USING DATA MINING TO IMPROVE BIKE TRAVEL

Weiqi Jiang and Yujia Guo

University of Waterloo, Waterloo ON N2L 3G1, Canada
w83jiang@uwaterloo.ca y337guo@uwaterloo.ca

Abstract. Bike-sharing system is a new type of public transport system which grows rapidly in recent years. It offers users total freedom to decide their destinations and routes, leading to unbalance station usage. At the same time, the flexibility provides a perfect microscopic perspective to understand users' trip patterns. In this paper, we consider to achieve two main goals, one is trip prediction, once user has decided origin station, we want to know the most likely destination. The other one is purposing a trip advisor to improve biased station usage. For trip prediction part, we first mapped the problem to binary classification problem and restricted possible prediction set. Then chose a suitable model and conducted parameter adjusting process. Established training and testing set to train model and evaluate performance. For trip advisor part, we first defined activeness for every station and then used distance as metric to filter shift candidates. We selected pair of stations with minimum activeness as the recommendation. At last, we conducted simulation to evaluate the improvement degree.

Keywords: Trip prediction, Bike-sharing system, XGBoost model.

1 INTRODUCTION

Bike-sharing system(BSS) developed significantly fast during recent years, especially in Asian area. The system offers users short vehicle journeys for free or with small charges. People can rent bikes from stations nearby and return the bike to any station available. Compared with other public transport systems, BSS has much less carbon footprint and higher flexibility. For commuters or students who lives in busy area where often suffer from traffic congestion, BSS seems to be a better choice than bus or taxi due to its high availability and flexibility. As a result, more and more cities start to develop their own bike-sharing system, e.g., San Francisco (Bay Area Bike Share), Washington, D.C.(Capital Bikeshare), Montreal (BIXI Montreal), New York (Citi Bike).

There are two types of users in bike-sharing system, 'subscribed users' and 'unsubscribed users'. 'Unsubscribed users' usually prefer temporary usage, but the average charges are slightly higher for them. For 'subscribed users', they are more likely to take regular travel. Based on this, the usage data of bike-sharing system can reflect individual's travel pattern to a large extent. If operators of bike-sharing systems could

extract travel patterns from data, this would give them suggestions to improve their service qualities or even offer more personal services.

Unlike other fixed-route public transport systems, bike-sharing system provides its users freedom to decide destinations, which can meet the needs of most categories of users. However, this high flexibility triggers problems for management. Since the system won't restrict destinations for users, it cannot control the flow of bikes, after a period of time, the distribution of available bikes will be unbalanced and unreasonable. Most bikes will concentrate at those overused stations and there will lake of locks to secure such many bikes, at the same time, for those underused stations, people cannot find enough bikes to meet their demands. To support such a claim, we analyzed a real-world bike-sharing system data. The data contains usage for one month which is November 2017. We counted the number of bikes borrowed from/ returned to each station respectively, the result can be seen in fig.1. In the figure, bigger and redder dot meant larger bike flow for one station, it is easy to spot that usage between each station were unbalanced, it was the summary for just one month, we had confidence to predict that after several months even years without any action to rebalance the bike flow, the situation could be much worse than it was in fig.1. Thus, after a period of time, the operators have to spend human source and financial source on rebalancing the bike distribution in order to keep the system running efficiently.

In this paper, we wanted to achieve two main functions. One is trip prediction, to be more specific, we proposed to predict the most likely destinations, once users had decided their origin stations. This problem is challenging because users travel patterns depend on various factors. The lack of personal information in the training data made precise prediction become even harder, in section 2.1, we will describe more details about the data we used.To address the trip prediction problem, we preprocessed the data we had, extracted some useful features, and established predict model. Took advantage of cross validation method to finish parameter adjusting process. Then used training set to fit model and test set to evaluate model's performance.

The other main goal we wanted to achieve is proposing a trip advisor. This advisor would recommend other pairs of origin stations and end stations which could help balancing the bike flow and station usage. Of course, the decision-making power was still in hands of users. To make this function achieved, we first needed to define 'activeness' for each station. Then we conducted simulation to quantify the improvement.

