Détection de vélos inutilisables grâce aux données ouvertes du système de vélos en libre service Citibike

Rémi Delassus, Romain Giot, Guy Melançon, Raphael Cherrier

Détection de vélos inutilisables grâce aux données ouvertes du système de vélos en libre service Citibike

R. Delassus[1,2] R. Giot[2] G. Melançon[2] R. Cherrier[1]

[1] QUCIT, park Newton, 213 cours Victor Hugo, 33130 Bègles, France
[2] Univ. Bordeaux, CNRS, LaBRI, UMR 5800, F-33400 Talence, France

remi.delassus@qucit.com, romain.giot,guy.melancon@u-bordeaux.fr, raphael.cherrier@qucit.com

Domaine principal de recherche : RFP
Papier soumis dans le cadre de la journée commune : OUI

Résumé

Le nombre de vélos partagés au sein d'un système de vélos en libre service ayant dépassé le million, il est nécessaire de détecter rapidement un vélo immobilisé en station à cause d'une panne. En effet, il n'est pas rentable en terme de nombre de trajets effectués et frustre les usagers qui s'attendent à trouver un vélo à cette station sans qu'ils ne sachent qu'il est en fait défectueux. Nous proposons une méthodologie d'extraction de caractéristiques et de détection d'anomalies sur une infrastructure distribuée de type « cloud computing » afin de détecter les vélos nécessitant une réparation. Nous montrons également pourquoi il est important pour nous de récupérer des données étiquetées remontant des applications mobiles à notre disposition.

Mots Clef

Détection d'anomalie, Données ouvertes, Masse de données, Architecture cloud, Vélo en libre service

Abstract

The number of bikes within a bikeshare system having reached more than a million in 2015, it seems necessary to detect in near realtime a broken bike rooted at a station. Indeed, it is not cost effective in terms of number of trips, and it frustrates users who were expecting to find a bike at that station without knowing that it is actually defective. We thus propose a methodology for feature extraction and anomaly detection on a distributed cloud infrastructure in order to detect bicycles requiring a repair. We also show why it is important for us to retrieve labeled data from mobile applications at our disposal.

Keywords

Anomaly detection, Open data, Big data, Cloud architecture, Bikeshare

1 Introduction

En 2015 on estime à 1 million le nombre de vélos partagés dans le monde ; ils sont utilisés de manière bien plus intensive qu'un vélo personnel (10,8 trajets par jour à Barcelone ; 8,3 à Lyon ; 8,3 à New York [3]), et connaissent une usure accrue. Comme cette usure augmente la probabilité de pannes, le nombre de vélos susceptibles d'être en panne est donc important, tout comme les actes de maintenance visant à remettre un vélo en bon état. Il est ainsi de tradition pour les utilisateurs de retourner la selle d'un vélo défectueux afin d'indiquer aux autres usagers son mauvais état. La qualité du service est améliorée grâce à cet investissement personnel : moins de temps est perdu à emprunter un vélo non fonctionnel, en sachant immédiatement qu'il est préférable d'en prendre un autre.
Bien que les équipes de maintenance effectuent des rondes afin de vérifier l'état de chaque vélo, des systèmes d'alertes ont été mis en place : un bouton sur chaque borne qui permet d'alerter sur le fait que le vélo garé là est défectueux (comme c'est le cas dans le système Citibike à New York [18]) ; une adresse mail ou un numéro de téléphone (par exemple avec le Vélib de Paris [7]) servant à recueillir les doléances ; ou encore un outil logiciel, qui va demander aux utilisateurs qui reposent leur vélo très peu de temps après l'avoir emprunté s'il est en bon état (nous pouvons voir ce système en place à Bordeaux, avec le VCub). Les vélos défectueux sont réparés sur place lorsque c'est possible (pneu dégonflé par exemple), ou emmenés par la maintenance pour être réparés dans un atelier spécialisé quand la panne le nécessite.
Nous avons montré qu'un système de vélos en libre service voit son nombre de trajets tripler en doublant son nombre de stations [15]. Ce nombre de trajets augmentant plus vite que linéairement, le nombre de pannes et le nombre d'utilisateurs souhaitant effectuer un trajet croit aussi rapidement. Les moyens mis en œuvre pour les supporter doivent d'autant plus être améliorés. On peut également penser aux usagers qui utilisent des applications mobiles pour préparer leur trajet et sont informés qu'un vélo est disponible en station sans savoir si ce dernier est fonctionnel ou non. Leur nombre augmentant de la même manière, la frustration générée est d'autant plus grande.
Afin de surmonter ce problème, nous nous intéressons à la

