Citi Bike Analysis: Predicting Potential Lack of Supply among Bikeshare Stations in NYC

Philip Ekfeldt
kae358@nyu.edu

Micaela Flores
mrf444@nyu.edu

Calliea Pan
cp2530@nyu.edu

Tony Xu
tx507@nyu.edu

I. Business Understanding

The Citi Bike bikeshare program offers short term bike rentals to customers in the New York City boroughs of Manhattan, Brooklyn, Queens, and Jersey City in New Jersey. Named after lead sponsor Citigroup, Citi Bike is operated by Motivate, which was acquired by Lyft in July 2018. The program currently operates a fleet of 12,000 bikes across approximately 750 stations, averaging about 62,000 daily rides as of September 2018 [1].

Citi Bike offers four pricing plans for customers:

- An annual membership with unlimited rides
- A three day pass with with unlimited rides
- A day pass with with unlimited rides
- A single ride rental

As the Citi Bike initiative is meant to offer an affordable transportation alternative for New York City residents and visitors, Citi Bike must ensure that its stations are sufficiently stocked to meet demand. When stations are improperly stocked relative to demand, two situations may occur:

- Customers wishing to check out a bike may be met with completely empty docks
- Customers wishing to check in a bike may arrive at a full station with no empty docks

In addition to riders transporting bikes throughout the city, Citi Bike carries out its own restocking operations to address demand. Citi Bike utilizes box trucks, vans, contracted trikes, and valets to redistribute bikes system-wide. In September 2018, Citi Bike staff rebalanced approximately 146,500 bikes across the city, averaging about 4,800 bikes per day [1]. However, even with Citi Bike's rebalancing operations, stations still experience outage periods, especially during the morning when riders commute to work and during the evening when they return home.

As part of its acquisition agreement with Motivate, Lyft has agreed to invest $100 million in Citi Bike to improve and strengthen the current system. Expansion plans include doubling the existing service area and tripling the number of bikes in the next five years [2]. As Citi Bike continues to grow, properly predicting demand will be key in the execution of its expansion plan. This is also the primary motivation for our project.

This project aims to design a classification model to predict periods of high demand as a proxy for bike shortages. The results would be used by management to properly direct rebalancing operations and reduce bike outages. Outages are currently a major customer grievance, and management will need to address issues of unmet demand if they are to maintain or grow its client base and revenue [3].

II. Data Understanding

The main data source of our project is Citi Bike's own publically available trip datasets [4]. Citi Bike provides historical datasets containing information about trips made by Citi Bike customers with features including:

- Start station (name, ID, longitude/latitude)
- End station (name, ID, longitude/latitude)
- Start time
- End time
- User type (which pricing plan the customer subscribes to)

The datasets are mostly clean and complete, with a few exceptions. In our analysis, we decide to focus only on bike trips made within the New York City boroughs from July 2018 to September 2018.

A few interesting statistics can be found from the raw dataset:

- The average bike trip is 17 minutes long
- 91% of trips are shorter than 30 minutes
- 65% of trips are shorter than 15 minutes
- The most active station is Pershing Square North close to Grand Central, with an average of ∼1000 interactions (check-ins and check-outs) per day

Some patterns across different types of stations also emerge. In Table I, we observe that there is a large spread in how stations are used. Some stations have much more bikes check-ins compared to bikes check-outs, leading to shortages in either bikes or empty docks. As mentioned previously, Citi Bike remedies these shortages by rebalancing bikes between stations.

TABLE I: The stations with the largest difference between number of check-outs and check-ins per day. A positive difference means more check-ins than check-outs.

