Demand Prediction in Bike Share Systems

Zhaonan Qu

January 8, 2018

1 Introduction

Bike Share systems are becoming increasingly popular in urban areas. With growing membership and expansion of service comes many operational challenges. A major challenge in their operations is the unbalanced demand and supply at bike stations as a function of time. Figure **1** shows number of bike trips in Jan 2017, aggregated into time intervals of 30 minutes according to start time, and summed across the entire month. We see that there is a clear temporal dependence of bike demand. This temporal dependence is coupled with geospatial depencence: work districts have a higher demand during evening rush hours whereas residential areas have a higher demand during mroning rush hours.

Most bike share systems employ active rebalancing to ease the pressure at peak times. This means transporting a certain number of bikes from inactive stations to more active stations, or between stations and storage, in order to maximize the usage of each bike and ease supply and demand inbalance problems across bike stations at different times.

A quantitative, predictive model for the demand and supply at bike stations would help operators plan bike transports more efficiently. This project aims to build such a model for bike arrivals at each station within one-hour time intervals, as a function of the following parameters:

- time of day, divided into 24 one-hour intervals
- whether a day is a work day or weekend/federal holiday
- hourly temperature
- hourly precipitation intensity

The output of our model will be the number of bikes departing from a station within the one-hour time interval.

Moreover, we also employ customer-level data, including:

- gender
- age

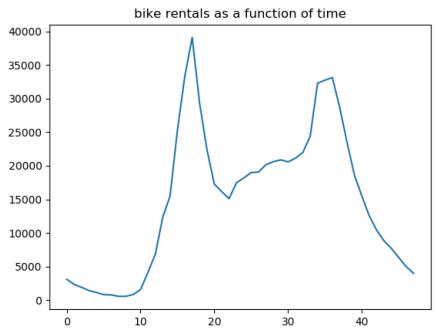

Figure 1: Bike usage in January 2017 as a function of time, discretized to 30-minute intervals.

tripdurati	starttime	stoptime	start statio	start statio	start statio	start statio	end statio	end statio	end statio	end statio	bikeid	usertype	birth year	gender
254	5/1/2017 0:00	5/1/2017 0:04	511	E 14 St & A	40.72939	-73.9777	394	E 9 St & Av	40.72521	-73.9777	27695	Subscribe	1996	2
248	5/1/2017 0:00	5/1/2017 0:04	511	E 14 St & A	40.72939	-73.9777	394	E 9 St & Av	40.72521	-73.9777	15869	Subscribe	1996	1
1120	5/1/2017 0:00	5/1/2017 0:19	242	Carlton Av	40.69779	-73.9737	3083	Bushwick	40.71248	-73.941	18700	Subscribe	1985	2
212	5/1/2017 0:00	5/1/2017 0:03	168	W 18 St &	40.73971	-73.9946	116	W 17 St &	40.74178	-74.0015	24981	Subscribe	1993	1
686	5/1/2017 0:00	5/1/2017 0:11	494	W 26 St &	40.74735	-73.9972	527	E 33 St & 2	40.74402	-73.9761	25407	Subscribe	1964	1
577	5/1/2017 0:00	5/1/2017 0:10	334	W 20 St &	40.74239	-73.9973	504	1 Ave & E	40.73222	-73.9817	28713	Subscribe	1956	2
523	5/1/2017 0:00	5/1/2017 0:09	335	Washingt	40.72904	-73.994	487	E 20 St & F	40.73314	-73.9757	15385	Subscribe	1994	1
419	5/1/2017 0:00	5/1/2017 0:07	336	Sullivan S	40.73048	-73.9991	369	Washingt	40.73224	-74.0003	18295	Customer	NULL	0
518	5/1/2017 0:00	5/1/2017 0:09	335	Washingt	40.72904	-73.994	487	E 20 St & F	40.73314	-73.9757	15608	Subscribe	1995	1
1296	5/1/2017 0:00	5/1/2017 0:22	291	Madison S	40.71313	-73.9848	291	Madison S	40.71313	-73.9848	17351	Subscribe	1994	1

Figure 2: Example of original data from Citi bike

combined with the above parameters, to predict the duration of a trip departing from a station.

