intérêt se traduit soit par un message envoyé au passager, soit par une demande de covoiturage envoyée via l'application ou soit par le fait d'accepter une demande reçue du passager. Sur les **127324** matchs analysés, **2033** ont "Driver-interact" vrai (1.6%).
- **"Passenger-interact"** : Ce label est similaire au label précédent à la différence près que dans ce cas il représente cette fois l'intérêt manifesté par le passager. Sur les **127324** matchs analysés, **2124** ont "Passenger-interact" vrai (1.68%).
- **"Changed"** : Ce label est une combinaison des deux précédents dans le sens où il représente les matchs pour lesquels le conducteur **ou** le passager ont se sont montrés

intéressés. Sur les **127324** matchs analysés, **3055** sont positifs (2.4%). Ce label fut choisi pour la suite de notre travail.

Une fois les labels extraits pour chacun des matchs, nous nous sommes attelés à extraire les variables permetttant de décrire chacune des observations. Ces descripteurs ont été construits à partir des données disponibles (voir chapitre 2). Une liste non exhaustive des variables extraites est diponible en annexe[.1]. Certains descripteurs ont été obtenus simplement par extraction des données ("driver-detour-duration" ou "driver-has-picture" par exemple). D'autres par contre ont nécéssité d'avantage de calculs ("weigthed-matchs-couple" par exemple).

3.2 Pré-traitement des variables

Compte tenu de l'importance cruciale qu'elles ont pour la validation de notre démarche, un soin particulier a été apporté aux pré-traitement des variables extraites afin de s'assurer de leur "Fiabilité". Parmi ces pré-traitements, on peut citer :

- **"Remplissage de données manquantes"** : Nous avons en effet été confrontés à l'absence d'information pour quelques un des exemples que nous devions analyser. Nous avons au cas par cas (en fonction de la variable) décidé de la meilleure solution à apporter à ce problème. En ce qui concerne les variables catégorielles, nous avons en général opté pour la création d'une nouvelle catégorie réprésentant cette absence d'information.
- **"Normalisation"** : En fonction des modèles de classification utilisés, cette étape s'est révélée être plus ou moins importante. Nous avons opté pour une "min/max normalization".
- **"outliers"** : Après avoir calculé des statistiques pour chacune des variables (moyenne, médiane, variance,...), nous avons porté une attention particulière aux points qui nous paraissaient abérants. Nous avons défini les outliers comme des points tels que :
$$|x - \bar{x}| \ > 2 * \sigma(x)$$
- **"Catégorisation"** : Il est également très important de réfléchir à une stratégie idéale afin de transformer les variables catégorielles. Le choix s'est porté entre un "label encoding" et un "one-hot encoding". Ainsi, s'agissant des variables possédant un nombre limité de catégories, nous avons réalisé un "one-hot encoding". Par contre les variables possédant un nombre élevé de catégories ont été transformées par "label encoding" afin d'éviter de faire exploser la dimensionnalité du problème.
- **"Réduction de dimension"** : Nous avons limité autant que possible le nombre de variables afin d'éviter d'avoir à utiliser une technique de réduction de dimension qui rendrait l'analyse de l'importance des variable plus compliquée.

Le jeux de données obtenu à l'issu de ce pré-traitement était constitué de **127324** échantillons (matchs) dont **3055** possédaient un label positif (2.4%). Chacun de ces échantillons était représenté par **184** variables.

Troisième partie

Modélisation

Chapitre 1

Choix des modèles

1.1 Méthodes de classification supervisée

La classification supervisée peut se définir de la manière suivante :

Definition 1.1 (classification supervisée)**.** ’
- Soit une variable d’entrée $X = (X^{(1)}, X^{(2)}, \ldots, X^{(3)} \in \chi$
- Soit une variable de sortie $Y \in \{-1, 1\}$
- $(X, Y) \sim P$, P une distribution inconnue
- Les données $D_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, (i.i.d. $\sim$ P)
- Trouver une fonction qui approxime in $F = f : \chi \to \{-1, 1\}$

Nous allons présenter ci-dessous quelques unes des méthodes de classification utilisées durant ce stage.

classification naïve bayésienne Le classifieur naïf bayésien est l’une des méthodes d’apprentissage supervisé les plus simples basée sur le théorème de Bayes. Il s’appuie sur le fait que les descripteurs sont conditionnellement indépendants c’est-à-dire qu’il suppose que l’existence d’une caractéristique pour une classe est indépendante de l’existence d’autres caractéristiques. La mise en place de cette méthode se fait en plusieurs étapes :
- On détermine les probabilités à priori de chaque classe.
- On applique la règle de Bayes pour obtenir la probabilité à posteriori des classes au point x :
$$P(y = c|X = x) = \frac{P(X = x|Y = c) * P(Y = c)}{P(X = x)}$$
.
- On choisit la classe la plus probable.