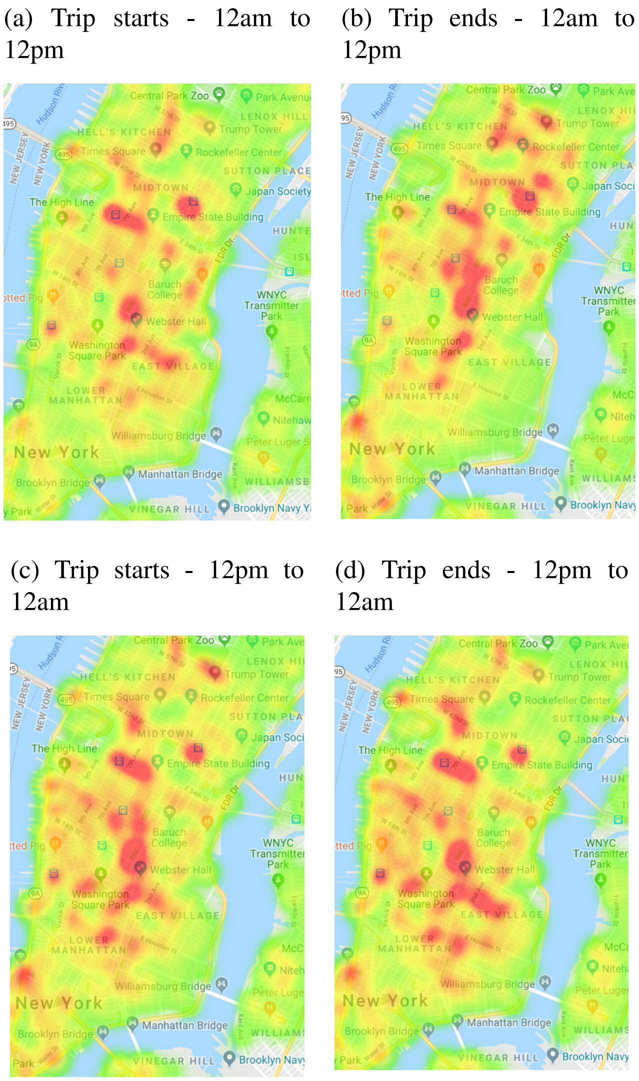
Station	Avg. diff/day	Neighb.
DeKalb Ave & Hudson Ave	37.6	Downtown Brooklyn
E 10 St & Avenue A	31.3	Alphabet City
Old Fulton St	21.8	Fulton Ferry District
...	...	...
Central Park West & W 68 St	-24.9	Central Park
Columbus Ave & W 72 St	-56.8	Lincoln Square
Grand Army Plaza & Central Park S	-59.4	Central Park

Looking at heatmaps of station usage in New York City, we also get a better understanding in how riders use the service. In Figure 1, we can see the activity at various stations, both in the morning and evenings (12 hours each). In all four maps there are three areas which stand out:

- Union Square
- Penn Station
- Grand Central

These areas are highly active throughout the day and have a high number of both trip starts and trip ends. This is intuitive since these locations are commuter hubs where many people would begin or end their commute on bikes.

Fig. 1: Heatmaps over station activity.



(a) Trip starts - 12am to 12pm

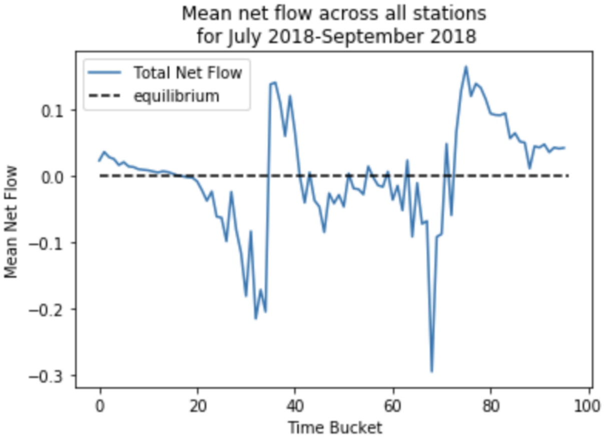
(b) Trip ends - 12am to 12pm

(c) Trip starts - 12pm to 12am

(d) Trip ends - 12pm to 12am

For any given station, the amount of bikes that are remaining for the next customer can vary strongly intra-day. This is because the net difference between bike check-in and check-outs rarely equals to zero, in fact, it is quite volatile throughout the day. In other words, inventory at each bike station does not stay within a tight range on its own. Figure 2 quantifies the problem clearly. During morning and evening rush hours, we see periods of sharp decline in net flow indicating strong demand that can lead to empty stations. As such Citi Bike needs to be prepared to reallocate bikes in advance of these sharp net flow falls in order to keep up with demand.