For the demand prediction problem, we use neural networks to implement both classfication and L2 regression, and for the duration prediction problem, we implement L2 regression with neural network. The models are built using Tensorflow's tf.estimator API.

2 Related Work

There have been several previous academic and non-academic works studying the bike share demand prediction problem. Most notable are work done by Cornell researchers [3] and [2]. The Data Science for Social Good Lab has a project on predicting the probability of when a bike station is empty or full, hosted on Github[1].

The paper [3] is unique in that one of their main covariates is the number of taxi trips during morning rush hours. They find that there is a substantial positive correlation between the number of bike trips and number of taxi trips. However, such data is not readily available when we want to make real-time predictions. So in our study we only employ data available from bike share system itself, combined with readily-available weather information. A point worth noting from [3] is that they aggregate bike stations based on neighborhoods characterized by economic and demographic variables, and find that analyzing pairwise trips at the neighborhood level instead of looking at individual stations in bike sharing systems can improve the predictions. In our study, we focus on station-level predictions, but plan to pursue their suggestion in future work.

3 Dataset and Features

We use system data available from Citi bike's website[2]. The Citi bike data set contains information about each bike trip taken, including starting date and time, starting station, end station, trip duration, and customer age and gender. See Figure 2 for a snippet of data file.

We use trip data from June to August 2017. There are several cleaning procedures that we performed on the data set. First, we discarded trips shorter than 60 seconds and longer than an hour. Trips that are too short are likely results of change of plans by the users. Trips that are too long could results from bikes that are not properly docked. Then we discarded trips taken by day-pass holders rather than subscribers. This is because spontaneous trips taken by visitors are not the focus of this study. However, we note that it is a perfectly meaningful task to study the usage patterns of day-pass holders alone for a separate project. We also discarded those trips that are taken by Citi bike crew on rebalancing trips, as they do not reflect usage by users.

In order to perform regression on the demand of bikes, we discrete each day into 24 one-hour intervals, count the number of bikes departing from each station in each time interval, fill in observations where the count is zero (since they are not reflected in the original data).

Then we determine whether each date is a work day, i.e. not weekend or federal holiday, and finally associate weather data to each date and time bucket. Hourly weather data is obtained from Dark Sky, using a python scraper modified from code provided on the Github page of the bikeshare prediction project by Data Science for Social Good[3]. See Figure 3 for a snippet of scraped weather data, which is stored in a PostgreSQL database.

[1] Available at https://github.com/dssg/bikeshare

[2] https://www.citibikenyc.com/system-data

[3] Can be found at https://github.com/dssg/bikeshare/wiki/Methodology. The mathematical model used by that project is Poisson Point Processes, which are used to model the arrival at individual stations. This is considerably different from the current