Support Vectors Machine Historiquement, les "Support Vectors Machines" ou "SVM" sont connus comme la prémière méthode "à noyaux" appliquée dans le domaine de la reconnaissance de forme. Ces méthodes jouissent encore d’une grande popularité de nos jours en raison de leurs bonnes performances (Boser et al., 1992). Ces méthodes peuvent être formulées sous la forme d’un problème d’optimisation convexe (Mairal, 2016) :

$$\min_{f\in\mathcal{H},\boldsymbol{\xi}\in\mathbb{R}^n} \left\{ \frac{1}{n}\sum_{i=1}^{n}\xi_i + \lambda\| f \|_{\mathcal{H}}^2 \right\},$$

subject to:

$$\xi_i \geq \varphi_{\text{hinge}}\left(\mathbf{y}_i f\left(\mathbf{x}_i\right)\right).$$

Nous avons également envisagé l'utilsation des SVM via une approche de **classification mono-classe**. L'objectif de cette approche était d'utiliser les SVM pour apprendre à reconnaître les instances d'une des classes (la classe négative en l'occurrence) afin d'identifier les instances de l'autre classe comme outliers de la première classe.
Diverses approches ont été proposées afin d'utiliser les SVM pour résoudre les problèmes de classification mono-classe. Tax and Duin (1999) a proposé une technique, appelée Support Vector Data Description (SVDD) consistant à déterminer la sphère (de centre c et de rayon r) contenant la majorité des observations d'apprentissage.
Une autre variante consiste à déterminer l'hyperplan qui sépare au mieux les données de l'espace vectoriel où elles sont représentées (Schölkopf et al., 1999). Ma and Perkins (2003a) et Ma and Perkins (2003b) utilisent cette technique pour la détection d'anomalies : après avoir projeté les séries temporelles dans un autre espace, ils utilisent une variante de la one-class SVM appelée **Support Vector Regression**. Les SVR ont l'avantage de donner une indication sur la probabilité que l'observation détectée soit une anomalie.

Formulation mathématique La première approche du problème comme on l'a précisé plus haut revient à déterminer la sphère optimale contenant les données. On peut introduire des pénalités autorisant certaines observations à se trouver à l'extérieur de la sphère. Cela peut être formulé de la façon suivante :

$$\begin{gathered} \min_{r,c,\xi} r^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i, \nu \textit{ tant un paramtre} \\ \textit{sous la contrainte}, \|\Phi(x_i) - c\|^2 \leq r^2 + \xi_i \textit{ pour tous } i \end{gathered} \tag{1.1}$$

En résolvant cette équation par une méthode de Lagrange, les KKT conditions nous permettent d'obtenir :

$$c = \sum_{i=1}^{n}\alpha_i \Phi x_i$$

Random Forest :
Les Random forests correspondent à des combinaisons de plusieurs arbres de décisions. Les arbres sont construits à partir de vecteurs aléatoires indépendants et identiquement distribués. Lors de la construction de l'arbre, une selection aléatoire des features est opérée à chaque noeud. Chaque arbre donne un classifieur que nous noterons $h(x, \theta_k)$ où x est le vecteur d'entrée et θ_k une instance du vecteur aléatoire θ.

Definition 1.2 (Random Forest)**.** :
Un random forest est un classifieur correspondant à un ensemble de classifieurs issus d'arbre $h(x, \theta_k),\ k = 1, ..$ où les θ_k sont des vecteurs aléatoires indépendants et identiquement distribués et l'ensemble des arbres donne lieu à un vote majoritaire pour la classe la plus populaire pour le vecteur d'entrée x.

Plusieurs raisons justifient le choix des Random Forest :
— Elles offraient les meilleures performances de toutes les modèles étudiés. Seuls les "Gradient Boosting" (implémentées à travers Chen and He (2015)) nous ont permis d'obtenir des résultats du même ordre.
— En terme de vitesse (à l'entrainemt comme à la phase de test) elle étaient largement meilleures que les "Gradient Boosting" ou "SVM".
— Elles donnent la possibilité d'étudier l'importance de chacune des variables dans le modèle, ce qui est d'une importance capitale pour Karos qui souhaite comprendre les critères qui rendent des opportunités de covoiturage plus "intéressantes" que d'autres afin d'améliorer sa qualité de service.

1.2 Classification Asymétrique

L'une des plus grande difficulté recontrée durant ce stage fut "**l'asymétrie des données**". En effet, nous avons indiqué dans le chapitre [chapitre 3] que le ratio de matchs ayant un label négatif par rapport à ceux qui ont un label positif était d'environ **1 : 50**.
Le problème de **Classification assymétrique** est très courant lorsqu'on analyse des données réelles. Un certain nombre de solutions sont proposées dans la littératures. Ces solutions peuvent être regroupées en trois groupes (Fernández et al., 2011).