Fig. 2: Mean net flow (check-ins - check-outs) over all stations for one day. Figure represents one day. See Data Preparation section for details.



A. *Other data sources*

In addition to Citi Bike's historic bike trip data, we found additional features that we think have explanatory power over people's behavior in using Citi Bike's service. Below are the list of other data sources:

- Google Maps API [5] to find geographic neighborhood of each station through reverse geocoding
- Citi Bike API [6] for station capacity which was not available in the trip data
- Historical weather data from NOAA [7] for Central Park, generalized and applied to all stations

III. Data Preparation

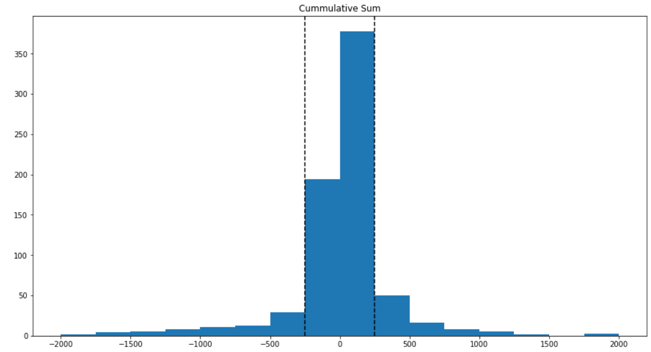
As mentioned above, the raw Citi Bike data includes trip histories across all stations, where each row represents a single bike trip. As we strive to predict demand via which bike stations are ”at risk” of becoming empty and thus need to be restocked, we have significantly transformed the raw data into a new format to suit these purposes.

Recorded in each row of the raw data is the start and end dates and times of each trip (among other features), and we would like to evaluate the time or time periods at which the stock at any particular station is nearing empty. As such, we have grouped individual trips in the original data into 15-minute time buckets per day, where time bucket 0 begins at 12:00 AM and time bucket 95 begins at 11:45 PM. However, after exploring the bike trip durations, it appeared that there were many late night trips that started before midnight and ended after midnight. Therefore, since activity did not halt exactly at 12:00AM, we considered each ”day” as the time frame from 4:00AM (time bucket 16) to 3:59AM (time bucket 15) the following day. Note that the hours of 12:00AM-3:59AM on July 1 were simply removed and the excess hours 12:00AM-3:59AM on October 1 were included. Thus, there are 96 rows per station per day, and each of these rows contains the number of bike check-outs (when a customer removes a bike) and bike check-ins (when a customer returns a bike) per time bucket. We then created a new column, called ”net flow”, which is the number of check-ins minus the number of check-outs that occured within each 15-minute interval.

Additional features were added to the data set to include information regarding station capacity (total number of bike docks available at each station), day of the week (0-6 representing Monday-Sunday), amount of precipitation (0, 1, 2 representing none, low, or high), and temperature (in Celsius) that occurred within that time bucket.

In addition to the aforementioned transformations, we noted that station IDs are randomly assigned numbers that did not provide any useful information regarding the stations; hence using them directly in the machine learning algorithm can lead to erroneous predictions. Thus, we replaced the station ID feature with three new features, so stations can be categorized by the combination of values among these features, namely: neighborhood, net flow variance, and cumulative net flow balance. The neighborhood feature, obtained via Google’s reverse geocoding API [5], is simply the neighborhood in which the station is located. This feature, given it is a string, was one-hot encoded to use in our model. The net flow variance of a station, a continuous variable, is the variance over the aforementioned net flow feature over all 92 days from July-September. The cumulative net flow balance was calculated by observing per station the change in the cumulative sum of the net flow from July 1 to September 30 (see Fig. 3). After plotting these values as a histogram, there appeared to be a significant concentration of stations with cumulative net flow sum within the values of -250 to 250. Therefore, we labeled stations falling within this range as neutral balance (noted with the black vertical lines in the figure), stations less than -250 as low balance, and stations greater than 250 as high balance.

