Predicting Capital Bikeshare Usage

Cynthia E. Cho

Abstract

Bike sharing programs are increasingly gaining momentum in large cities across the U.S. With the increase of bike flow and usage, predicting bike demand has become a focus in recent studies and the relevant factors that impact usage; predicting bike demand is a part of a larger effort to facilitate efficient bike rebalancing and expansion of docking stations. This project looks at different temporal and weather features to isolate key features that contribute to hourly bike demand and uses three tree-based approaches to build parsimonious models to predict demand. The findings of this project identify a subset of features to build the models and demonstrate that while some algorithms may have more complexity to them, they may perform comparably against less complex models. In addition to comparing models through performance scores, the importance of factoring in runtime is also addressed.

1. Introduction

The largest bike sharing program in the United States at the time started in the District of Columbia on September 2010 by Capital Bikeshare (CaBi). The bike sharing program served several neighboring cities with 500 stations and 4,300 bicycles. Bike sharing programs have been a growing program in the large cities across various countries because it promotes mobility and flexibility, an eco-friendly and healthy alternative to automotive transportation, reduction in traffic, and provides greater accessibility to surrounding areas for users. They have essentially become a part of the transportation network in large cities.

With the increase of bike usage, there is also the need to address the issue of bike demand and understanding what drives that demand. This is an important focus as it is a function of another problem, one that will not be addressed in this project, but the challenge of effective and efficient rebalancing of bike docking-stations. A common problem with the increase in bike sharing programs and use is the issue of empty or full bike docking-stations, which can be solved through understanding the factors that impact bike usage. Using those factors to predict demand will help facilitate rebalancing to meet demand.

The objective of this project is to observe the impact of important features on bike demand and using those features to forecast the number of bikes that are going to be rented every hour with supervised learning techniques. To perform the project, three tree-based approaches are deployed: Decision Trees, Random Forest, and Gradient Boosting and enhancing the models through hyper-parameter tuning to ensure that the models are producing optimal results given their constraints. The parsimonious models will be compared against their baseline counterpart models that use all features. A final comparison will be done between the final parsimonious models to see if there is any difference between the approaches through performance results. The comparison is not necessarily to choose a "best" model, but to observe comparability and possible tradeoffs that take exist when using various techniques.

2. Literature Review

The increase in popularity for bike sharing programs and the large generation of historical trip data has become an interest in the transportation community. Studies have been done and will continue to be done on assessing the important factors that affect the flow and usage of bike sharing programs as utilization increases.

Various studies on assessing the factors that impact the flow and usage of bike sharing systems have looked at weather features such as temperature, humidity, and weather quality/condition. Gebhart and Noland (2014) site the importance of weather factors such as humidity and temperature on the Capital Bikeshare system in Washington D.C. and conclude that bike trips are affected primarily by cold temperature, rain, and high humidity. Temporal features such as time of day, working day versus non-working day, land-use such as universities, retail space, restaurants, and other commercial enterprises influence bike flow and usage (Imani et al., 2014). Research by Buck and Buehler (2012) have also done a study on the Capital Bikeshare program and conclude that time, population, and bike demand is positively and statistically significant in bike utilization stating that for every 1,000 people, there is an additional 0.5 rides per day. In the article, *Predicting Bike Usage for New York City's Bike Sharing System*, Singhvi et al. (2015) used taxi flow and usage, time, weather, and travel patterns between neighborhoods during morning rush hour to predict weekday morning rush hour bike demand. To provide some granularity and some precision, they aggregate the demand by neighborhood to make their bike usage prediction.

In addition to looking at factors that drive demand, studies have also been done in conjunction to predict new locations and using time and travel patterns between neighborhood locations (Singhvi et al., 2015). Additional studies have supported similar conclusions on predicting new location rather than expanding bike docks in existing locations based on surrounding land-use (Imani et al., 2014). With many prediction models being created, the next approach using flow and usage prediction has been to focus on the optimization of bike sharing with a focus on rebalancing efforts. In the paper by Mahony & Shmoys (2015), their research uses New York City's Citibike data to analyze demand by focusing on a specific time frame (rush hour) and derive potential solutions for bike rebalancing during mid rush-hour and address to solve the routing challenges associated with traffic when trying to rebalance bike docks during mid rush-hour use. With the abundance of historical bike trip data being generated as the number of users increase, studies have been done to predict flow and usage using predictions to enhance bike sharing programs. The objective of this project is similar to the studies done in the past and will look at the impact of the temporal and weather characteristics on bike demand.