time	summary	precipintensity	precipprobability	precipaccumulation	temperature
2017-06-01 00:00:00	Clear	0	0		60.75
2017-06-01 01:00:00	Partly Cloudy	0	0		60.57
2017-06-01 02:00:00	Partly Cloudy	0	0		59.65
2017-06-01 03:00:00	Clear	0	0		59.57
2017-06-01 04:00:00	Clear	0	0		61.98
2017-06-01 05:00:00	Clear	0	0		63.15
2017-06-01 06:00:00	Partly Cloudy	0	0		64.95
2017-06-01 07:00:00	Partly Cloudy	0	0		68.65
2017-06-01 08:00:00	Clear	0	0		71.09
2017-06-01 09:00:00	Clear	0	0		74.15
2017-06-01 10:00:00	Partly Cloudy	0	0		74.78
2017-06-01 11:00:00	Partly Cloudy	0	0		77.21
2017-06-01 12:00:00	Partly Cloudy	0	0		77.25
2017-06-01 13:00:00	Breezy and Partly Cloudy	0	0		78.09
2017-06-01 14:00:00	Partly Cloudy	0	0		76.64
2017-06-01 15:00:00	Clear	0	0		76.77
2017-06-01 16:00:00	Partly Cloudy	0	0		75.24
2017-06-01 17:00:00	Clear	0	0		74.16
2017-06-01 18:00:00	Clear	0	0		72.04
2017-06-01 19:00:00	Clear	0	0		70.8
2017-06-01 20:00:00	Clear	0	0		70.08
2017-06-01 21:00:00	Clear	0	0		69.21
2017-06-01 22:00:00	Clear	0	0		66.83
2017-06-01 23:00:00	Clear	0	0		64.51
2017-06-02 00:00:00	Clear	0	0		62.35
2017-06-02 01:00:00	Clear	0	0		60.6
2017-06-02 02:00:00	Clear	0	0		58.98
2017-06-02 03:00:00	Clear	0	0		57.66
2017-06-02 04:00:00	Clear	0	0		60.29
2017-06-02 05:00:00	Clear	0	0		63.48
2017-06-02 06:00:00	Clear	0	0		65.37
2017-06-02 07:00:00	Clear	0	0		67.27
2017-06-02 08:00:00	Clear	0	0		69.43
2017-06-02 09:00:00	Clear	0	0		71.44
2017-06-02 10:00:00	Partly Cloudy	0	0		73.13
2017-06-02 11:00:00	Partly Cloudy	0	0		72.71
2017-06-02 12:00:00	Partly Cloudy	0	0		72.86
2017-06-02 13:00:00	Partly Cloudy	0	0		72.91
2017-06-02 14:00:00	Drizzle	0.0067	0.67		74.22
2017-06-02 15:00:00	Breezy	0	0		73.7
2017-06-02 16:00:00	Breezy	0	0		72.5
2017-06-02 17:00:00	Clear	0	0		70.36
2017-06-02 18:00:00	Clear	0	0		68.09
2017-06-02 19:00:00	Clear	0	0		66.42

Figure 3: Weather data scraped from Dark Sky and stored in a PostgreSQL database.

Start_Time	Start_Station	Start_Station_L	Holiday	Count	Precipitat	Temperature
0	40.73221853	-73.98165557	0	4	0	60.75
0	40.73221853	-73.98165557	0	2	0	62.35
0	40.73221853	-73.98165557	0	0	0.0105	58.46
0	40.73221853	-73.98165557	0	3	0.0005	57.06
0	40.73221853	-73.98165557	0	2	0	52.37
0	40.73221853	-73.98165557	0	2	0	57.53
0	40.73221853	-73.98165557	0	1	0	55.4
0	40.73221853	-73.98165557	0	3	0	75.93
0	40.73221853	-73.98165557	0	2	0	78.67
0	40.73221853	-73.98165557	0	0	0.0008	77.13
0	40.73221853	-73.98165557	0	2	0	66.48
0	40.73221853	-73.98165557	0	2	0	63.45
0	40.73221853	-73.98165557	0	2	0	76.25
0	40.73221853	-73.98165557	0	0	0.0071	70.69
0	40.73221853	-73.98165557	0	1	0	73.04
0	40.73221853	-73.98165557	0	1	0	68.58

Figure 4: Processed data for bike demand prediction. Count is the number of bikes departing from the station within the one-hour time interval.

The final dataset consists of ~1.3 million observations, which we split into 70% training set, 20% validation set, and 10% test set. We also performed data normalization and found that this procedure improved predictions slightly.

A snippet of the processed data for demand prediction is shown in Figure 4, where the column "Count" is the number of bikes departing from a station in a specific time bucket. This data is then fed into various regression and classification models for training, evaluation, and prediction. We also normalized each feature, as well as divided the "Count" column by 10. For classification, we grouped the counts to intervals of length 4, for a total of 9 categories, where the last category is any number that is larger than 30.

For the prediction of trip duration, we use the same features, as well as the age and gender of the rider. The dependent variable in this case is the duration of the trip.

As an illustration of how weather affects bike usage, we selected a pair of stations with relatively high usage during morning rush hours. Station "Pershing Square North" is near Grand Central, and Station "East 24th Street and Park Avenue South" is in the Flatiron District, with shops, office space, as well as a university. Figure 5 shows the number of trips from the former to the latter in March 2017 between 6 to 6:30 am, as a function of the temperature during that time interval. We see a clear dependence on temperature. Moreover, bike usage patterns have a strong dependence on whether a given day is work day or weekend/holiday. So we also select this categorical variable as one of our features. For the problem of predicting trip duration given starting station and additional features such as customer age and gender, we used similar processing methods.

