Pittsburgh Bike Share Data: Progress Report

Junwei Zhuge
1st Year Graduate Student in MSIS
Program (School of Information Science)
University of Pittsburgh
Juz25@pitt.edu

Ayush Mishra
1st Year Graduate Student in
University of Pittsburgh
amm428@pitt.edu

ABSTRACT
In this project, two different approaches to predict Pittsburgh Bike Sharing Demand are studied. The first approach tries to predict the exact number of bikes that will be rented using classification models. The second approach tries to predict the approximate count number of rented bikes at a station using regression models. The aim of this project is to predict the bike share counts for different stations across the city of Pittsburgh. The raw data is transformed to obtain actual results. In this Progress report, we made use of K-NN, LOGISTIC REGRESSION(LR), and NAÏVE BAYESIAN(NB) methods to train our model and predict the counts. A further aim is also to compare these counts against the rack quantities at each of the bike stations. The predictions based on our models can help determine the optimum time matrix to "re-balance" the bike accumulation and restore them to their optimum levels across the city.

Keywords
Classification; Regression; K-NN; LOGISTIC REGRESSION; NAÏVE BAYESIAN; Re-balance; Bike Sharing Demand

1. INTRODUCTION
In this report, we first combine the Q3 and Q4 bike share data into a single entity to ease the processing. This gives us a consolidated view of the data and we can work on this to give us a better view of the response variable that we need to predict. At this point however, we do not have the actual response variable. We further process the dataset to get rid of "missing values". After this, we split some consolidated columns such as "StartTime" and "StopTime into more edifying and workable columns.

Our goal is to predict the counts that are *rented out* and are *deposited in* to each of the stations each hour. Thus, we move one step forward by splitting the above mentioned consolidated columns into more processible data such as "StartMonth", "StartDay", "StartHour", "StopMinute", etc. This brings us closer to determining the response variable.

We aggregate the dataset by grouping together predictor variables and determining the counts. This count data is our response variable. The detailed methods of our predictions are shown in the following sections.

2. RELATED WORK
First, we plot some density plots based on those potential useful attributes, including "Hour", "Day", "Month", and "UserType". Plots are showed in Figure 1 to Figure 4.

According to Figure 1, the "Pitt Bike Trips by Hour of Day" plot, the fastigium of Pitt bike trips is between 8 - 20. According to Figure 2, the "Pitt Bike Trips by Day of a Month" plot, we will not take account of "Day" element, instead, we will calculate weekdays and weekends from Month and Day, and take weekdays and weekends as variables. According to Figure 3, the "Pitt Bike Trips by Month" plot, July and August have the most bike trips, and the bike trip amount declines month by month till December. In Figure 4, 1 represent Member (pay as-you-go customer), 2 Subscriber (deluxe and standard monthly member customer), and 3 Daily (24-hour pass customer).