In the reference [1], Zhang spot the imbalanced problem in the bike usage data. Motivated by this problem, they proposed a model to predict the destination and correspond duration. Once the operators of BSS have a big picture about bike flow, they could rebalance the bike distribution efficiently. They first proposed to transform the destination prediction problem to be a binary classification problem and took advantage of MART (Multiple Addictive Regression Tree) to solve it. For the duration prediction problem, they applied Lasso regression model as the base regression model. In training section, they randomly generate none-existed trip as the negative samples. Then, Accuracy, Precision, Recall and F1-score were applied to evaluate the destination prediction performance. MAE(Mean Absolute Error) and R^2 to evaluate the duration prediction performance. In all, their method was well-organized and had got a reasonable result.

In reference[2], Hu, Yang and Shu proposed a trip advisor to recommend the optimal pair of original and return stations. First, they presented a probabilistic forecast method to deal with the demand prediction problem which aims to make sure users could find bikes and unused lockers. The forecast method could be divided into two main parts, one was coarse-gained prediction using random forest model, the other was fine-gained prediction using Monte Carlo method. Then they introduced a concept of activeness of station to quantify the active level of stations. In the experiment section, they applied the probabilistic forecast method as a filter to set the candidates for the next step. The next step is to choose the start and end station pair from the candidates which got a maximum activeness difference. The criteria which were used to evaluate the performance of probabilistic forecast were Precision, Recall and F-measure. Average and standard deviation were applied to evaluate the bike usage distribution. Thorough simulation, the result showed that the percentage of the most used bike decreased by 33.6% and usage time decreased by 28.6%.

In reference[3], authors analyzed passenger pattern in public transit through mining smart card and socioeconomic data. They apply a generative model-based clustering approach to cluster all the customers. For each passenger, they established a 'weekly profile' according to their travel history. Then, used information retrieval methods to achieve clustering. Having something in common with reference[3], in order to improve location prediction accuracy, scholars of reference[4] first clustered all users. For each cluster, it owned its unique travel patterns. After that, they applied Markov model to predict the next location given previously visited locations.

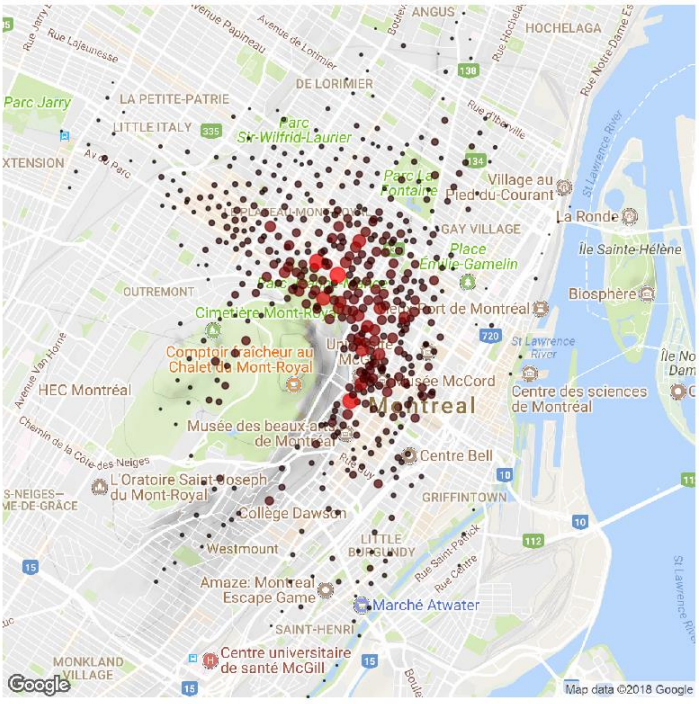

Fig. 1. Station usage of November 2017 at Montreal city.

2 DATASETS DESCRIPTION

2.1 Dataset details

We will describe more details of a real-world data from a bike-sharing system in this section. The dataset we used came from BIXI Montreal organization which owns 6200 bikes and more than 540 stations spread out across the areas of Montreal city. The datasets are public and can be download from its official website, for now, it released data from year 2014 to 2017. We mainly used trip data from 2016 to 2017 as well as data about station information. In total, it has about 4.7 million of trip records. The records have 6 dimensions, are start date, start station, end date, end station, duration and label to indicate membership respectively. The station datasets contain station code, station name and correspond latitude and longitude for each station.