A similar trend in usage as a function of precipitation intensity also exists.

approach.

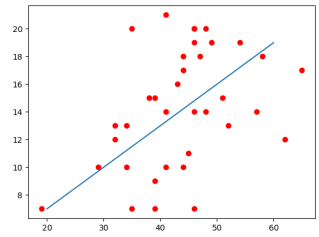

Figure 5: Weather dependence of bike usage between Pershing Square North and East 24th Street and Park Avenue South

We note that there are other covariates that one may consider. For example, [3] considers the number of taxi rides during rush hours. Data on the number of available bikes and docks at time of prediction can also be helpful, although this data is not publicly available. Another covariate is road and traffic condition. If there is a road closure, massive traffic jam, or delay in the subway system, this will naturally affect bike demand and supply at some stations. These covariates should be considered for a more sophisticated model.

4 Methods and Experiments

4.1 Linear Regression

Our baseline model is linear regression. The loss function for linear regression is given by

$$\ell(\theta) = \frac{1}{m}\sum_{i=1}^{m}(\theta x^{(i)} - y^{(i)})^2 + \alpha\|\theta\|^2$$

where α is the strength of regularization to prevent overfitting and we have used the convention that $x^{(i)}$ has entry 1 for the intercept term. The loss is minimized with gradient descent,

$$\theta \leftarrow \theta - \beta\nabla\ell(\theta)$$

where β is the learning rate.

4.2 Regression with L2 loss and Neural Network

As a comparison for both prediction problems, we used a three-hidden-layer neural network with L2 loss with regularization. The forward propagation for this neural network is given by

$$\begin{aligned}
\hat{y} &= W^{[3]}a^{[2]} + b^{[3]} \\
a^{[2]} &= \mathrm{ReLU}(W^{[2]}a^{[1]} + b^{[2]}) \\
a^{[1]} &= \mathrm{ReLU}(W^{[1]}a^{[0]} + b^{[1]}) \\
a^{[0]} &= \mathrm{ReLU}(W^{[0]}x + b^{[0]})
\end{aligned}$$

with regularized loss function

$$\ell(W,b) = \frac{1}{m}\sum_{k=1}^{m}(\hat{y}^{(k)} - y^{(k)})^2 + \alpha\sum_{i=1}^{3}\left(\|W^{[i]}\|^2 + \|b^{[i]}\|^2\right)$$

The conventional optimization method for back-propagation is gradient descent. For each mini batch, we calculate the gradients of ℓ with respect to $W^{[i]}, b^{[i]}$, and use the update rule

$$\theta \leftarrow \theta - \beta\nabla_p\ell_{MB}$$

where θ is a parameter and β is the learning rate.

For the implementation of neural network through DNNRegressor on Tensorflow, we use AdamOptimizer proposed in [1] that comes with the Tensorflow package. Briefly, the idea of the Adam (adaptive moment estimation) optimization method is as follows. As in gradient descent, it updates weights based on gradient, but the main difference with gradient descent is that Adam maintains parameter-specific learning rates and also adpatively changes them based on the first and second moments of the gradients of the weights in recent iterations. More specifically, it calculates an exponential moving average of the gradient and its norm squared, and parameters β_1 and β_2 control the decay rates of the two moving averages.

4.3 Classification with Softmax Function and Neural Network

We can also turn the prediction problem into a classification problem. More precisely, we assign each count to one of the 9 categories separated by $[-1, 4, 8, 12, 16, 20, 24, 28, 32, 1000]$, i.e. categories of length 4 starting from $[0, 4)$, $[4, 7)$, to $[32, \infty)$. Then we implement a classification algorithm, with softmax output, cross-entropy error, and 3-layer neural network. The set-up is similar to L2 regression with neural network. At the last layer, instead of using

$$\hat{y} = W^{[3]}a^{[2]} + b^{[3]}$$

we apply the softmax function to the output, which is now a vector in $\mathbb{R}^9$, so that

$$\hat{y} = \text{softmax}(W^{[3]}a^{[2]} + b^{[3]})$$

where for a vector v,

$$\text{softmax}(v)_i = \frac{e^{v_i}}{\sum_j e^{v_j}}$$

and cross entropy loss between observation $y \in \mathbb{R}^9$ and prediction $\hat{y} \in \mathbb{R}^9$, where $y_i = \delta_{i\ell(y)}$ is 0 except when $i = \ell(y)$ the class label of y, is given by

$$-\sum_i y_i \log \hat{y}_i = -\log \hat{y}_{\ell(y)}$$

5 Results and Discussion

5.1 Trip Duration Prediction

[Under development, need more computation power to train on the dataset.]