3. Methods and Data

3.1 Data and Preprocessing

The dataset is from the UCI Machine Learning Repository; it is an aggregation of trips by hour of the Capital Bikeshare trips made during the years 2011 and 2012 and has 17,393 instances. The dataset contains various weather and temporal features. The dependent variable is the count - the number of bikes rented in an hour within a 24-hour period. From the exploratory analysis, a couple of important observations worth mentioning is the detection of outliers and the importance of the temporal feature, hour. Because the program started in September 2010, and the dataset contains observations from 2011 to 2012, high value count between 700-977 were identified as

outliers. However, they were not removed since it was not movement observed during a single day but fluctuations that took place over the year as the program gained more users. The hour feature identified distinguishing patterns between working days and non-working days and the types of user. What was extrapolated is a large portion of users are daily commuters and use is heaviest during working days.

During the preprocessing stage, using a correlation matrix, highly correlated features were removed. For instance, season was highly correlated with months with a positive correlation value of 0.85 and therefore removed. The choice to keep month instead of season came down to the granular detail that months provide for bike usage in comparison to months where all three months within a season provided no variability. The categorical features are as follows: working day, holiday, and weather condition, where working day and holiday are binary values, and weather condition has multi-values to represent the different weather conditions that bikes were utilized. The two binary categorical features working day and holiday were factorized. Rather than converting the weather condition feature into dummy variables, it was converted to an ordinal scale using frequency values for each category: 1 – Clear, Few clouds, Partly cloudy, Partly cloudy, 2 – Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist, 3 – Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain, Scattered clouds, and 4 – Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog. The continuous features are as follows: hour, temperature (in Fahrenheit), year, humidity, month, weekday, and wind speed. The values span from 1 to 977. The training/testing and validation was done using an 80/20 split.

3.2 Feature Selection

During the feature selection process, three feature selection methods were utilized to finalize the features that would be making it into the final model: the Recursive Feature Elimination (RFE) method, Filter Method – Univariate Selection (FUS) method, and Feature Importance method using Random Forest (FIRF) and Gradient Boosting (FIGB). The Recursive Feature Elimination method is a greedy search that works to find the best subset of features by recursively removing them and builds a model with the remaining features. The Filter Method uses the mutual information regression and measures the dependence between the features and provides scores ranking from highest to lowest. The last feature selection method is performed using feature importance from the two ensemble methods: Random Forests and Gradient Boosting. The feature importance is calculated for each method based on how much each feature contributed in the construction of the model. For instance, the more a feature was used in the Gradient Boosting iterations, the higher its importance relative to the other variable within the model. The three feature selection methods were applied due to variability in the methods and feature selection to see if there was any consistency in the selected features.

From Table 1., the RFE method and FIGB selected the same features with only a difference of ordering between working day and year. The FUS and FIRF methods shows the most variability in the importance of the features added to the model. The difference comes from the following features: weekday, wind speed, working day, holiday, and weather condition. Based on the frequency of low placement in the feature selection process relative, the features wind speed, weekday, and holiday were removed. For instance, weekday may have been the seventh importance feature according to the FIRF method, but its constant low inclusion in the other three methods, the feature was removed. The same thought process was used to remove wind speed and holiday from the feature set.

Table 1. Feature Selection

Recursive Feature Elimination (RFE) Method	Filter (FUS) Method	Feature Selection	
		Random Forest (FIRF) Method	Gradient Boosting (FIGB) Method
hour	hour	hour	hour
temperature	temperature	temperature	temperature
year	humidity	year	working day
working day	month	working day	year
humidity	year	humidity	humidity
month	wind speed	month	month
weather condition	weather condition	weekday	weather condition
weekday	working day	weather condition	weekday
wind speed	weekday	wind speed	wind speed
holiday	holiday	holiday	holiday

The final subset of features are hour, temperature, working day, year, humidity, month, and weather condition were incorporated into the model.