2.2 Feature extraction

Before we utilized those datasets to conduct further process, we had to conduct data cleaning and feature extraction cause the dimensionality of origin datasets was rather limited. First, we used bool function to detect whether there existed missing values. Then, based on our daily experience, we assumed that if a trip record with same origin and destination and the duration was less than 120 seconds at the same time, this record was invalid. Because, this situation meant users suddenly want to quit travel or something went wrong with the bikes in most cases. It contributed little to the trip pattern, and we deleted those records from the datasets.

Next part we would introduce here was feature extraction. Based on the brief description about features of datasets above, due to lack of more personal information, we could only extract one feature ‘membership’ about user, we used 1 to represent subscribed user and 0 for unsubscribed user. Besides type of user, start time and date also have strong relationship with the usage, it is nature that there are more usages during daytime than night time and more usages in summer than winter. Normal hours and peak hours also affect the density of the usage. So based on those assumption, we extracted 3 features about date and time. ‘month of travel’: the dataset we downloaded only contains records from April to November, it was easy to understand, because it would be too cold to ride bikes plus the road condition would become unsafe for ride during winter, we used 4 to indicate April, 5 to indicate May and so forth. ‘start hour’: whether it is day time or night time, rush hour or normal hour certainly influence the usage. To be more specific, we used number 0 to 23 to represent the 24 hours in a day. ‘weekdays’: during weekdays, commuters and students are more likely to have regular trip plan, in weekends, the number of tourists or one-time users will increase. We set 0 for Sunday, 1 for Monday and so forth. At last but not least, the location of start station and end station, the distance between them all have their influence on the trip patterns, in total we extracted 7 features from station location information. That is, ‘start station code’, ‘end station code’, correspond ‘start station latitude’, ‘start station longitude’, ‘end station latitude’, ‘end station longitude’. And distance between origin and destination. Finally, we got 11 features in total.

3 TRIP PREDICTION MODEL

3.1 Model details

In this section, we will describe the model we used for trip prediction problem. But before that, we needed to make some clarification about the trip prediction problem itself. This problem aims to make a prediction about top 3 most likely destination. The input is start station code, start hour, month, weekdays and membership. Then the output is the top 3 inference scores and correspond destination code.

Here, we formalized the whole prediction process. For a given origin and destination pair, we used (s_o, s_d) to represent it, where s_o indicated origin and s_d indicated destination. Furthermore , for training samples with (s_o, s_d) pair, it can be represented as vector $D(s_o, s_d) \in \mathbb{R}^N$, where N is the number of features extracted in section 2.2. For each pair (s_o, s_d) we want to know the inference score touched to it. The inference score can be interpreted as the likelihood for a trip start at s_o and finish at s_d. We represented the relevance scores as $P(s_o, s_d)$, $0 \leq P(s_o, s_d) \leq 1$. Thus, the problem can be formalized as building a function $f: \mathbb{R}^N \rightarrow P(s_o, s_d)$, which means function f map the input feature vectors to their inference scores.

3.2 Destination filtering

For any given origin station, there are about 540 possible destinations, if we calculated inference scores for every possible station, it would be computational expensive and pointless to some extent. We need metrics to do filtering and then build a candidate set. Guided by this idea, we decided to find top N most frequent used destination for every station, which could be done by counting and sorting the total amount of trips end up at a given station. However, a problem occurred, what is the suitable value for N? To deal with this problem, we tried different value of N and computed correspond percentage of trips whose destination was one of those top N station. The result is shown in fig.2. After observing fig.2. we decided to choose 200 as the value for N, we thought this percentage is high enough to keep most of information and the correspond cover percentage is 90.79%. This means for any given station, 90% of trips which start at this station end up at within in its top 200 destination set.