Modification du jeu de données

L'objectif de ces méthodes est d'obtenir un jeu de données moins asymétrique soit en retirant des observations de la classe majoritaire, soit en rajoutant des observations de la classe minoritaire. Un des risques évident de ces méthodes serait la perte d'information liée à la suppression de certaines observations. Un autre risque serait lié à la création du nouveau jeu de données "artificiel" qui aurait une distribution différente du jeu de données "réel", ce qui serait à l'origine de modèle éloignés de la réalité. Deux techniques qui rentraient dans ce groupe de solution seraient :

- Filtrer ou acquérir de nouvelles données :
Filtrer les données de la classe majoritaire revenait dans notre cas à supprimer certains matchs ayant un label négatif, tout en conservant tous les matchs ayant un label positif. Celà s'est réalisé en supprimant les matchs considérés comme "outliers" (voir section 3.2). Cette approche s'est avérée inefficace.
L'acquisition de nouvelles observations positives était dans notre cas impossible car elle serait suivie par davantage observations négatives. En effet pour Karos, le nombre d'opportunités de covoiturage évoluent plus rapidement que le nombre d'opportunités converties à cause de plusieurs facteurs (nouveaux utilisateurs, nouveaux modes de transport multimodaux, ...).

- Ré-échantillonner le jeu de données :
Cette approche consiste à effectuer un **sur-échantillonnage**, un **sous-échantillonnage** ou alors un mix des deux. Nous avons implémenté différentes techniques de ré-échantillonage à travers le paquet "**imbalanced-learn**". Nous avons considéré des ré-échantillonnages aléatoires ou non (heuréstiques), et et différentes valeurs de ratio.

- **Sur-échantillonnage** : l'utilisation de ces techniques était justifiéé compte tenu du nombre peu élevé d'échantillons positifs. Les techniques implémentées sont : "Random majority under-sampling with replacement", "Extraction of majority-minority Tomek links" et "Under-sampling with Cluster Centroids".
- **Sous-échantillonnage** : l'utilisation de ces techniques était justifiéé compte tenu du nombre important d'échantillons négatifs. Les techniques implémentées sont : "Random minority over-sampling with replacement", "SMOTE" et "bSMOTE".
- **Mix des deux** : Il s'agit soit d'effectuer un sur-échantillonage suivi d'un sous-échantillonnage ("SMOTE + ENN ", "SMOTE + Tomek links") ou d'utiliser une méthode ensembliste ("EasyEnsemble", "BalanceCascade ").

Même si certaines de ces techniques ont produit des résultats encourageant, nous avons pris la décision de ne pas les implémenter en production car nous n'avons pas réussi à les optimiser et réduire les temps de calculs qu'elles entrainaient.

Modification des algorithmes

Afin d'utiliser les modèles de classification binaire présentés ci-desus, certaines modifications ont dues être apportées afin de les adapter au problème de classification asymétrique que nous rencontrions :

- **Choix de la mesure de performance** : l'utilisation de l'accuracy comme mesure de performance n'est pas valable dans ce cas car implicitement celà reviendrait à sélectionner les modèles ayant tendance à classifier les observations comme négatives. Au contraire, notre objectif était d'optimiser la capacité des modèles à reconnaître les observations positives d'où la nécessité d'utiliser d'autres mesures de performances. Nous reviendrons plus en détails sur ces mesures dans la prochaine partie du rapport.
- **Pénaliser les modèles** : l'objectif de ces techniques est d'apporter des coûts additionnels chaque fois que les modèles commettent des erreurs de classification sur des exemples de la classe minoritaire durant la phase d'apprentissage ; ces coûts additionnels ont pour objectif de biaiser les modèles afin qu'ils apportent une plus grande attention à la classe minoritaire.
  La majorité des modèles de la librairie scikit-learn implémentent cette option à travers les paramètres **class weight** et **sample weight**. Quelques modèles sont également conçus pour explitement résourdre ce problème (Weighted Random Forest, penalized-SVM, penalized-LDA, ...).

Interprétation du problème

Nous avons également essayé d'aborder les problèmes sur d'autres angles qui en théorie auraient permi de mieux prendre en compte l'asymétrie du jeu de données. Parmis ces différentes approches, on peut citer :

— **Détection d'anomalies** : Il s'agit d'un domaine consacré à la détection d'évènements rares et peu fréquents comparés à la majorité des événements considérés comme "normaux". La classification mono-classe est l'une des méthodes les plus couramment utilisées pour résoudre ces problèmes. Nous avons montré plus tôt dans ce rapport comment nous avons essayé d'utiliser les SVM mono-classe pour résoudre ce problème. D'autres algorithmes sont proposés dans la littérature (Parzen density estimation ou Positive Naive Bayes Classifier).

— **Méthodes ensemblistes** : L'idée derrière ces méthodes peut se résumer en "l'union fait la force". En effet, ces techniques consistent à obtenir une prédiction finale en combinant plusieurs autres prédicteurs (Brownlee, 2016). Différentes approches sont envisageables dans ce cas.

Une première approche serait de diviser les observations de la classe majoritaire en différents groupes et d'utiliser les observations de la seconde classe afin d'entraîner un modèle pour chacun des groupes. La prédiction du label d'une nouvelle observation se fera ensuite en combinant ces différents modèles (vote par majorité par exemple). Les implémentations de ce type de techniques diffèrent énormement les unes des autres notamment en fonction de la façon dont sont construits les groupes : de façon experte ou par classification non-supervisée. Durant ce stage nous avons envisagé différentes approches dont notamment en séparant les matchs en fonction du mois de leur création. Cette approche s'est avérée inefficace.