Fig. 3: Histogram of Cumulative Sum of Net Flow



These three features provided a sufficiently unique way to identify the stations in place of using the station ID number. It should be noted that there is possible data leakage that occurs when using cumulative net flow balance and net flow variance as features in our model. This is because both features were calculated using values over all 92 days in the three months, so early time buckets are ”assigned” these values without having seen the data yet to have achieved them.

Finally, our dataset required self-engineered Y values labeling as the raw data from Citi Bike did not provide ’at risk’ flags for periods of high demand on its own. We devised two methods for identifying ’at risk’ time buckets and them. The rationale for having two methodologies is be elaborated in the Model Evaluation part of this report as it affects the performance of various classifiers and our model selection decisions. In this section we outline the difference in labeling methodology.

The Y labels in the first method were calculated by assigning the value 1 to time buckets where (1) the difference between the maximum and minimum of the daily net flow cumulative sum exceeded 90% of that station’s total capacity and (2) the cumulative sum of net flow that occurs at that time bucket is the minimum of the entire day. In order to eliminate the flagging of small, insignificant fluctuations in net flow in condition (2), if there were any repeated occurrences of the minimum value in close sequence, only the first occurrence was labeled 1. An example of this labeling can be found in Figure 4.

The above method resulted in very few Y=1 labels (0.24% of the total data). During our Model Evaluation step, we devised a second Y labeling method to remedy this highly unbalanced classification. As such, in the second method, instead of labeling the single time bucket where the minimum net flow is reached, we flagged all the time buckets in the region between the previous local maximum and the following local minimum. Specifically, the second method finds the intense periods where the net reduction of bikes at a station within a 2.5 hour period is at least 80% of the capacity of each station, and flags all time buckets within this period. This new method of relabeling the Y values resulted in an increased number of 0.68% of data points having label 1. A comparison of the two labeling methodologies can be found in Figures 4 and 5.

Fig. 4: Flagging example for the first method of labeling (red dot indicates Y=1)

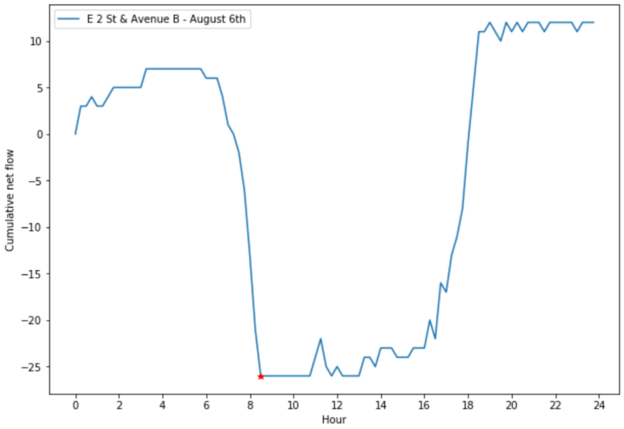
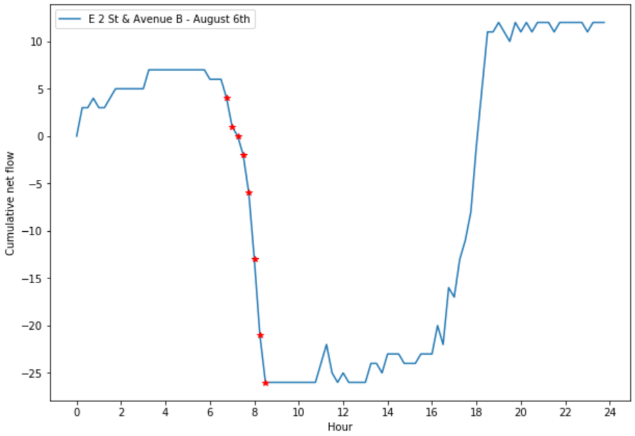


Fig. 5: Flagging example for the first method of labeling (red dot indicates Y=1)



IV. Modeling & Evaluation

Our model evaluation entails several important steps in order to tackle the complexity of our dataset. Below are the key points of complexities and the required decisions we made during our model evaluation:

- Mixture of numeric and categorical features - Logisitc Regression or Decision Tree classifer
- Highly imbalanced dataset - percentage of up-sampling of Y=1 labels
- Self-engineered Y labeling - method 1 or method 2 of Y-labeling

To counter the effects of having a highly imbalanced data set, we up-sampled the instances of Y=1 in our training data during the model validation step. We also ran our classifier model on the dataset with the second method of Y-value labeling so that the dataset is slightly less imbalanced.