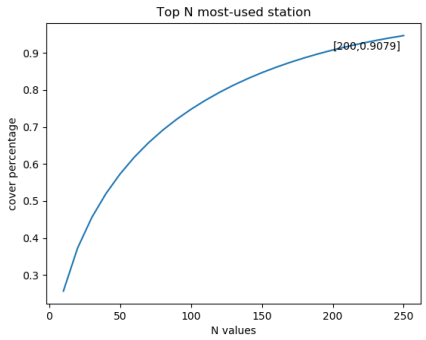

Fig. 2. Relationship between N values and correspond percentages of covered trip

3.3 Algorithm

As mentioned above in section 3.1. The trip prediction problem can be mapped to a binary classification problem. There are many models and algorithms satisfy the demand, such as Random Forest(RF), Gradient Boosting Decision Tree(GBDT), and a state of art algorithm eXtreme Gradient Boosting(XGBoost)[5] which was what we used in this paper. RF model is based on bagging idea, it selects training sets and training features randomly for every single tree in the forest, which increase the randomness of model significantly. The existence of randomness enables every single decide tree to be unique and increases diversity of system, thus the performance of model will be improved. GBDT is an application of boosting methods, when training a new tree in the model, it fits the residual of all existed trees. As the result, prediction performance will be improved recursively through the whole training process. The training process of GBDT model is serial, so compared with parallel training process of Random Forest, GBDT is more time expensive. XGBoost is a meta-model for GBDT model. The algorithm adds regular term which is used to control the complexity of model into objective function. The regular term is a function about number of leaves and L2 norm of leaf score.

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{k} \Omega(f_k) \tag{1}$$

$$\Omega(f_t) = \gamma T + 0.5\lambda \sum_{j=1}^{T} w_j^2 \tag{2}$$

Where $l(y_i, \hat{y}_i)$ is training loss and $\Omega(f_k)$ is the complexity of the tree. T represent the number of leaves and w_j^2 is L2 norm of leaf scores.

Different from classic GBDT model, towards cost function, xgboost algorithm does taylor series expansion, introducing both first derivative and second derivative into optimization process. This actually improve the accuracy performance.

4 TRIP ADVISOR MODEL

4.1 Activeness definition

To narrow the gap between stations' usage, we need to quantify the usage of stations first. Our method is to define an activeness level for every station and find a way to reduce the standard deviation of all activeness values. We assumed that a station was more active if more bikes rent from or return to this station. So the activeness of stations can be given as

$$a = \frac{gamma * \sum b_i + (1 - gamma) * \sum r_i}{N} \tag{3}$$

Where $\sum b_i$ is the whole number of bikes borrowed from station i. $\sum r_i$ is the whole number of bikes return to station i. N represent the whole number of samples been used to establish activeness. Gamma is a damping factor to control bias, if $1 \geq gamma > 0.5$, the activeness definition will be partial to borrow number than return

number, vice versa. From formulation (1), we can see that the activeness for every single station will recursively updated as more data flowing in.

4.2 Activeness update and station shift

Given the formulation to define activeness level, we still need algorithm to balance the activeness level between close stations, and update the activeness with the process.

The function of the trip advisor we want to achieve is that once users have decided origin and destination, advisor offers an alternative origin and an optional destination, this origin and destination pairs helps balancing the station activeness. If users think this station shift will affect their experience, they feel free to reject this recommend.

In this paragraph, we will talk about the details of how to choose alternative stations. The candidates will be selected based on distance, to be more specific, for a given origin or destination, the alternative station will be selected from its nearby station. We conducted some experiments to estimate the influence of different distance threshold. The result is shown below in fig.3.

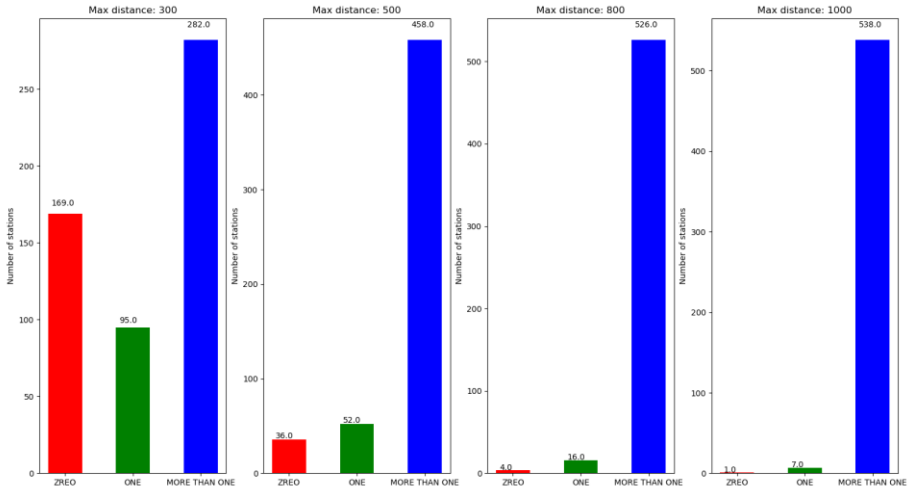

Fig. 3. Number of station shift choice for a given max distance

From fig.3. we can see that if we set distance threshold as 300 meters, about 170 stations have no nearby stations within such a short distance, near 100 stations only get one nearby station. With the increase of distance threshold, number of stations which own no or just one nearby station drops significantly. When the threshold reach 1000 meters, almost all the stations have at least 2 other stations nearby. So we set the max distance as 1000 meters. Once we have decided the near station sets, we should select stations with minimum activeness as the recommendation.