détection de ces pannes (ou « anomalies ») afin d'améliorer la qualité des systèmes de vélo en libre service : (i) d'une part en prévenant plus tôt les opérateurs, qui peuvent ainsi cibler directement les vélos défectueux et être plus réactifs ; (ii) d'autre part en prenant en compte les pannes au sein d'une application mobile de préparation de trajet afin de fournir aux utilisateurs le nombre de vélos réellement fonctionnels en station (et pas le nombre de vélos présents).

Une solution pour résoudre le problème est d'utiliser un modèle d'apprentissage automatique classant à partir de données étiquetées (vélo normal, vélo avec anomalies) des vélos au cours du temps. Dans le scénario idéal, il faut obtenir un partenariat avec un opérateur qui fournit une vérité terrain (et intégrer une étape de collecte de données dans le protocole de l'équipe de maintenance – ce qui peut être difficile à mettre en place), telles que des données qui indiquent pour chaque vélo les dates de début et de fin de maintenance ainsi que la raison pour laquelle elle était nécessaire.

Si aucun partenariat n'est établi ou si le partenaire n'est pas en mesure de fournir de telles données, la difficulté repose sur l'évaluation des modèles face au peu d'informations disponibles : bien que les données d'occupation des stations soient la plupart du temps disponibles en libre accès sur les sites des opérateurs, il est rarement possible d'obtenir des données sur les trajets effectués (New York, San Francisco) ; de plus, aucun opérateur ne fournit librement de données précises sur ses opérations de maintenance.

Dans ce travail, nous nous intéressons à ce second cas avec les données ouvertes fournies par l'opérateur Motivate à New York sur une période de deux ans et demi : (i) les trajets de chacun de ses vélos [13] ; (ii) des rapports mensuels sur les activités de l'opérateur [12] indiquant entre autre le nombre de réparations effectuées (ce qui nous permet d'approximer la pertinence de nos modèles). Nous proposons un modèle de détection d'anomalies hors ligne qui détecte les vélos ayant un comportement anormal, signe d'une possible panne.

L'originalité de notre étude réside dans la considération d'un vélo en tant qu'entité ; plutôt que d'être centré sur l'état du système ou des stations., Par exemple si un vélo défectueux reste immobile dans une station durant un grand laps de temps, il sera plus pertinent de s'intéresser au fait que le vélo soit immobile qu'au fait que la station ne se vide pas. À notre connaissance, il n'existe pas dans la littérature de travaux où l'on cherche à détecter les vélos défectueux de façon automatique. À noter qu'étant donné que la quantité de données à manipuler est trop importante pour pouvoir utiliser une machine standard, les différentes expériences ont été réalisées en utilisant une grappe de machines de type « Big Data » [9] (c'est-à-dire, une grappe de machines standards plutôt qu'une grappe orientée calcul parallèle) hébergeant une plateforme Hadoop [19] avec Spark [20].

La suite de l'article est organisée de la façon suivante. La section 2 présente l'expérimentation que nous avons menée (analyse des données, extraction des caractéristiques pertinentes, création du détecteur) ; la section 3 présente et discute les résultats obtenus, tandis que la section 5 conclut ce papier et présente des pistes d'améliorations pour les travaux futurs.

2 Expérimentation

Cette section présente la méthodologie employée pour obtenir le détecteur d'anomalies à partir du jeu de données ouvert.

2.1 Jeu de données

Les données utilisées correspondent aux trajets effectués par les utilisateurs de Citibike, le système de vélo en libre service de New York. Elles sont récupérées sur le site de l'open data de Citibike [13], sous la forme de fichiers CSV mensuels. Nous avons utilisé les données de Juillet 2013 à Novembre 2015 ce qui représente plus de 23 millions de trajets, effectués en 894 jours par 9 423 vélos distincts et nécessite 4 Go de stockage.