5.2 Demand Prediction: L2 Regression with Neural Network

We found that regression with deep neural networks greatly outperforms linear regression, so in this section we focus on L2 regression with deep neural network.

After 8000 epochs of gradient descent for deep neural network with hidden units $[64, 32, 64]$, the mean squared error for training and validation sets are 16.25 and 16.45, respectively. Figure 6 shows the training loss and validataion loss as a function of epochs. As can be seen in the plots of the errors, the errors steadily decrease over time and the error on the validation set is only fractionally higher than that of the training set. Note that since we divided the number of trips by 10, the mean squared errors in the training plots should be multiplied by 100. This corresponds to a root mean squared error (RMSE) of around 4.

While for a target variable (number of bikes) that ranges between 0 and around 50, an RMSE of 4 is not very small, we must analyze the data and see what kind of prediction accuracy is achievable given the quality of the data. For some stations, even given the same parameters and similar weather conditions, there is still intrinsic variance in the number of bikes departing from that station. This means that the variability of number

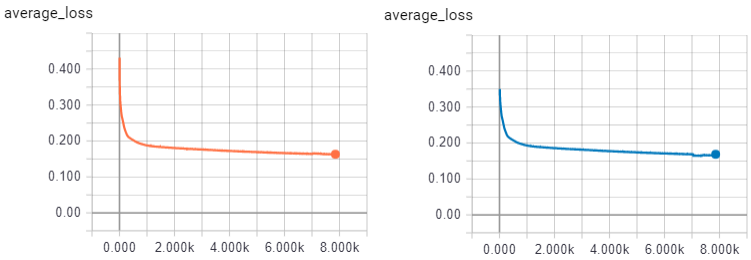

Figure 6: Left, training loss of DNN and right, evaluation loss of DNN.

	Actual_Count	Predicted_Count
373784	0.7	[0.442002]
848407	0.3	[0.455413]
289188	0.3	[0.266898]
1163392	0.6	[0.277636]
1271877	1.2	[1.73117]
29090	0.0	[0.0831463]
256596	1.2	[0.967924]
205980	0.7	[0.478312]
110251	0.1	[0.0546227]
266262	0.6	[0.403609]
699367	0.4	[0.444265]
447037	0.1	[0.306618]
847729	0.0	[0.397116]
1295428	0.1	[0.210978]
293420	0.5	[0.267062]

Figure 7: Actual number of rides compared to numbers predicted by trained neural network. Counts are divided by 10 during data normalization, so numbers above should be multiplied by 10.

of bikes cannot all be explained by our current features. What more features should we add? The most obvious is information on the number of available bikes and docks at a station at a give point in time. However, this historical data is not publicly available, but we are in the process of talking to Citi bike about obtaining this data. Beyond that, we note that the demand and supply at nearby stations are correlated with each other: If a bike station is out of bikes but there is a relatively well-stocked station a few blocks away, some customers will walk over to rent bikes from that bike station, increasing the short-term demand at that station. Therefore, it is sensible to group stations based on their neighborhoods and make predictions on those neighborhoods. After all, operators of bikeshare systems care more about the demand in neighborhoods rather than specific stations. Still, the current prediction accuracy provides a solid estimate of bike demand given available parameters. Below we show a snippet of the predictions made by our trained model on the test set. Note that there are some entries where the actual number of bikes is low, whereas the predicted number is high. This could result from the fact that there are no bikes available at a station, even though demand would have been high if bikes were available. This kind of discrepancy can be solved by including station-level data. Many other predictions show a trend of slight underestimation, which could be alleviated by combining bike stations in the same neighborhood, as the higher-than-expected demand could result from nearby stations running out of bikes.