Une autre approche serait de créer plusieurs modèles en utilisant une ou plusieurs des techniques présentées ci-dessus (en utilisant différentes techniques de ré-échantillonage par exemple) et en les combinant afin d'obtenir un modèle plus robuste.

Une autre approche que j'ai trouvée très intéressante est issue de l'article Zhang et al. (2008). Il présente un algorithme "FloatCascade" qui est une sorte d'adaboost à plusieurs niveaux. Cette méthode consite à utiliser de nombreux "prédicteurs faibles" possédant une précision proche de 1 mais un rappel (sensibilité) très faible. En utilisant Adaboost (se concentrer sur les observations mal classifiées à l'étape précédente), l'objectif est alors de combiner ces derniers afin d'améliorer la sensibilité du modèle résultant tout en conservant la bonne précision.

Chapitre 2

Optimisation des modèles

Nous avons montré dans la section précédente que différents paramètres devaient être pris en compte afin de déterminer le modèle de classification optimal. Il faut également rappeler que chaque modèle possède à son tour de nombreux méta-paramètres à optimiser. Nous avons autant que possible essayé, à chaque fois qu'un choix devait être fait, de sélectionner le modèle permettant d'obtenir les meilleures perfomances de classification selon un critère bien déterminé.

2.1 Mesure de performance

Dans la section précédente de ce rapport, nous avons montré la nécessité d'utiliser une mesure de performance autre que l'**accuracy** en raison de l'asymétrie importante de la distribution des données. En effet, maximiser ce critère convenait implicitement à sélectionner les modèles ayant tendance à classifier toutes observations comme appartenant à la classe négative. L'objectif de notre analyse était par contre de maximiser la capacité des modèles à reconnaître les observations de la classe positive. Pour évaluer cette capacité, quatre informations doivent être prises en compte :

- TP = Vrais Positifs = Nombre d'exemples prédits positifs à raison.
- FP = Faux Positifs = Nombre d'exemples prédits positifs à tort.
- TN = Vrais Négatifs = Nombre d'exemples prédits négatifs à raison.
- FN = Faux Négatifs = Nombre d'exemples prédits négatifs à tort.

Ces informations sont en général présentées sous la forme de ratio qui témoignent de la qualité des modèles : le **rappel** et la **précision** :

- $Rappel = \frac{TP}{TP+FP}$
- $Prcision = \frac{TP}{TP+FN}$

Afin de maximiser à la fois la précision et le rappel, les statisticiens utilisent en général plusieurs techniques dont :

- La **courbe precision-recall** :
  Il s'agit juste de la courbe qui montre l'évolution de la précision en fonction du rappel selon la configuration du modèle. Le modèle idéal est celui qui a $Rappel = Precision = 1$ quelque soit la configuration. L'idée est alors de choisir le modèle

qui s'en rapproche le plus c'est-à-dire celui qui a la "Surface sous la courbe" (AUC) la plus élevée.

— La **F1 score** :
Il s'agit de la moyenne harmonique de la précision et du rappel d'un modèle donné. Pour différentes configurations d'un modèle, seule celle offrant le meilleur score est alors conservée.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Cette formule montre que ce score tend à produire des modèles ayant une précision et rappel du même ordre. Par exemple un modèle ayant une précsion de 1 et un rappel de 0.1 aura un score inférieur à un autre modèle ayant une précision et un rappel de 0.2. L'objectif que l'on souhaitait obtenir était pourtant d'obtenir une précision élevée (proche de 1) et un rappel correct. Nous avons proposé d'autres métriques basées sur des moyennes de la précision et du rappel, avec des poids fixés de façon heuréstique sans trop de succès. Nous nous sommes finalement résolus à utilser le score F1 comme mesure de peformance des modèles mais en fixant un seuil minimal de précision.

2.2 Validation croisée

Il est également important de présenter la méthode de validation utilisée au cours de cette analyse. En effet la technique la plus utilisée en analyse de données est la technique dite de **validation croisée** qui consiste à séparer les obsersations utilisées pour entrainer les modèles de ceux utilisés pour les évaluer.
Une contrainte supplémentaire devait être prise en compte : il fallait tenir compte de l'ordre chronologique. En effet, l'évolution dans le temps des variables nous imposait de vérifier que toutes les observations de test étaient postérieurs à celles utilisées pour entrainer les modèles.
Nous avons alors utilisé une version modifiée de la **K-folds cross-validation** dans laquelle les observations sont d'abords ordonnées par ordre chronologique avant d'être divisées en groupes. Ensuite lorsqu'un groupe est choisi pour évaluer les modèles, seuls les groupes antèrieurs à ce dernier peuvent être utilisés pour l'apprentissage.Les observations dans chaque groupe sont par la suite rangés de façon aléatoire.