After determining the best up-sampling percentage, we then conducted a separate cross validation to tune the hyper parameters to further improve our model performance. The detailed steps of our process are outlined below.

A. *Evaluation Metrics*

We evaluate our model's performance based on the combination of recall (or true positive rate), precision and F1 Scores. Because our data is highly imbalanced, accuracy scores alone would be very misleading. Furthermore, we want to show the trade-off between recall and precision as management can use that information, combined with their knowledge of the cost of dispatching Citi Bike restocking teams and of the potential profit loss from missed demand, to make more intelligent business decisions.

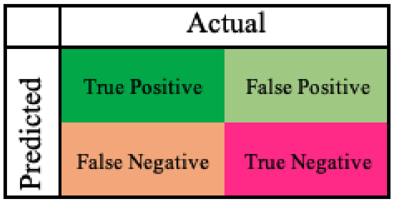
For example, if cost of unmet demand is high, then management would want a model that offers high recall. On the other hand, if the wasted operational costs of dispatching restocking teams is high, then management would want a model that favors high precision.

That said, without economic information, we use the F1 score, which is the harmonic mean of recall and precision, to select the optimal up-sampling percentage (F1 score is a balanced metric between recall and precision).

Note that we are not emphasizing the false positive rate metric in this analysis. Because our data is highly imbalanced, the false positive rate is always very low; hence there is little meaning in examining that metric.

To clarify, below are a few legends to help the reader understand this section.

Fig. 6: Confusion Matrix Legend



	Actual	
Predicted	True Positive	False Positive
	False Negative	True Negative

Definitions of Evaluation Metrics:

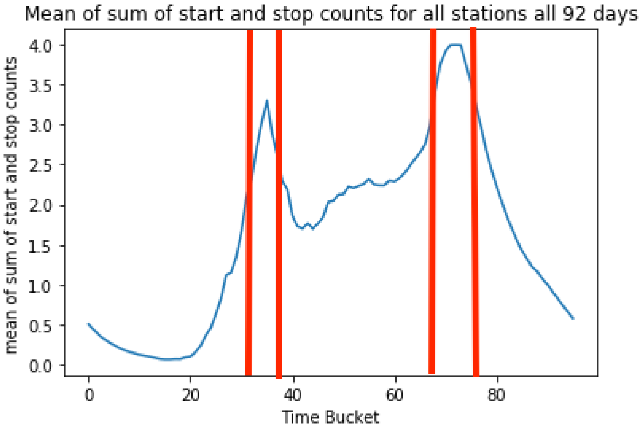
- Recall or TPR or Sensitivity = TP/(TP + FN)
- Precision = TP/(TP+FP)
- F1 = (2 x recall x precision) / (recall + precision)
- FPR or 1-specificity = FP/(FP+TN)

B. Baseline Model and Base Rate

For our Baseline Model, we believe Citi Bike's management team would most likely flag time buckets to be "at risk" using simple business heuristics.

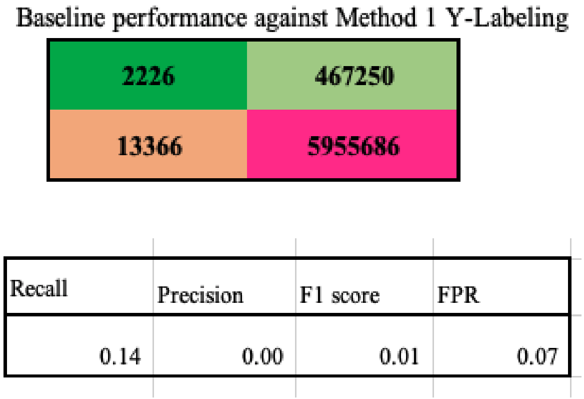
As we know Citi Bike tracks the traffic of all bike stations, we can use a simple metric - the sum of bike check-in and check-out counts for each time bucket and take the average of that metric across all stations for the entire July 1st to September 30th period. Per Figure 7, this shows that traffic activity, measured by the sum of check-in and check-out peaks between the hours of 8:30AM to 9:00AM and 5:30PM to 6:15PM (time buckets 34, 35, 36, 70, 71, 72, and 73).