5.3 Demand Prediction: Classification with Softmax Function and Neural Network

One important issue we need to address is that the number of classes in the data is very imbalanced: the frequency of class 0 is as high as 70.80%. This means that by simply predicting class 0 every time, the accuracy is as high as 70.80%. Clearly we need to address this problem. In this case, we utilize the weight_column functionality of the DNNClassifier in Tensorflow. This functionality allows a feature column that assigns a weight to each class in the loss function. We set the weights of the 9 classes to be inversely proportional to their frequencies. This solved the problem of dataset imbalance quite well: accuracy on the validation set rose

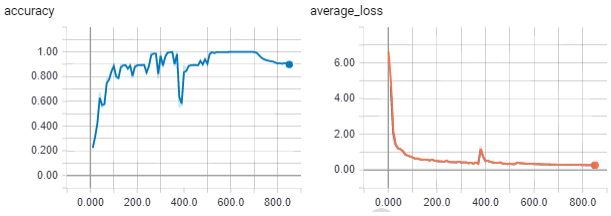

Figure 8: Left, prediction accuracy on validation set. Right, cross-entropy error on training(orange) and validation(blue) sets.

0	[b'0']
0	[b'0']
0	[b'0']
3	[b'3']
8	[b'6']
2	[b'2']
0	[b'0']
0	[b'0']
0	[b'0']
0	[b'0']
...	...
0	[b'0']
0	[b'0']
0	[b'0']
0	[b'0']
2	[b'2']
0	[b'0']
0	[b'0']
1	[b'1']

Figure 9: Actual class of the number of rides (left) compared to classes predicted by trained neural network (right).

to above 90%, with the optimal stopping of training at around 600 epochs, when the accuracy on validation set is 100%, while the average cross entropy loss for both training and validation sets continuously dropped during training. See Figure 8 for accuracy and average loss as a function of number of training epochs.

By examining predictions of trained model vs. actual class of each observation, we see that most predictions on the test set are correct, and even for mis-classified observations, the predictions made by the trained model are close to the actual label, usually within one category. This corresponds to an error of around 4, consistent with the regression results, and indicating that an error of 4 is intrinsic with the available data and selected features.

6 Conclusion and Future Work

We performed regression and classification with neural network to predict the demand for bikes at bike share stations, as a function of time, day of week, temperature, and precipitation, and to predict the duration of bike trips as a function of age and gender of customer. Regression with neural network yields an RMSE of around 4 for bike demand. While this is not considered low, it reflects the intrinsic limit of the dataset. We hope to improve the accuracy of regression by incorporating historical data on available docks and bikes at stations. We grouped the number of bikes into bins of size 4, and found that training a neural network yields a classification accuracy on validation set that is above 90%. The same approach can be applied to predict the number of bikes arriving at a station, i.e. the supply of bikes. Given the predictions for both demand and supply, a rebalancing plan can be formulated.

As rebalancing efforts by Citi bike accounts for a significant amount of bike transfers, it is also important to incorporate those transfers, if we are able to access more detailed data.

A fundamentally different approach could yield a more accurate prediction. So far we have only focused on stations instead of bike trips. We can instead model trips between pairs of stations as a function of time, temperature, holiday, etc. Moreover, we may consider grouping stations according to specific neighborhoods. This captures the intuition that a bike station within a few blocks of another bike station is likely to have

correlated supply and demand patterns, and can serve as a backup if other station nearby run out of docks or bikes.

If we can combine station-level data (available bikes and docks at a given time) with trip-level data, we can consider modeling the system as a network. This opens up the possibility of using Markov Chain models, which we are currently working on.

References

[1] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[2] Eoin O'Mahony and David B Shmoys. Data analysis and optimization for (citi) bike sharing. In *AAAI*, pages 687–694, 2015.

[3] Divya Singhvi, Somya Singhvi, Peter I Frazier, Shane G Henderson, Eoin O'Mahony, David B Shmoys, and Dawn B Woodard. Predicting bike usage for new york city's bike sharing system. In *AAAI Workshop: Computational Sustainability*, 2015.