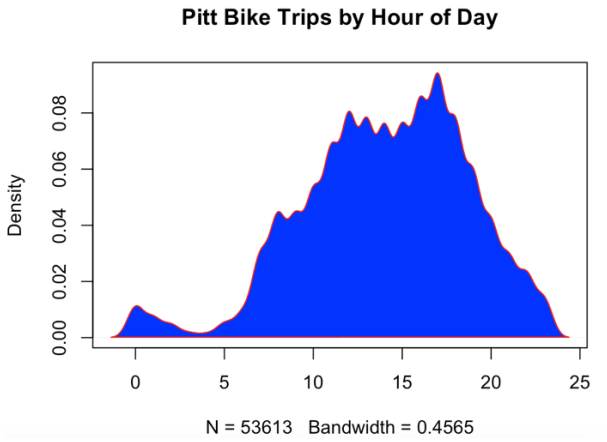

Figure 1- Pitt Bike Trips by Hour of Day

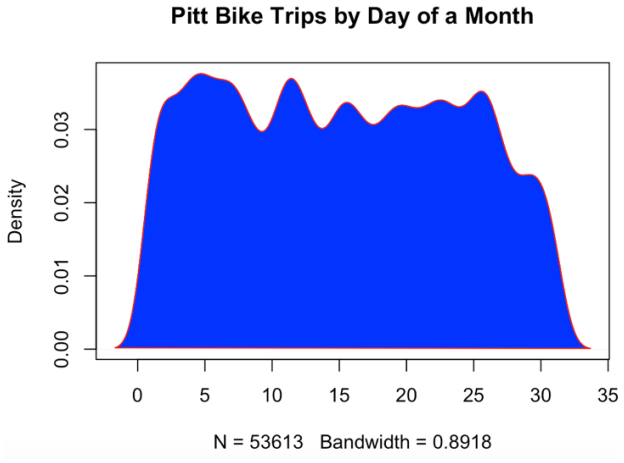

Figure 2- Pitt Bike Trips by Day of Month

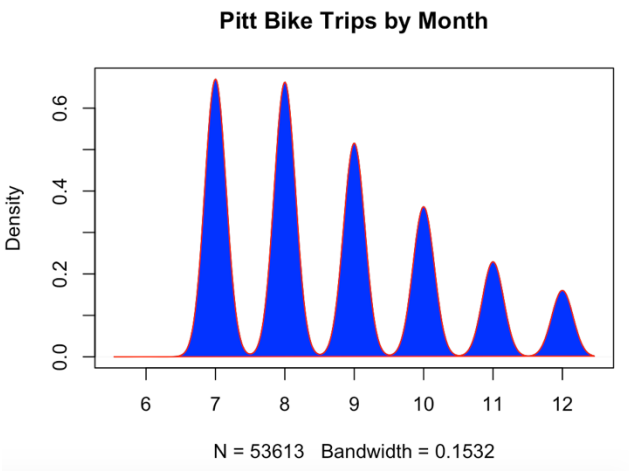

Figure 3- Pitt Bike Trips by Month

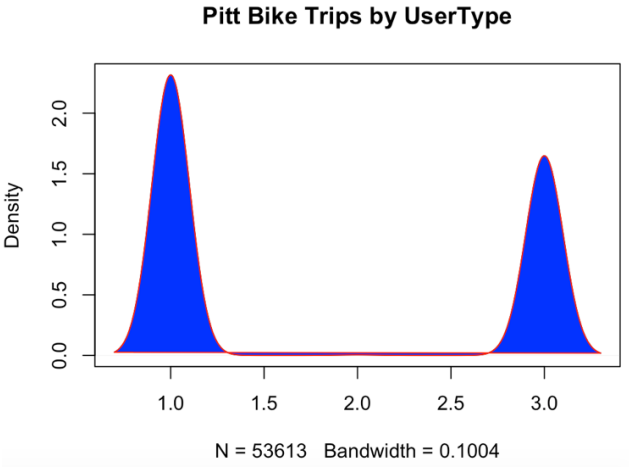

Figure 4- Pitt Bike Trips by UserType

Besides, we deleted some attributes that are not important for the results including "StartMinutes" and "TripId", etc., and add some important attributes such as "Weekdays" and "Qnty". Split the data set into two data sets, "Start" data set and "Stop" data set. Calculated the "StartCount" and "StopCount" separately from these two data sets based on "UserType", "StationId", "Month", and "Day". Finally, we calculated the "Final Count" from 'StartCount" and "StopCount". If the Count is larger than 0, it means bikes

In our final table, the variables will be "StationID", "UserType", "Month", "Weekday", "RackQnty", and the "y" will be the bike trip count of a specific station at a day, values with "-" means out, "+" means in. So we can compute the final bike amount of a station at one day, so we can decide the rebalancing problem based on the amount and the "density plot based on Hour".

What's more, according to the Figure 1, "Hour of Day" seems an important variable. However, the "StartHour" and "StopHour" of every trip between training and testing set cannot be exactly the same, so I delete these two column, but I will use "Hour" element as an important factor to decide what time (a time period in a day) is the best period to retransfer the bikes, just as "density plot based on StartHour" showed.

3. DATA SET

The data set used for building our models is from preprocessing after multiple iteration, including deleting many attributes and adding useful attributes, and splitting variables to get more meaningful variables.