Chaque trajet est décrit par un certain nombre de caractéristiques hétérogènes : des données temporelles (heure de départ, heure d'arrivée, durée du trajet), des données géographiques (coordonnées de la station de départ, coordonnées de la station d'arrivée), l'identifiant du vélo ayant effectué le trajet, le nom des stations, pour des vélos empruntés par un abonné : son année de naissance et son sexe.

2.2 Extraction de caractéristiques

Dans le but de pouvoir détecter des anomalies en temps réel par la suite, nos méthodes de détection se feront grâce à des algorithmes d'apprentissage supervisé. En effet, une fois que la méthode de détection est apprise par l'algorithme, il suffira, pour quelqu'un disposant en temps réel des dernières données, d'extraire les caractéristiques appropriées d'un nouvel échantillon pour déterminer instantanément s'il est normal ou non. Nous effectuons une étape de transformation sur les données pour calculer de nouvelles caractéristiques ayant un sens pour notre méthode d'apprentissage. Dans ce but, nous nous intéressons, pour chaque vélo, à des fenêtres de 7 jours, glissant de jour en jour. Elles permettent de déterminer si un vélo est dans un état anormal en fonction des informations recueillies les 7 jours précédents. Environ 4 800 000 fenêtres sont considérées (si pendant les 7 jours d'une fenêtre un vélo ne circule pas, alors la fenêtre correspondante n'est pas créée), chacune d'entre elles contenant les caractéristiques suivantes : nombre de trajets effectués, nombre de stations de départ distinctes, nombre de stations d'arrivées distinctes, nombre de trajets étant des boucles (même station de départ et d'arrivée), distance moyenne parcourue, distance de trajet maximale, distance de trajet minimale, durée moyenne des trajets, durée de trajet maximale, durée de trajet minimale. Ces caractéristiques ont été choisies car elles permettent, en l'absence de données de référence, de vérifier que les choix faits par la méthode de reconnaissance sont sensés. Par exemple, si une partie significative des vélos considérés comme défectueux effectuent

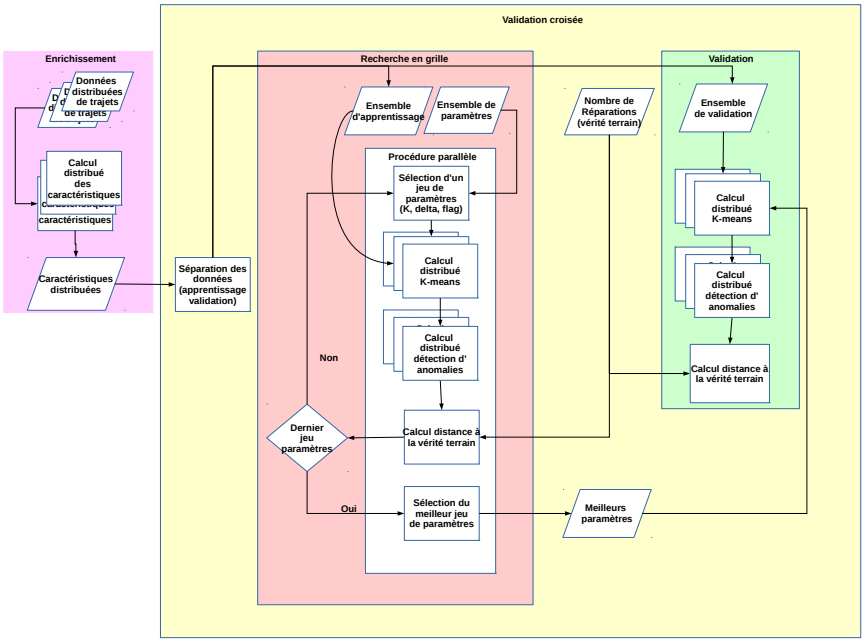

FIGURE 1 – Résumé de la procédure employée pour apprendre et utiliser le détecteur d'anomalies

régulièrement de longs et nombreux trajets, on peut facilement conclure que l'algorithme n'est pas bon. Aussi, elles permettent d'agréger naturellement les données résultantes du découpage en fenêtres.