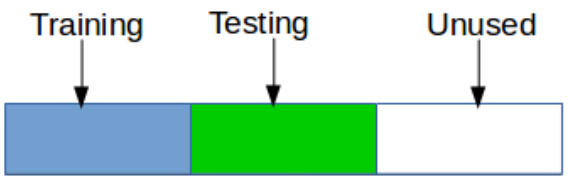

FIGURE 2.1 – Illustration de la validation croisée

2.3 Selection des variables

Nous avons dans un premier temps entraîné le modèle en utilisant l'ensemble des variables afin d'avoir une idée de l'importance de chacune d'entre elles. Les résultats sont présentés ci-dessous.

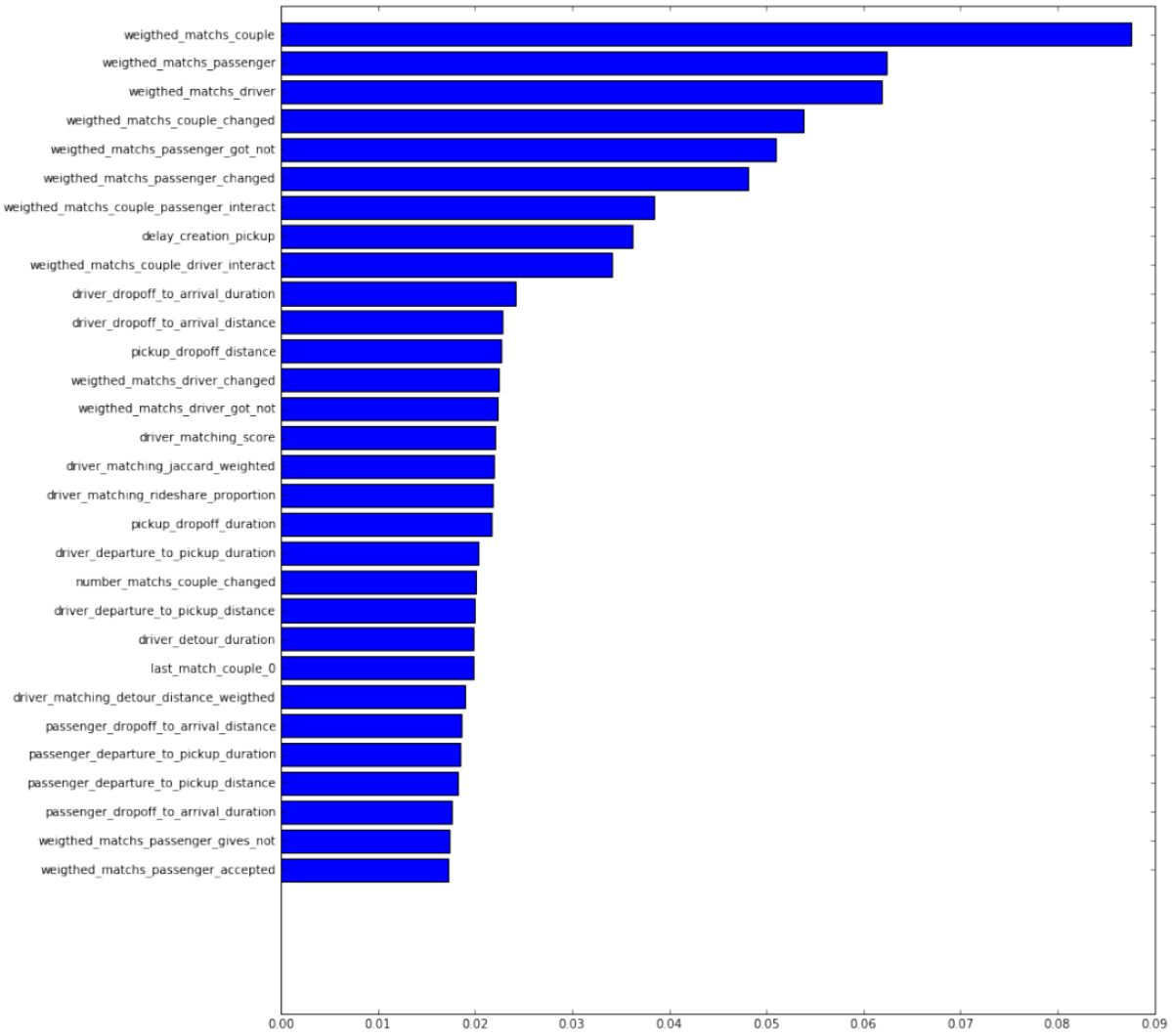

FIGURE 2.2 – Importance des variables pour le modèle

On note par contre que les variables les plus importantes apportent des informations plus ou moins similaires (intuition confirmée par la corrélation entre elles). Nous avons alors essayé d'implémenter des techniques de sélection de variables mais aucun des modèles résultant n'étaient aussi performant que le modèle initial.