Fig. 7: Baseline Model: Selecting High Traffic Time Buckets based on Historic Activity



Therefore a reasonable baseline prediction for Citi Bike management would be to simply flag those time buckets at any station on any day as "at risk" periods. We calculated the evaluation metrics for that baseline prediction first against our method 1 Y-labels (we later calculated them against our method 2 Y-labels). These metrics are used as benchmarks for model parameter fine-tuning.

Fig. 8



Baseline performance against Method 1 Y-Labeling

2226	467250
13366	5955686

Recall	Precision	F1 score	FPR
0.14	0.00	0.01	0.07

C. *Model Selection*

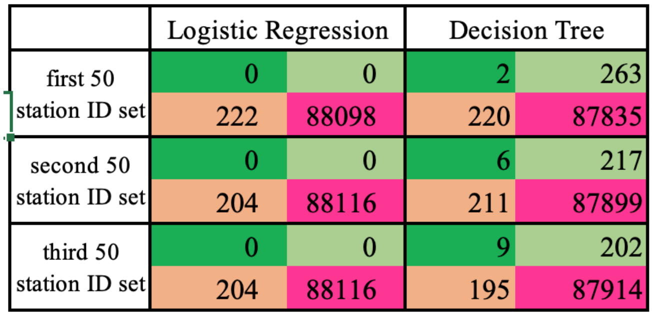
For model selection, we considered two types of machine learning models: Logistic Regression and Decision Tree. We decided to begin with Logistics Regression as it is often used as a starting model for binary classification problems. Additionally, this appeared to be a logical choice since the number of data points (n = 6,438,528) far exceeded the number of features (m = 89).

Likewise, we decided to use Decision Tree as our second model since Decision Tree is able to consider relationships across dependent features to determine an instance's class. For example, if a bike station is located in a neighborhood that has high traffic all the time, then the day of the week may not be as important in determining which time bucket will be at risk. However, for a station located in Midtown East, where many people commute to for work, the day of the week would be an important feature because of differing rider behavior between weekdays and weekends.

To systematically determine which model is best suited for our analysis, we decided to use 3 random subsets of 50 stations each as a preliminary check of model performance. Whichever model resulted in better performance we would then fine tune using cross validation on the entire dataset. Below are the results for each of the three subsets. Note that each subset has 353,280 instances in its training set and 88,320 instances in its testing set.

Looking at Figure 9, Decision Tree was the better model because Logistic Regression was not able to identify any class=1 instances. Logistic Regression most likely did poorly because our data consists of a number of categorical features that may have been difficult for Logistic Regression to predict.

Fig. 9: Comparison of Logistic Regression and Decision Tree Performance on Subsets of 50 Station IDs



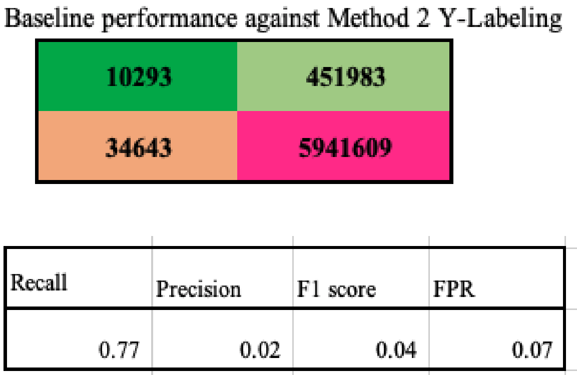
	Logistic Regression		Decision Tree	
first 50 station ID set	0	0	2	263
	222	88098	220	87835
second 50 station ID set	0	0	6	217
	204	88116	211	87899
third 50 station ID set	0	0	9	202
	204	88116	195	87914

The results from our preliminary model selection analysis also underscored the effect of our highly imbalanced Y-labels. This led us to create a second labeling method to better balance our data without compromising the true occurrences of ”at risk” time buckets.