2.3 Détection d'anomalies

Ainsi, chaque observation, identifiée par un numéro de vélo et le jour suivant la fenêtre, est un vecteur dans un espace multidimensionnel où chaque dimension de cet espace est une des caractéristiques extraites (ici l'espace est de dimension $n = 10$). La détection d'anomalies à partir de ces caractéristiques centrées et réduites [1] se fait en deux étapes. Premièrement, un partitionnement à l'aide de l'algorithme des K-moyennes [11] est appliqué afin de générer K partitions qui représentent K comportements différents pour les vélos. L'espace multidimensionnel contenant les observations est alors séparé en K parties indicées par $k \in [1; K]$. Toute observation appartient à une partition k de taille M^k et est notée $O_i^k, i \in [1; M^k]$. Chaque partition est dotée d'un représentant, R^k. Ce dernier est situé dans l'espace en prenant pour l'ensemble de ses caractéristiques $\{R_j^k\}, j \in [1; n]$ la moyenne des observations de la partition ; $R_j^k = \frac{\sum_{i=1}^{M} O_{ij}^k}{M}$. Cette séparation des comportements normaux en K partitions nous semble nécessaire car rien ne prouve qu'il n'y ait qu'une unique classe de comportements. Par exemple à Barcelone, les vélos ont tendance à descendre les collines le matin aux heures d'embauche, quand les zones d'habitations situées en hauteurs se vident, mais ne remontent pas le soir car la pente est trop raide [4] : en fonction de sa géolocalisation, un vélo n'aura pas le même comportement. De plus, sans ce partitionnement, un grand groupe de comportements similaires pourrait être étiqueté comme anormal, alors qu'ils représentent seulement un usage différent du système.

Ensuite, chaque partition correspondant à un type de comportement est filtrée afin de détecter des observations trop différentes des autres (les anomalies). Ce sont celles qui sont les plus éloignées du représentant de la partition (le prototype), donc des comportements éloignés du comportement typique, qui sont filtrées. Un deuxième paramètre δ, à déterminer également, sert à régler la distance au représentant à partir de laquelle une observation est considérée comme anormale, et donc le vélo qu'elle représente comme défectueux sur cette période de temps. Soit la distance entre une observation et le prototype :

$$dist(O_i^k, R^k) = \sqrt{\sum_{j=1}^{n} (O_{ij}^k - R_{j^k})^2} \qquad (1)$$

et notre filtre :

$$anom(O_i) = \begin{cases} 1 & \text{si} \quad dist(O_i^k, R^k) > \delta * \sigma^k \\ 0 & \text{sinon} \end{cases} \qquad (2)$$

où $anom$ est la fonction qui vaut 1 si l'observation O_i est anormale (vélo défectueux) et 0 si l'observation O_i est normale (vélo en bon état), σ^k l'écart type des distances au prototype.

2.4 Protocole expérimental

Deux paramètres (K,δ) doivent donc être fixés.
Ils sont optimisés en effectuant une recherche en grille où la combinaison de différentes valeurs pour chaque paramètre

1. Les ordres de grandeur des différentes caractéristiques n'étant pas comparables, il est nécessaire d'effectuer ce traitement afin de ne pas donner plus d'importance à une caractéristique qu'aux autres.

est testée. La combinaison minimisant l'erreur (définie plus loin) est sélectionnée. Les paramètres testés pour K sont (2, 3, 4, 5) et pour δ sont (1, 1.25, 1.5, 1.75, 2, 2.25, 2.5, 2.75, 3). Nous ne cherchons pas de K au delà de 5 car ce paramètre permet de distinguer différents cas d'usages du vélo
autoriser un grand K permettrait de décrire un grand nombre de comportements, ceux-ci ne seraient probablement pas représentatifs et le modèle serait en état de sur-apprentissage. L'ensemble des valeurs possible pour le paramètre δ est choisi en fonction de la distribution gaussienne des caractéristiques extraites des données. En effet, δ étant un facteur multiplicatif de l'écart type, on sait qu'au delà de $\delta = 3$ la majorité des observations seraient classées comme normales, et en dessous de $\delta = 1$ la majorité des observations seraient classées comme anormales.