5 EXPERIMENTS

5.1 Experiment settings

In the trip prediction part, we conducted feature extraction mentioned in section 2.2 for data of year 2016 and 2017, the total number of records during those two years is more than 4.7 million. It is too large for us to do the training. So we kept only one samples every 100 samples. Finally, we got about 47000 positive training samples, which were assigned with label 1. To avoid class unbalanced problem, we decided to generate equal-sized negative samples randomly and assigned those samples with label 0. For feature 'membership', 'subscribed user' and 'unsubscribed user' can be assigned in equal chance. Origin and destination code will be randomly selected from whole station set. Departure time can be randomly generated from April 1st 0 am to November 30th 23 pm. Those more than 90000 samples form the whole training set.

Then selected 1500 real-world records from feature extracted dataset randomly, combined with other 1500 randomly generated negative test samples to form the whole test set with 3000 samples.

For trip advisor part, we used data from year 2016 to 2017 to establish the initial activeness for every station.

Fitting models with training set, in this paper, we compared the performance of Random Forest, classic Gradient Boosting Decision Tree and GBDT with XGBoost algorithm. To make the result of this comparison convincing, we all used default parameters for three models. The accuracy, precision and recall rate for those three models are shown in fig.4. From fig.4. we can see that XGBoost model performed slightly better in terms of all three metrics.

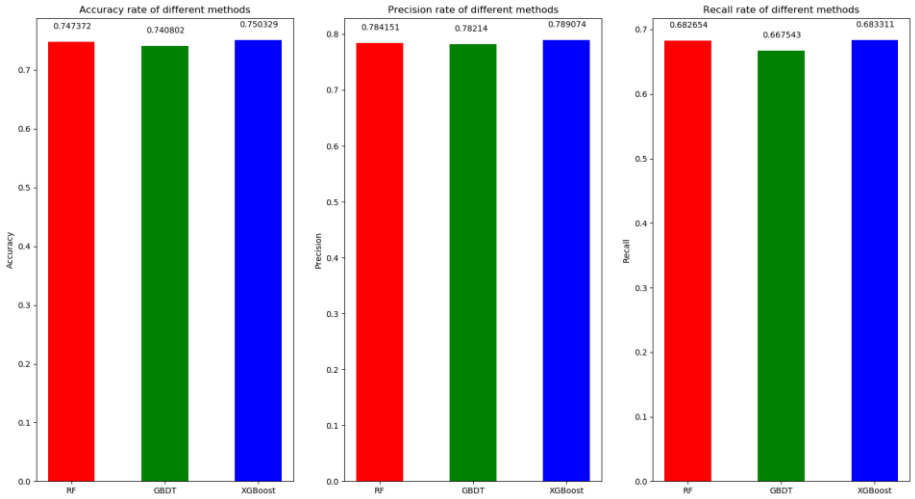

Fig. 4. Accuracy, Precision and Recall rate of RF, GBDT and XGBoost model

Furthermore, we evaluated the time efficiency of three models, to be more specific, we computed average training time and correspond variance for three models. The result is given in fig.5.