Quatrième partie

Mise en production

Après avoir sélectionné le modèle de classification optimal et optimisé ses paramètres, une partie importante du stage fut consacrée à la mise en production de ce dernier afin qu'il puisse fonctionner avec les différents services de l'application. L'application fonctionnant en micro-services, celà est passé par la création d'un service spécifique nommé "scoring" qui devrait être appelé après la création d'une opportunité de covoiturage afin d'en évaluer la qualité.
Dans cette partie du rapport, nous allons dans un premier temps décrire le service implémenté puis nous présenterons quelques résultats obtenus en situation réelle.

Chapitre 1

Description du service

1.1 Description générale

Rôle :
Le service "Scoring" est chargé de calculer la probabilité qu'un match donné (opportunité de covoiturage) suscite de l'intérêt chez au moins l'un des potentiels covoitureurs (l'un d'entre eux envoie un message ou une demande de covoiturage).
A terme il devrait permettre de calculer la probabilité qu'un match soit effectivement converti en covoiturage.

Accès au service :
Profitant de la flexibilité qu'offre Django et le routing d'URLs, l'accès à la majorité des services de l'application se fait à travers une URL spécifique. Un paramètre est requis afin d'accéder au service. Ce dernier correspond à l'identifiant du match à analyser.
La sécurité est assurée grâce à une authentification par token. Pour un match 000, l'URL correspondante sera de la forme :

$$https://base-url/scoring?match-id = 000$$

Format de retour :
Une fois les calcul terminés, le service renvoie un résultat au format JSON qui comprend entre autre un champ "Output" qui contient la probabilité que le match donne lieu à une intéraction.

1.2 Description Détaillée

Le service ainsi mis en place permettait dans l'ordre d'effectuer de façon automatique les opérations nécessaires : Extraction des données, calcul des variables, entrainement des modèles et prédiction.

Result : *Proba*, Score réprésentant la qualité d'un match
Parameter : $Match - id$ Identifiant du match à analyser

*(*Une fois que le service est appelé) ;
if *Première requête du service* **then**

1. Extraire tous les matchs qui devraient avoir eu lieu.
2. Calculer les labels correspondants à ces matchs.
3. Prétraitement et features extraction
4. Calcul des variables du match à analyser
5. Entraînement d'un modèle
6. Prédiction et renvoie de *Proba*

end
else

1. Mise à jour des matchs qui devraient avoir eu lieu.
2. Mise à jour des labels correspondants à ces matchs.
3. Mise à jour des features
4. Calcul des variables du match à analyser
5. Prédiction et renvoie de *Proba*

end

Algorithme 1 : Description du fonctionnement du service "Scoring"

Chapitre 2

Test du modèles

Une fois créé, le service fut par la suite testé sur des jeu de données réels afin d'en évaluer les performances. Il a été comparé à deux critères empiriques qu'utilisait Karos pour juger de la qualité d'une opportunité de covoiturage :

— **GOOD** : "driver-matching-rideshare-proportion" > 0.5 & "driver-matching-detour-distance-weigthed" < 2500.
— **GOLD** : "driver-matching-rideshare-proportion" > 0.8 .

2.1 Sur un jeu de données important

Ce jeu de données contenait **230716** opportunités de covoiturage calculées par Karos entre le **06/05/2015** et le **11/08/2016**. Il a été divisé en un jeu de données de test et un autre d'apprentissage :

— **X-train** : **178341** opportunités de covoiturage calculées par Karos entre le **06/05/2015** et le **08/07/2016**. **2,2 %** de ces matchs ont un label positif.
— **X-test** : **52375** opportunités de covoiturage calculées par Karos entre le **08/07/2016** et le **11/08/2016**. **1,9 %** de ces matchs ont un label positif.

Test-score :

-	FN	FP	TP	F1 score	Rappel	Précision
Scoring	196.0	11538.0	792.0	0.118937	0.064234	0.801619
GOOD	784.0	5161.0	204.0	0.064222	0.038024	0.206478
GOLD	839.0	2248.0	149.0	0.088035	0.062161	0.150810

TABLE 2.1 – Résultats sur un jeu de données important.

Train-score :

-	FN	FP	TP	F1 score	Rappel	Précision
Scoring	0.0	39801.0	3951.0	0.165650	0.090304	1.000000
GOOD	3250.0	17819.0	701.0	0.062392	0.037851	0.177423
GOLD	3508.0	10765.0	443.0	0.058447	0.039525	0.112124

TABLE 2.2 – Résultats sur un jeu de données important.

2.2 Sur une journée

Nous avons également sélectionné une journée quelquonque et analyser les matchs proposés par Karos au cours de celle ci. La journée choisie fut celle du **12/08/2016**. 796 matchs furent analysés et uniquement 13 d'entre eux ont suscité un intérêt.

-	FN	FP	TP	F1 score	Rappel	Précision
Scoring	2.0	302.0	12.0	0.073171	0.038217	0.857143
GOOD	13.0	66.0	1.0	0.024691	0.014925	0.071429
GOLD	9.0	37.0	5.0	0.178571	0.119048	0.357143

TABLE 2.3 – Résultats sur une journée type.