Afin de ne pas sur-apprendre notre modèle, nous procédons à une étape de validation croisée. Une séparation est faite au mois d'août 2014 : les données précédant le mois d'aout 2014 servent d'ensemble d'apprentissage tandis que les données suivantes servent à vérifier que le modèle appris fonctionne aussi bien sur des données inconnues.

Bien que nous ne disposons pas de vérité terrain sur le jeu de données étudié, à partir des rapports mensuels [12] nous disposons du nombre de vélos ayant été réparés par l'opérateur pour chaque mois. Cette information peut être utilisée pour estimer l'efficacité de notre modèle de prédiction [2]. La formule 2.4 donne la méthode d'optimisation des paramètres :

$$\arg\min_{K,\delta} \sum_{m \in mois} (Anomalies_{K,\delta}(m) - Reparation(m))^2 \quad (2)$$

avec $Anomalies_{K,\delta}(m)$ le nombre d'anomalies (un vélo peut être considéré comme anormal plusieurs fois) en utilisant les paramètres et $Reparation$ le nombre de réparations effectuées par l'opérateur au mois m.

La figure 1 résume l'ensemble de l'approche employée.

2.5 Implémentation

L'expérimentation est développée grâce à Spark 2.6.0. Elle est exécutée sur une grappe de 15 machines de 50 Go de RAM, avec 48 threads. Les données sont stockées dans HDFS. La configuration comprend 50 exécuteurs avec 5 cœurs par exécuteur (il y a donc plusieurs exécuteurs par machine) et la possibilité d'utiliser 5Go de RAM par exécuteur. L'implémentation est également exécutée sur une machine standard (8 threads, 32Go de RAM) afin de vérifier la pertinence d'une architecture distribuée.

3 Résultats

3.1 Résultats

L'implémentation sur une architecture de type cloud est justifiée par la taille des données. En effet, sur une machine standard les temps de calculs sont longs : l'extraction de caractéristiques requiert 25 minutes tandis qu'il faut 5 heures pour obtenir les paramètres optimaux pour la détection d'anomalies. Sur la grappe, les données extraites ont été générées en 2min30s (temps incluant la lecture et l'écriture sur disque) tandis que la détection d'anomalies (temps incluant la lecture et l'écriture sur disque, l'apprentissage avec recherche en grille et la validation) est effectuée en 17min. Le meilleur modèle correspond aux paramètres (K=2, δ=2.5).
À noter que dans ce travail, nous n'avons pas encore pu répondre exactement au problème posé, qui est la détection de dysfonctionnement des vélos en libre service, en raison de l'absence de données étiquetées précises. Nous avons travaillé sur un problème légèrement différent qui est la détection des réparations des vélos, ce qui nous a permis de valider notre approche. Cependant en disposant de données de vélos étiquetées le même processus pourrait être utilisé (en modifiant la fonction d'optimisation).
Pour chaque fenêtre de temps, un vélo peut se trouver dans trois états différents : il a été classé par le détecteur comme étant « fonctionnel », « anormal », ou il n'a pas été classé car il n'y a pas d'observation pour ce vélo durant cette semaine. Il est alors dans un état « non utilisé ». Les fenêtres étant, pour chaque vélo, ordonnées dans le temps de manière continue (une par jour), on peut extraire pour chacune un couple représentant l'état actuel et l'état suivant. On obtient alors 9 transitions possibles (9 couples d'états possibles) dont on représente la répartition en figure 3. On voit que le changement d'état dominant est l'état « anormal » vers l'état « fonctionnel », signe d'une réparation. On pourrait s'attendre à voir une proportion plus importante de transitions « fonctionnel » vers « anormal », cela vient sûrement du fait que l'on optimise la distance entre le nombre d'états anormaux mensuels et le nombre de réparations déclarées, et non pas le nombre de vélos en état anormal. En effet, pour parvenir à minimiser cette distance, le modèle n'a pas plus intérêt à déclarer de nouveaux vélos comme anormaux (ce qui augmenterait la proportion de transition fonctionnel vers anormal) que de garder des vélos en état anormal. Les résultats obtenus semblent donc cohérents.
La figure 2 présente le nombre de vélos en état d'anomalie dans le mois à la fois sur ce que nous détectons et ce qui est fourni par l'opérateur (en rouge). Deux courbes basées sur nos résultats sont présentées : en bleu, le nombre total d'anomalies (ce que nous optimisons), en vert, le nombre de vélos en état d'anomalies (ce qui semble plus proche de la réalité). On voit que la zone de validation (Aout 2014-Novembre

2. à noter qu'avec cette information, nous pouvons vérifier que la quantité de vélos détectés comme étant anormaux est cohérente, mais pas que ceux ci soient réellement ceux présentant des anomalies.