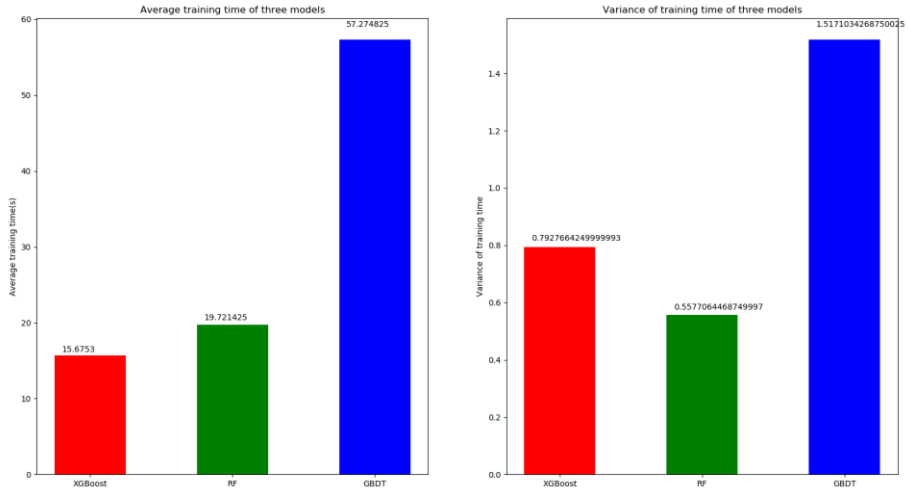

Fig. 5. Average training time and correspond variance of three different models

It is clear that XGBoost model owns highest time efficiency, it cost only one fourth of training time of GBDT model. Based on all the results mentioned above, we decided to apply XGBoost as our model to tackle the trip prediction problem.

5.2 Model parameter settings

We took advantage of cross validation(K_folds, k=5) to adjust some key parameters for XGBoost model. We used accuracy score, precision score and recall rate as metrics, if it was hard to decide the optimal value, we would introduce F1-measure score as the fourth metric.

First, we needed to find the optimal number of estimators. But before that, we set other parameter as followed: learning_rate=0.1, max_depth=5, min_child_weight=1, subsample=0.8, colsample_bytree=0.8, objective='binary:logistic', reg_alpha=0.1. To make this report short, we only presented the results around optimal values.

For parameter 'n_estimators'. The accuracy score and precision score reached their peak both at 220, so we set 220 to 'n_estimators' in afterward experiments.

The second parameter we wanted to estimate was 'max_depth'. Three metrics reached their peaks at different value of 'max_depth'. So we used F-measure as the forth metric to decide the optimal value. Finally, we thought 9 is the most suitable value. Then came to the parameter 'min_child_weight', we tested this value ranging from 1 to 9. It was lucky that three metrics reached global maximum or local maximum at the same value, that is 5.

The last parameter we estimated was 'gamma', it was a parameter which had strong relationship with loss function. The less value it was, the more conservative the

model was. After taking accuracy score and F-measure score both into consideration, 0.2 was the best value we got.

Using testing set to evaluate the final result. it is shown in fig.6.

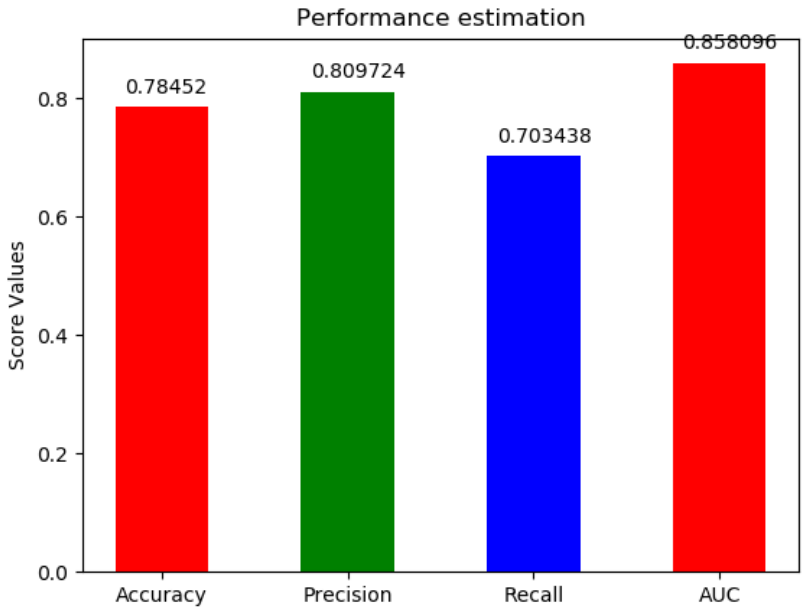

Fig. 6. Performance of XGBoost in terms of four metrics

5.3 Measure improvement degree of unbalance usage

After using data from year 2016 to 2017 to establish initial activeness for stations, we conducted simulation to evaluate the effectiveness of our algorithm. Due to the lake of real-time record, we used the same dataset to simulate the situation. But at this time, for any input station pairs (s_o, s_d), we pretended that the user got second chance at that time to decide whether to accept our recommendation. We used standard deviation as metrics to measure the improvement degree. After every 100 samples processed, we updated activeness for every stations, ensuring the change of activeness exerted its influence on the recommendation result in time. The result is shown below, from the figure, we can see that if all users accept our recommendation, the maximum decrease percentage of standard deviation of activeness is more than 35%, if only half of user accept the suggestion, the decrease percentage is 25%.