First, we combine two data frames, Q3 and Q4 into one, and then delete StationName attribute, only using StationID to build the model. In this way, the data set would look more clean. And delete rows with missing values, because we have a huge data set, the deleting missing values would not influence much. For the timestamp attribute, we split StartTime into StartDay, StartMonth, StartYear, StartHour, and StartMinute, and split StopTime in the same way. And based on the density plot, we delete those meaningless variables.

In the second phase of preprocessing, we add a Weekday attribute. It doesn't mean weekday, instead, it means days in a week. We calculated "Weekdays" from year, month and day separately for StartTime and StopTime, matching each "Day" attribute with a "Weekday". Although there are only totally 4 weekdays finally got in the dataset, Tuesday, Thursday, Friday, and Sunday, they still could be very important and meaningful variables to our models.

Then we need to compute the final count of one station, one user type and at a specific time. First, we extract rental "Stop" data, and merge with station data, left outer join on "ToStationID", and create dataframe with group and "Count" variable. Do the same way on "Start" data to calculate the "StartCount". And then merge two sets by the same station id and the same user type and the same day, "Start" and "Stop" into one to calculate the "FinalCount" by StartCount minus StopCount.

Finally, we split the data set into training set and testing set. Training set is the dataset which days are before 20th in every month, and testing set is after 20th in every month.

Data Column	**Description**
y	Count = StartCount - StopCount
StationId	All Pittsburgh station ID
Usertype	User Type (1 - Member (pay as-you-go customer), 2 - Subscriber (deluxe and standard monthly member customer), 3 - Daily (24-hour pass customer))
Month	In which Month
RackQnty	The rack quantity of one station
Weekdays	Days in a week (Tuesday – 5, Thurday – 4, Friday – 1, Sunday – 3)

Table 1- Dataset Variables and Descriptions

4. METHODS

For the initial processing we choose to apply classification models to predict the bike final count. But it seems classification cannot work in our project, because there are too many classes if we deal the project as classification issue. And then we decided to use

regression models, applying K-NN (1,5,10), LR and NB methods to show the contrasts in the predicted data. We created a new method to measure the accuracy of our model. In consideration of the bike sharing demand issue in the real world, it is not that meaningful to predict the exact number of bike sharing count of one station, and it seems not possible to predict the exact number. Life is full of changes. We decided to measure the "true positive" by a *range*. If the predicted number is in the range of [x-2, x+2] (x is the real value in testing set), we consider it as *AT* (Approximately True). These methods give us a peek at the prediction of the response counts as described in the following.

4.1 Classification

This part is written by Ayush.

4.2 Ridge Regression

Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity. When multicollinearity occurs, least squares estimates are unbiased, but their variances are large so they may be far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors.

lm.ridge(y ~ ., data = training, lambda = 0.1)

In our project, we implemented a ridge regression model, setting the lambda to 0.1, and use our preset method to calculate the accuracy.

$$accuracy = \frac{\text{Approximately True Count}}{size}$$

Where:

Approximately True Count: if the predicted value is larger than actual value – 2 or larger than actual value – 2, which we defined before, *size*: the total number of actual values.

First we use [actual value – 1, actual value + 1] to measure the accuracy, the result is good but not good enough, and because 1 and 2 don't have much difference, also considered the station size, we finally decided to use [actual value – 2, actual value + 2] to measure the accuracy. The result is pretty high, around 0.741.

4.3 Linear Regression

This model returns the most basic regression results. The variables are all of first order:

lm(y ~ StationID + Month + Day + Weekday + UserType, data = training)

Based on the measure method provided before, the prediction gives us an accuracy of 0.7358.

4.4 Polynomial Regression

This model provides a slightly lower performance fit in data with 2nd order variables in addition to what we had in the linear regression model.

The variables are of first as well as second order:

lm(y ~ StationID + Month + Day + Weekday + UserType + I(Month^2) + I(Weekday^2), data = training)

Based on the measure method provided before, the prediction gives us an accuracy of 0.7308.
This gives us the worst accuracy result for now in these three regression models.
So we decided to modify the variables in the polynomial model.

lm(y ~ StationID + Month + Day + Weekday + UserType + I(Month^2) + I(Weekday^2) + I(Day^3) + I(StationID^3), data = training)

This model's performance decreases drastically as compared to the former polynomial model. Based on the measure method provided before, the prediction gives us an accuracy of 0.4203. This drastic drop in accuracy renders further use of nonlinear models unusable.