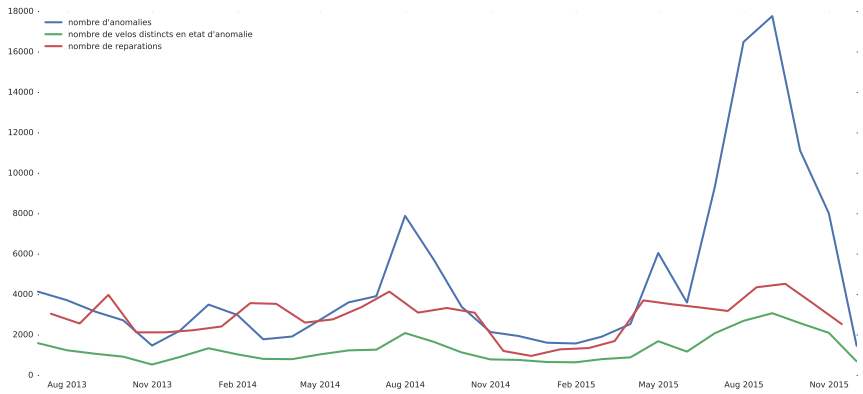

FIGURE 2 – On observe en vert le nombre réparations déclarées. Le comptage brut du nombre d'anomalies (fonction utilisée pour l'optimisation) est en bleu. Le pic correspond à des vélos qui sont restés longtemps en état d'anomalie. En rouge le nombre de vélos en situation d'anomalies. La courbe se situe sous la courbe verte car le modèle n'est pas entraîné à minimiser la distance entre ces deux courbes, mais la distance entre le nombre d'anomalies détectées et le nombre de vélos réparés dans le mois. Les valeurs avant Août 2014 correspondent au performances sur le jeu d'apprentissage.

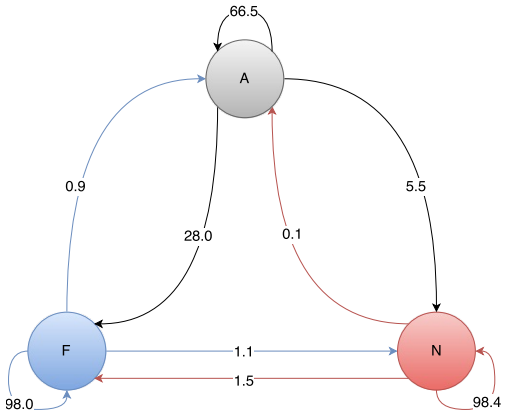

FIGURE 3 – Chaîne de Markov de la classification obtenue (A=anomale, F=fonctionnel, N=non utilisé).

2015) ne semble pas souffrir de sur-apprentissage, ce qui montre également que l'approche est valide. Cependant, le nombre de vélos en état anormal est sous-évalué.

3.2 Discussion

Notre modèle sert à séparer les vélos en trois catégories : les vélos normaux, les vélos anormaux, le vélos non utilisés. L'optimisation est faite de manière à réduire la distance au nombre de vélos emmenés en magasin pour y être réparés. Ce nombre ne correspond pas nécessairement au nombre de vélos défectueux, et de ce fait notre modèle est plus à même de détecter des vélos réparés que des vélos défectueux. Cette limitation peut naturellement être améliorée en disposant d'un étiquetage des anomalies de meilleur qualité.

Étant donné le peu de données étiquetées en notre possession (30 valeurs qui correspondent aux 30 mois de l'étude), la validation croisée pourrait être faite en utilisant une méthode effectuant plus de choix (Leave One Out plutôt que que validation avec deux ensembles) et obtenir un intervalle de confiance sur les performances obtenues.