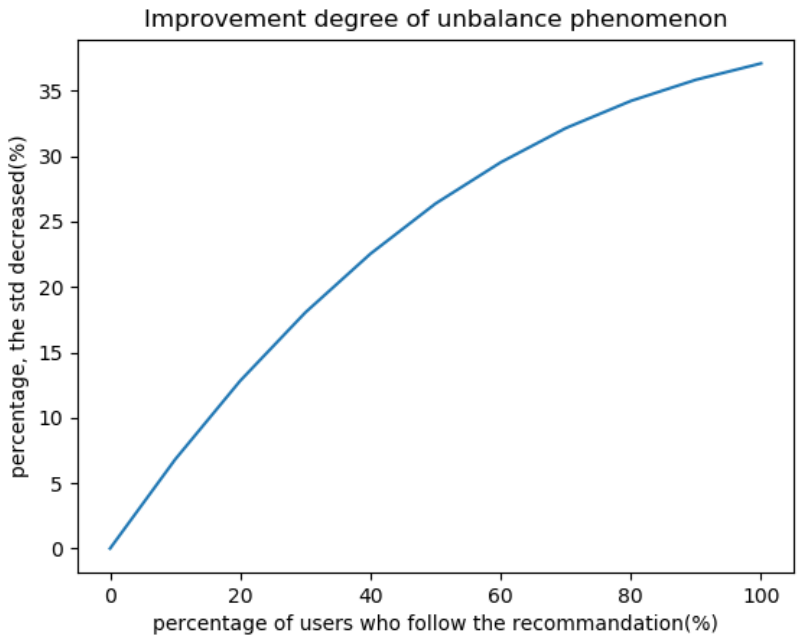

Fig. 7. Improvement degree of unbalance usage

6 CONCLUSION

In conclusion, in this paper, we achieved our two functions, namely, trip prediction and trip advising. Our innovation point lies on the new algorithm we had introduced, Different from the using MART, or other classic machine learning models such as GBDT, random forest. We introduced a state of art model XGBoost, it was much faster and performed better compared with other models. Then, we applied some metrics to adjusting the parameter for the new model. In general, the prediction performance was acceptable. Besides, we spot the unbalance problem between the usage of stations. To solve this problem, we defined 'activeness' for every station and updated the activeness with input flowing in.

There are still so many domains which we can explore deeper in the future. To be more specific, if we had more personal information, we could establish a personal document for every single user, then based on those documents, we could produce clustering and make our prediction result more precise and more personalized. Besides, we could find a more suitable objective function for XGBoost model in this specific situation. Furthermore, we could improve our trip advisor by adding more functions into it, for example, recommending the shortest path or nearby bus stations, restaurants.

References

1. Zhang, J., Pan, X., Li, M., & Yu, P. S.: Bicycle-Sharing System Analysis and Trip Prediction. IEEE International Conference on Mobile Data Management 2016, pp.174-179. IEEE(2016).
2. Hu, Ji & Yang, Zidong & Shu, Yuanchao & Cheng, Peng & Chen, Jiming. (2017). Data-Driven Utilization-Aware Trip Advisor for Bike-Sharing Systems. 167-176. 10.1109/ICDM.2017.26.
3. El Mahrsi, Mohamed & Côme, Etienne & Baro, Johanna & Oukhellou, Latifa. (2014). Understanding Passenger Patterns in Public Transit Through Smart Card and Socioeconomic Data: A case study in Rennes, France.
4. Purnama, I. B. I., Bergmann, N., Jurdak, R., & Zhao, K. (2016). Characterising and Predicting Urban Mobility Dynamics by Mining Bike Sharing System Data. Ubiquitous Intelligence and Computing and 2015 IEEE, International Conference on Autonomic and Trusted Computing and 2015 IEEE, pp.159-167. IEEE.
5. Chen, T., & Guestrin, C. (2016). Xgboost: a scalable tree boosting system. 785-794.