4.5 Logistic Regression

This model returns the same regression results as the linear model.

The variables are all of first order, but in this case threated as factors in stead of numeric:

lm(y ~ StationID + Month + Day + Weekday + UserType, data = training)

Based on the measure method provided before, the prediction gives us an accuracy of 0.7358.

Overall, the ridge regression gives us the best result in this project.

Model	Accuracy (%)
Classification	
Ridge Regression	74.10
Linear Regression	73.58
Polynomial Regression_1	73.08
Polynomial Regression_2	42.03
Logistic Regression	73.58

Table 2- Model Comparison

5. DISCUSSION

Our best model is ridge regression model, the accuracy is still not very high, but it is pretty high compared with the results of other groups. It can be so high thanks to the measurement method we created. It is created not just because we want to increase the accuracy, but more importantly, it is based on the actual situation in the real world. To compute and predict the exact bike sharing number of one station needs much more effort, and it cannot be 100% accurate still. To retransfer 5 bikes into one station is about the same with retransferring 4 bikes into one station or 6 bikes. Usually bikes would not be totally rented out, especially in US, the population who ride bikes is not as big as the number of bikes a station can afford. People always drive their cars or just walk.

6. CONCLUSION

Two approaches were taken to try and predict the bike sharing demand. In the first approach, the classification model, initial result gave a really low accuracy of 0.0131. The result contained a high variance which could be due to not having a large enough training samples or not selecting features that would be optimizing the problem. Although we tried many classification models to build the model, it seems not possible to get a good result using classification model, which could be due to having too many classes of the respond variable - final count. So we choose to apply regression models to do this project. We tried linear regression, polynomial regression, logistic regression and

ridge regression, in which the ridge regression has the best result. And this best result also profits from the new measurement method that we use a range [actual value – 2, actual value + 2] to measure the predicted value. If the predicted value is in this range, we consider it as a true value. And we sum all the true values as "AT", and then divide "AT" by the total number of actual values in testing set the to get the accuracy. This measurement is very important for our project, because it contains our understanding of the bike sharing demand task.

As for the regression approach, the four models, ridge regression, linear regression, polynomial regression and logistic regression gave different results up to around 74.1%, with ridge regression outperforming those other three models. The results contained a high variance as well which could be due to the same reasons mentioned above.

7. FUTURE WORK

With the predictions at hand, we wish to address the rebalancing problem. We wish to use the results obtained by the ridge regression model to obtain the best fit cost matrix. In order to achieve this, we want to create a location based cost matrix that can maximize the operability by maintaining the bike flow equilibrium at minimal cost. There are some problems we need to address:

- The location based cost-matrix needs to be weighted according to its proximity to each of the stations.
- An algorithm needs to be in place for determining the best possible method to re-balance. This is the most challenging problem; we will discuss later.
- Proximity problem:

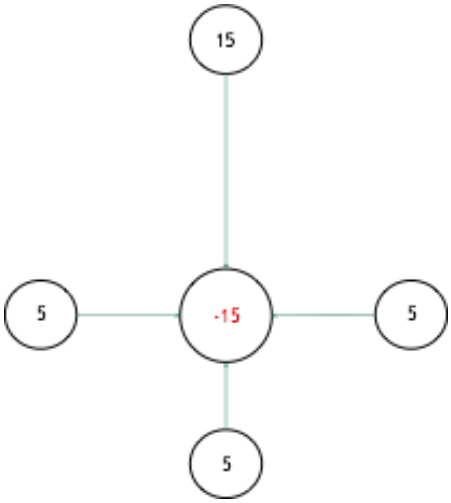

Figure 5- Re-balancing example_1

For example, there are four stations that have positive values of final count (A=15, B=5, C=5, D=5), which means we need to retransfer the bikes in those four stations to other stations. Now we have a station (E=-15) which has a negative value of final count at the center of those four stations.