On notera aussi que par souci de performance, l'optimisation ne se fait que sur deux paramètres (K et δ) alors qu'il n'y a pas de raison que toutes les partitions partagent un même δ optimal. Il reste donc une marge d'amélioration qui pourra être faite en cherchant un δ^k optimal par partition k.

Afin d'évaluer les performances de notre modèle, nous nous sommes basés sur les rapports mensuels de Motivate, dans lesquels figurent le nombre de vélos emmenés en magasin pour réparation. Cependant, il n'est pas indiqué la nature des réparations, nous ne savons donc pas quelle proportion de ces réparations étaient assez importantes pour que le vélo puisse être considéré comme défectueux. En effet, il y a plus de 3 000 vélos réparés chaque mois, pour 6 000 vélos en circulation, ce qui laisse penser que tous les vélos emmenés au magasin ne sont pas dans un état d'usure sévère.

Comme nous l'avons mentionné, la méthodologie utilisée pourrait donner de meilleurs résultats avec une vérité terrain (pannes et nature des pannes), qui pourrait être obtenue soit grâce à une collaboration avec un opérateur, soit par une collecte. Cette collecte est envisageable à travers l'application Bikepredict, développée par Qucit [14]. C'est par les usagers du vélo en libre service que cette application mobile est utilisée car elle permet de savoir jusqu'à 45 minutes en avance si des places ou des vélos seront disponibles en station. Par la suite, elle récupère des données d'accélérométrie afin de connaître le mode de transport de la personne. Ces données pourraient éventuellement permettre de détecter un vélo roulant de manière suspecte. Aussi, il sera bientôt possible à travers cette application de signaler un vélo défectueux et d'indiquer le type de panne. Alors l'information obtenue sera complète et pourra servir à entraîner

des modèles. On notera également qu'une fois les modèles entrainés à partir des données récoltées, il est tout à fait envisageable pour un opérateur ayant accès à tout moment aux dernières mise à jour des données de l'utiliser en temps réel sans avoir à le ré-entrainer. Enfin, on remarque que l'optimisation en fonction du nombre de réparations déclarées permet seulement à notre modèle de reconnaître un vélo qui a été emporté par la maintenance, ce qui est différent d'un vélo défectueux ayant besoin d'être emporté par la maintenance. Cela changera lorsque nous pourrons optimiser en fonction du nombre de vélos déclarés défectueux par les utilisateurs.

4 Travaux existants

À notre connaissance, il existe peu de travaux dans la littérature sur la détection de vélos nécessitant une réparation dans les systèmes de vélos libre service. Dans [8], les auteurs utilisent un jeu de données couvrant une période de deux mois, du système Citibike, associé à une simulation de prêts pour prédire qu'un vélo est en panne ainsi que le nombre de vélos en panne dans une station à l'aide d'un modèle bayésien (les prédictions ne sont donc pas basées sur des évènements réels). Cependant, nous pouvons trouver des travaux où les données d'utilisation d'un tel système sont utilisées pour effectuer (i) de la prédiction d'usage de systèmes de vélos libre service [6, 17, 10], (ii) du réordonnancement de vélos dans les stations [16, 1] (iii) de la détection d'anomalies à l'extérieur du système [5, 2].

5 Conclusion

Il est nécessaire de détecter rapidement un vélo immobilisé en station à cause d'une panne. En effet, il n'est pas rentable en terme de nombre de trajets effectués, et frustre les usagers qui s'attendent à trouver un vélo à cette station sans qu'ils ne sachent qu'il est en fait défectueux. Nous proposons une méthodologie d'extraction de données et de détection d'anomalies sur une infrastructure distribuée de type « cloud computing » afin de détecter les vélos nécessitant une réparation.
Les anomalies sont détectées en filtrant les individus trop éloignés du représentant de leur partition calculée à l'aide de k-moyenne. Les paramètres du détecteur sont calculés à l'aide d'une validation croisée et d'une recherche en grille sur un jeu de données réel de 30 mois.
Les résultats obtenus semblent cohérents et réalistes. Cependant, nous avons vu que le manque de données étiquetées de façon précise est un gros handicap. C'est pourquoi nos travaux futurs se porteront sur la récolte de données à travers une application mobile. Ces nouvelles données permettraient d'ajuster les paramètres du modèle afin d'approcher plus précisément la réalité.