The second method of labeling the Y values resulted in the percentage of positive classes increasing from 0.24%, to 0.68%. In addition, method 2 gives a more robust flag because it highlights a period of continuous ”at risk” time buckets as opposed to the last time bucket of an ”at risk” period. Refer to Figures 4 and 5 in the Data Preparation section for a visualization of this difference.

For good measure, we also re-computed the baseline model performance using our second method of Y-labeling. We used these metrics as benchmarks for model improvement going forward. Note that recall for the baseline model using this alternative method of Y-labeling is much higher (0.77), yet precision continues to be very poor (∼0.02).

Fig. 10



Baseline performance against Method 2 Y-Labeling

10293	451983
34643	5941609

Recall	Precision	F1 score	FPR
0.77	0.02	0.04	0.07

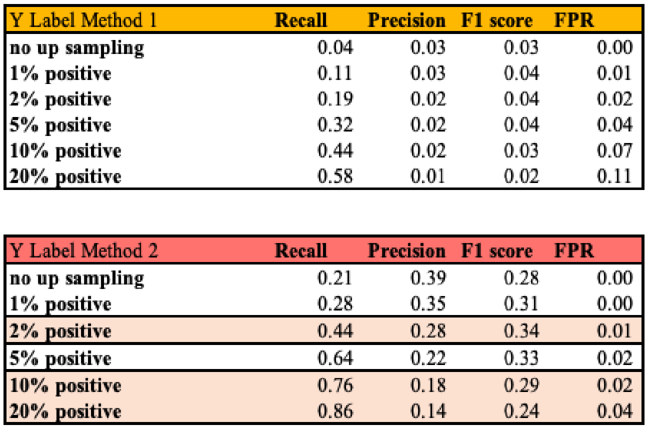
D. *Validation - Up-sampling*

Given the results from our model selection, where we consistently have very low precision, we decided that we need to employ up-sampling, in other words, increase the number of Y=1 labels in our training set, in order to more precisely identify ”at risk” time buckets. To determine the best amount of up-sampling, we conducted a number of validation tests where the amount of Y=1 labels comprised 1%, 2%, 5%, 10% and 20% of the training set. We first split our data into the standard 80% training and 20% testing and adjusted the amount of Y=0 labels to render the desired Y=1 percentage. Then, we validated this up-sampled training set on the performance on the original testing set.

In Figure. 11 we provide the performance of each up-sampled percentage, as well as side-by-side comparison of the two techniques of Y-label. It is clear that method 2 of Y-labeling offers a more balanced outcome between recall and precision, particularly in improving precision. If we compare these results to the baseline model results, precision improved from a consistent ∼ 0.02 to between 0.14-0.39).

Using the F1 score, the best performance came from 2% up-sampling with the Decision Tree classifier. That said, up-sampling percentage can also be chosen based on Citi Bike management’s preference in the trade-off between recall and precision. For example, in Figure 11, a 10% or larger up-sampling can be used if management wants recall to be greater than or equal to the baseline recall of 0.77.

Fig. 11: Comparison of Decision Tree Performance across Both Labeling Methodologies



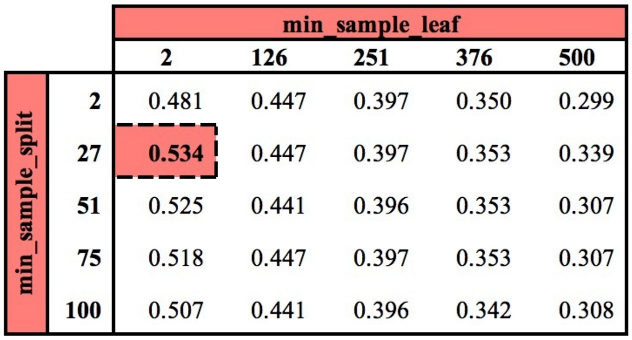
Y Label Method 1	Recall	Precision	F1 score	FPR
no up sampling	0.04	0.03	0.03	0.00
1% positive	0.11	0.03	0.04	0.01
2% positive	0.19	0.02	0.04	0.02
5% positive	0.32	0.02	0.04	0.04
10% positive	0.44	0.02	0.03	0.07
20% positive	0.58	0.01	0.02	0.11