Cinquième partie

Conclusions et perspectives

Chapitre 1

Difficultés rencontrées & perspectives

1.1 Difficultés rencontrées

Somme-toute, la démarche employée au cours de cette étude a été la plus rigoureuse possible. Nous avons alors essayé d'appliquer la plus grande rigueur à chaque fois qu'un choix a dû être fait afin de résoudre un problème spécifique.
La première difficulté à laquelle nous avons fait face relevait du problème industriel qui devait être résolu. En effet, cette difficulté s'est notamment ressentie lors de la définition du problème dans la mesure où l'on essayait de déviner ce qui "plairait" à un utilisateur, chose qui relève du subjetif et est donc dificilement mesurable. Il était alors difficile de faire des choix afin de recentrer le problème et de fixer les limites de notre étude. Pour cette raison, nous avons décidé de nous limiter à des critères mesurables et savions à l'avance que les résultats à obtenir ne seraient pas extraordianires. Il a permis d'obtenir les premiers résultats sur lesquels l'on s'est appuyé par la suite.
Un autre obstacle qu'il a fallu surmonter fut la réduction soudaine de la durée de mon stage. En effet, j'ai pris la décision au cours de ce stage de m'orienter vers une carrière dans l'entrepreneuriat. J'ai alors saisi une opportunité qui s'offrait à moi et ainsi réduits d'un tier la durée du stage. Je profite pour remercier à nouveau toute l'équipe Karos pour leur compréhénsion.
A ces difficultés, l'on peut ajouter les problèmes rencontrés couramment dans un projet d'analyse de données : manque de données, qualification de ces dernières, bruit, etc. Les méthodologies employées pour résoudre ces problèmes ont été présentées tout au long de ce rapport.

1.2 Perspectives

Le travail ainsi a permis d'obtenir des résultats très encourageants. Il permet en outre d'émettre des recommandations pour la suite du projet.
Une analyse experte des résultats de détection obtenus devrait être faite. Cette analyse permettrait alors d'évaluer la pertinence de ces résultats.

Chapitre 2

Apports du stage

Après avoir réalisé des précédentes expérience professionnelle dans un laboratoire de recherche et dans un grand groupe, ce stage de 6 mois représente ma première expérience longue en startup. Je m'y suis entièrement impliqué et, grâce à l'encadrement de mes responsables de stage, j'en ai tiré de nombreux enseignements tant sur le plan professionnel que sur le plan personnel.

2.1 Apports sur le plan pédagogique

Tout d'abord, cette expérience m'a permis de participer à toutes les phases d'un projet d'analyse de données : de la collecte de données à la validation des modèles statistiques. Ce fut pour moi l'occasion d'appliquer une palette de compétences variées que j'ai acquises au cours de ma formation (analyse statistique, traitement du signal, réseau et télécommunication, ...).
En parallèle, il m'a permis de développer des compétences en programmation et notamment de me familiariser avec le développement cloud (Google Cloud Compute). J'ai également découvert l'architecture du système d'information de l'entreprise et pour la première fois accédé à un système d'information géographique.
De plus, j'ai pu approfondir mes compétences en ce qui concerne l'analyse statistique des données en général et l'analyse de séries temporelles en particulier. Bien que l'apprentissage statistique et notamment les méthodes d'optimisation convexes restent les domaines de recherche qui m'intéressent les plus, le développement et la mise en production de ces solutions ont suscité en moi un intérêt particulier. Le développement d'algorithmes étant une problématique à la frontière de ces 2 domaines, je pense y continuer par la suite.
En outre, le fait de réaliser ce stage au sein d'une startup m'a permis de découvrir le fonctionnent d'un tel environnement de travail et de comprendre les différences entre ce monde et le monde de la recherche académique.

2.2 Apports sur le plan personnel

En plus des apports sur le plan professionnel évoqués, cette expérience a permis de révéler et de développer en moi des qualités telles que :

- L'autonomie et l'auto apprentissage : les techniques et outils utilisés tout au long du stage ont été pour la plupart le fruit de recherches et d'initiatives personnelles et d'entretiens avec mes encadrants et les différentes personnes impliquées dans le projet.
- Le sens de l'organisation : durant ce stage, certaines missions devaient être réalisées en parallèle. Il a donc fallu bien m'organiser afin de toutes les mener à terme. Aussi, compte-tenu du temps imparti, il m'a fallu travailler avec beaucoup d'efficacité. Mon responsable de stage a d'ailleurs, dans ce sens, été un moteur pour moi.
- La communication : cette qualité a été essentielle tout au long de ce projet car il m'imposait de collaborer avec des professionnels issus de secteurs d'activités différents . J'ai en outre dû m'imprégner du vocabulaire utilisé dans l'entreprise. De plus, la rédaction de ce rapport de stage fut également un exercice qui m'a permis de développer ma capacité à présenter de la meilleure façon possible mes idées.

« On peut toujours apprendre ce qu'on ne sait pas, non ce qu'on croit savoir »

cette citation de Gustave THIBON indique parfaitement l'état d'esprit dans lequel je me trouve à la fin de ce stage : j'ai conscience du travail immense que je dois fournir et des connaissances que je dois acquérir afin de pouvoir être opérationnel et performant dans le métier que je souhaite faire à l'avenir. C'est donc boosté et confiant que je termine cette expérience.