Remerciements

Nous tenons à remercier Citibike qui met à disposition publiquement les données des trajets effectués à New York et les rapports mensuels de maintenance, ainsi que l'ANRT pour le financement de la convention CIFRE 2014/1303.

Références

[1] Philipp AESCHBACH et al. « Balancing Bike Sharing Systems through Customer Cooperation–A Case Study on London's Barclays Cycle Hire ». In : *IEEE Conference on Decision and Control*. 2015.

[2] Roel BERTENS, Jilles VREEKEN et Arno SIEBES. « Beauty and Brains : Detecting Anomalous Pattern Co-Occurrences ». In : *arXiv preprint arXiv :1512.07048* (2015).

[3] BIKESHARINGNAPOLI. *Bike share boom : 7 cities doing it right*. `http : / / www . bikesharingnapoli . it / it / best - practices/`. En ligne ; accédé le 18/02/2016. 2013.

[4] Gabriel Martins DIAS, Boris BELLALTA et Simon OECHSNER. « Predicting occupancy trends in Barcelona's bicycle service stations using open data ». In : *SAI Intelligent Systems Conference (IntelliSys), 2015*. 2015, p. 439–445.

[5] Hadi FANAEE-T et Joao GAMA. « Event labeling combining ensemble detectors and background knowledge ». In : *Progress in Artificial Intelligence* 2.2-3 (2014), p. 113–127.

[6] Nicolas GAST et al. « Probabilistic Forecasts of Bike-Sharing Systems for Journey Planning ». In : *The 24th ACM International Conference on Information and Knowledge Management (CIKM 2015)*. 2015.

[7] *How to be a good citi bike citizen*. `http://blog.velib.paris.fr/blog/2012/04/03/le-serveur-vocal-a-votre-ecoute/`. En ligne ; accédé le 18/02/2016. 2012.

[8] Mor KASPI, Tal RAVIV et Michal TZUR. « Detection of unusable bicycles in bike-sharing systems ». In : *Omega* (2015).

[9] Pirmin LEMBERGER et al. *Big Data et Machine Learning – Manuel du data scientist*. Dunod, 2015.

[10] Yexin LI et al. « Traffic Prediction in a Bike-Sharing System ». In : *SIGSPATIAL'15*. 2015.

[11] J. MACQUEEN. « Some methods for classification and analysis of multivariate observations ». In : *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1 : Statistics*. 1967, p. 281–297.

[12] MOTIVATE. *NYCBS Monthly Operating Reports*. `https://www.citibikenyc.com/system-data/operating-reports`. En ligne ; accédé le 15/02/2016. 2016.

[13] MOTIVATE. *System Data*. `https : / / www . citibikenyc.com/system-data`. En ligne ; accédé le 15/02/2016. 2016.

[14] QUCIT. *bikepredict*. https://www.qucit.com/bikepredict/. En ligne ; accédé le 18/02/2016. 2015.

[15] QUCIT. *Vélos en libre-service : l'importance de la taille du réseau*. http://www.qucit.com/2015/10/15/velos-en-libre-service-limportance-de-la-taille-du-reseau/. En ligne ; accédé le 18/02/2016. 2015.

[16] Tal RAVIV, Michal TZUR et Iris A FORMA. « Static repositioning in a bike-sharing system : models and solution approaches ». In : *EURO Journal on Transportation and Logistics* 2.3 (2013), p. 187–229.

[17] Raphaël Cherrier ROMAIN GIOT. « Predicting Bike-share System Usage Up to One Day Ahead ». In : *IEEE Symposium Series in Computational Intelligence 2014 (SSCI 2014). Workshop on Computational Intelligence in Vehicles and Transportation Systems (CIVTS 2014)*. 2014, p. 1–8.

[18] SUSI. *How to be a good citi bike citizen*. http://velojoy.com/2013/07/02/turned-bike-share-saddle/. En ligne ; accédé le 18/02/2016. 2013.

[19] Tom WHITE. *Hadoop : The definitive guide*. " O'Reilly Media, Inc.", 2012.

[20] Matei ZAHARIA et al. « Spark : Cluster Computing with Working Sets. » In : *HotCloud* 10 (2010), p. 10–10.