First we need to build a distance matrix of those stations based on their real longitude and latitude. In this case, we have to calculate the total distance between B and E, C and E, and D and E, compared with the distance between A and E. If the distance AE is larger than BE+CE+DE, we should retransfer bikes from B, C and D to E, instead of retransferring bikes from A to E, because it is farther from A to E. However, this is just a very simple case, containing only 5 stations with only one "center station". In our project, we need to deal with a much more complicated situation like the graph below:

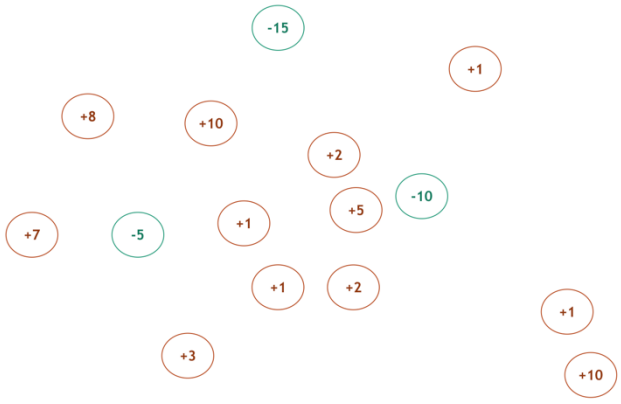

Figure 6- Re-balancing example_2

In this case, the start station is really important for the retransferring process. From which station we start to retransfer the bike has a big influence on the efficiency and the final result of re-balancing. And the route of retransferring also matters, which is decided by choosing which station as a bike provider for another station. The weighted matrix could be really complicated when the number of stations reaches a high value.

Actually, a very challenging problem for us need to be considered before those problems discussed above is we need a spatial data set to deal with this task. For now, we only have .csv file which only contains plain data information with no spatial data information.

Therefore, we decided to use methods and models on spatial data analytics field to deal with the re-balancing task. Putting this normal data mining task into spatial data analytics field is a challenge. But when we get the spatial information and after we switch to spatial field, it could be much easier for us to think about the weighted matrix problem and retransferring problem.

8. ACKNOWLEDGMENTS

I would like to express my deep gratitude to professor Yuru Lin for providing us such an excellent and pragmatic project task. From researching this project, we know how to use data mining knowledge to resolve real world tasks. I would also like to thank Teaching Assistant Ruoxuan Chen for her surprisingly delicate assist.

I would also like to extend my thanks to the technicians of the laboratory of the Information Science department for their help in offering me the resources in running the program.

Finally, I wish to thank my parents for their support and encouragement throughout my study.

9. REFERENCES

[1] NCSS Statistical Software. 2016.Ridge Regression. 335, 1. DOI= http://www.ncss.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Ridge_Regression.pdf.

[2] Mahmood Alhusseini. 2014. Prediction of Bike Sharing Systems for Casual and Registered Users. DOI= http://cs229.stanford.edu/proj2014/Mahmood%20Alhusseini,Prediction%20of%20Bike%20Sharing%20Demand%20for%20Casual%20and%20Registered%20Users.pdf.

[3] Kaggle Competition. Bike Sharing Demand. https://www.kaggle.com/c/bike-sharing-demand.

[4] KELSEY E. THOMAS. 2016. What 22 Million Rides Tell Us About NYC Bike-Share. 1 (January 28, 2016). DOI= https://nextcity.org/daily/entry/citi-bike-new-york-city-bike-share-data.

[5] Ray, S. 2015. Kaggle Bike Sharing Demand Prediction – How I got in top 5 percentile of participants? *BUSINESS ANALYTICS*. DOI= http://www.analyticsvidhya.com/blog/2015/06/solution-kaggle-competition-bike-sharing-demand/.

[6] Bay Area Bikeshare OPEN DATA CHALLENGE. 2014. http://www.bayareabikeshare.com/datachallenge-2014.

[7] Predictive bikeshare rebalancing. 2014. https://github.com/dssg/bikeshare.

[8] Raviv, T., Tzur, M. and Forma, I. 2013. Static repositioning in a bike-sharing system: models and solution approaches. 2, 3 (Aug. 2013). 187-229. DOI= http://link.springer.com/article/10.1007/s13676-012-0017-6.