Y Label Method 2	Recall	Precision	F1 score	FPR
no up sampling	0.21	0.39	0.28	0.00
1% positive	0.28	0.35	0.31	0.00
2% positive	0.44	0.28	0.34	0.01
5% positive	0.64	0.22	0.33	0.02
10% positive	0.76	0.18	0.29	0.02
20% positive	0.86	0.14	0.24	0.04

E. *Hyper-parameter Tuning*

To find the best parameters for the Decision Tree model, we did a grid search to find the optimal parameters *min_sample_leaf* and *min_sample_split*. The results can be found in Figure 12. For each parameter value pair, we did 5-fold cross validation on a training set up-sampled to 2% Y=1 instances. In doing so, we found that a *min_sample_leaf* value of 2 and *min_sample_split* of 27 gave the best F1 score of 0.534.

Fig. 12: Table of grid-search F1 scores for the decision tree model.



		min_sample_leaf				
		2	126	251	376	500
min_sample_split	2	0.481	0.447	0.397	0.350	0.299
	27	**0.534**	0.447	0.397	0.353	0.339
	51	0.525	0.441	0.396	0.353	0.307
	75	0.518	0.447	0.397	0.353	0.307
	100	0.507	0.441	0.396	0.342	0.308

V. Deployment

This model should be deployed as a web based application to be used by Citi Bike's inventory rebalancing team. Every 24 hours the application would re-evaluate the risk status of each station and inform the rebalancing team when outages are to occur. The application could then be adjusted to be based on historical rolling data so that as the rebalancing team improves its performance and eliminates bike shortages, the instances of Y=1 predicted by the model would decrease to zero. As new stations are introduced, the application would also be helpful in predicting bike shortage behavior compared to the baseline heuristic we used in model evaluation.

To evaluate the robustness of the model, it should be monitored over extreme periods, such as days with very high temperatures or high volumes of precipitation, to confirm that it is still accurately predicting outage periods. Other events to stress test include traffic altering events, such as the UN assembly or holiday parades.

Two potential risks involve the trade-off between the true positive rate and false positive rate of the model. If the true positive rate is too low, the model runs the risk of under predicting occurrences of bike shortages, leading to increased customer dissatisfaction from empty stations and potential revenue loss. If the false positive rate is too high, the model then runs the risk of mislabelling stocked stations as out of stock, resulting in pointless deployments of the rebalancing team and unnecessarily incurring associated costs. Ultimately, the right balance of the true positive rate and false positive rate depends on the cost of dispatching the rebalancing team versus the potential revenue loss from dissatisfied customers. If we obtain economic information from Citi Bike's management team, we can use that to fine tune our model parameters to maximize economic value. If the cost of dispatch outweighs the potential revenue loss, the model should be adjusted to decrease the false positive rate as to minimize incorrect dispatches. If the potential revenue loss is greater than the cost of dispatch, the model should be adjusted to increase the true positive rate as to maximize customer satisfaction.

Ethically, so long as the identity of the riders are kept private and no stations are unfairly favored over others, the deployment of the solution should not cause any issues.

References

[1] Citi Bike September 2018 Operating Report
https://d21xlh2maitm24.cloudfront.net/nyc/September-2018-Citi-Bike-Monthly-Report.pdf

[2] Citi Bike expansion plans
https://www.citibikenyc.com/blog/citi-bike-is-going-to-dramatically-expand

[3] Bike shortage article
https://nyc.streetsblog.org/2018/09/26/its-not-your-imagination-something-is-seriously-wrong-with-citi-bike-right-now/

[4] Trip data source — Citi Bike. Retrieved October 25, 2018, from
https://s3.amazonaws.com/tripdata/index.html

[5] Geocoding API — Google Developers. Retrieved November 30, 2018, from
https://developers.google.com/maps/documentation/geocoding/

[6] Station Information — Citi Bike. Retrieved November 27, 2018, from
https://gbfs.citibikenyc.com/gbfs/en/station_information.json

[7] National Oceanic and Atmospheric Administration. Historic Weather Data. Retrieved November 20, 2018, from https://www7.ncdc.noaa.gov/CDO/cdodataelem.cmd