Annexes

.1 Description des variables

- `"pickup_dropoff_duration"` : Durée estimée du trajet.
- `"pickup_dropoff_distance"` : Distance estimée du trajet.
- `"pickup_distance"` : Durée estimée du trajet.
- `"driver_detour_duration"` : Durée du détour estimé que fera le conducteur.
- `"driver_detour_distance"` : Distance du détour estimé que fera le conducteur.
- `"delay_creation_pickup"` : temps en heure entre la création du match et l'heure de départ du covoiturage.
- `"number_matchs_couple"` : nombre de matchs (opportunités de covoiturage) précédemment proposés à ce couple d'utilisateurs.
- `"number_matchs_couple_changed"` : nombre de matchs précédemment précédemment proposés à ce couple d'utilisateurs sur lesquels il y a eu intéraction.
- `"number_matchs_couple_accepted"` : nombre de matchs précédemment proposés à ce couple d'utilisateurs ayant abouti à un covoiturage.
- `"weigthed_matchs_couple"` : somme pondérée (poids exponentiels décroissant dans le temps : exp(-TETA*deltaJours) de matchs précédemment proposés à ce couple. Si pour un match donné le couple de covoitureurs avait matché la veille et 3 jours auparavant alors `"weigthed_matchs_couple"` = exp(-TETA*1) + exp(-TETA*3).
- `"weigthed_matchs_couple_changed"` : somme pondérée des matchs précédemment proposés à ce couple sur lesquels il y a eu intéraction.
- `"weigthed_matchs_couple_accepted"` : somme pondérée des matchs précédemment proposés à ce couple ayant abouti à un covoiturage.
- `"weigthed_matchs_couple_driver_interact"` : somme pondérée des matchs précédemment proposés au conducteur sur lesquels il y a eu intéraction.
- `"weigthed_matchs_couple_passenger_interact"` : somme pondérée des matchs précédemment proposés au passager sur lesquels il y a eu intéraction.
- `"last_match_couple"` : vaut 0 si le dernier match du couple correspond à un covoiturage, 1 si le driver à interagi (driver du match dont on extrait les variables), 2 si le passager à interagi, 3 si aucun des 2, et 4 si le match n'existe pas.
- `"Weigthed_matchs_user_got_not"` : somme pondérée des matchs pour lesquels le user ne reçoit pas de réponse (je n'ai par erreur pas tenu compte du cas il il fait un

cancel proposal).

— `"Weigthed_matchs_user_gives_not"` : somme pondérée des matchs pour lesquels le user ne donne pas de réponse (je n'ai par erreur pas tenu compte du cas il il fait un cancel proposal).

Bibliographie

INDDIGO S.A.S. ADEME. Etude nationale sur le covoiturage de courte distance. Technical report, Agence de l'Environnement et de la Maitrise de l'Energie, 2015.

Shivani Agarwal. A study of the bipartite ranking problem in machine learning. 2005.

Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

Jason Brownlee. 8 tactics to combat imbalanced classes in your machine learning dataset, 2016. URL `http://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/`. Consulté le 19/08/2016.

Tianqi Chen and Tong He. xgboost : extreme gradient boosting. *R package version 0.4-2*, 2015.

Stéphan Clémençon. A lecture on statistical ranking, 2016.

Alberto Fernández, Salvador García, and Francisco Herrera. Addressing the classification with imbalanced data : open problems and new challenges on class distribution. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 1–10. Springer, 2011.

Junshui Ma and Simon Perkins. Time-series novelty detection using one-class support vector machines. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 3, pages 1741–1745. IEEE, 2003a.

Junshui Ma and Simon Perkins. Online novelty detection on temporal sequences. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–618. ACM, 2003b.

Julien Mairal. Machine learning with kernel methods, 2016.

Mixpanel. Jql : Api overview, 2016. URL `https://mixpanel.com/jql/`. Consulté le 19/08/2016.

Buyue Qian, Hongfei Li, Jun Wang, Xiang Wang, and Ian Davidson. Active learning to rank using pairwise supervision. In *Proc. 13th SIAM Int. Conf. Data Mining*, pages 297–305. SIAM, 2013.

Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. In *NIPS*, volume 12, pages 582–588, 1999.

David MJ Tax and Robert PW Duin. Data domain description using support vectors. In *ESANN*, volume 99, pages 251–256, 1999.

wikipedia. Mixpanel, 2016a. URL `https://en.wikipedia.org/wiki/Mixpanel`. Consulté le 19/08/2016.

wikipedia. Learning to rank, 2016b. URL `https://en.wikipedia.org/wiki/Learning_to_rank`. Consulté le 17/08/2016.

Xiaoxun Zhang, Xueying Wang, Honglei Guo, Zhili Guo, Xian Wu, and Zhong Su. Float-cascade learning for fast imbalanced web mining. In *Proceedings of the 17th international conference on World Wide Web*, pages 71–80. ACM, 2008.