3.3 Regression Methods and Model Generation

Three tree based supervised machine learning techniques are deployed on the dataset to apply regression methods. The purpose of using the three different methods was to compare the performance of the approaches on the dataset based on each method's model building process. Decision Trees are a single iteration process where the tree is built starting from the root node. The tree is created through recursive partitioning based on a feature's value using a selection criterion. In this project, the criterion is the mean squared error ("mse"), which will be used for both Decision Trees and Random Forest. Random Forest builds many trees to create a "forest" or an ensemble of trees through bootstrap aggregation where the nodes are split based on different subset of the feature space. The trees are created in parallel to collectively gather information and predictions are based on the votes of all the trees, unlike the decision tree. On the other hand, Gradient Boosting is a stage-wise iterative process that seeks to minimize a loss function using regression trees. The prediction model is created by forming an ensemble of weak learners and turning them into strong learners through the iterative process of learning from the errors of the previous iteration. The technique uses a weighted voting scheme on the difficult weak learners to help with the learning process and by uses gradient descent to decide on when and how to add new learners to the ensemble. Gradient Boosting trains models through an additive and sequential process. The project is done using Scikit-learn, a free software machine leaning library for Python. The algorithms used are the Decision Tree regressor, Random Forest regressor and Gradient Boosting regressor.

Prior to running feature selection, a full feature model (model I) was created using the Decision Trees Regressor, Random Forests Regressor, and Gradient Boosting Regressor with a 5-fold cross-validation process as baseline models. After feature selection, using the selected subset of features and a 5-fold cross validation grid search to find the optimal hyperparameters for each model, and model II (the final model) was generated. The purpose behind model I and model II is to compare the performance between a model using all its features and to see if a parsimonious

model can be created using less features to achieve comparable results with the added benefit of fine-tuning the model. Hyper-parameter tuning is performed on model II because the hyper-parameters control the learning process and need to be fine-tuned so that using the constraints and measures, the machine learning model can optimally address the problem. To tune the hyper-parameters of the models, GridSearchCV, a module within Scikit-learn was used. Table 2. shows the resulting hyper-parameters used in model II to achieve the final results seen in model II.

Table 2: Hyper-parameters for the final model

Decision Tree:	minimum samples for node split: 31, minimum samples leaf: 5
Random Forest:	number of trees: 1,000; criteron: 'mse'; max number of features for a split:0.75; minimum samples for node split: 7, minimum samples leaf: 1
Gradient Boosting:	number of trees: 1,000; learning rate: 0.2, loss function: least squares regression; minimum samples for node split: 5; minimum samples leaf: 1

Post feature selection and grid search, model II with the three regressors is trained on the full training set and then applied to the validation set. To evaluate performance, Root Mean Square Error (RMSE) and Explained Variance scores are generated.

4. Results/Discussion

Table 3.

		RMSE	Standard Deviaion (+/-)	Explained Variance	Standard Deviaion (+/-)	Runtime
		Training: 5-fold cross validation				
Model I: baseline model (full feature set)	Decision Tree	60.52	5.07	0.89	0.01	0.23363
	Random Forest	55.78	1.70	0.91	0.00	6.56173
	Gradient Boosting	73.16	4.98	0.84	0.01	1.97144

		RMSE	Standard Deviaion (+/-)	Explained Variance	Standard Deviaion (+/-)	Runtime
		Training: 5-fold cross validation (post gridsearch)				
Model II: final model (subset features)	Decision Tree	56.41	3.8	0.90	0.01	0.14086
	Random Forest	48.67	2.33	0.93	0.01	295.05516
	Gradient Boosting	51.21	0.64	0.92	0.01	24.59064
		Validation Set				
Model II: final model (subset features)	Decision Tree	55.93	-	0.91	-	0.02206
	Random Forest	46.50	-	0.94	-	13.62939
	Gradient Boosting	50.78	-	0.92	-	2.98797

** Standard deviation of dependent variable, count, is 188.31.

One thing to notes is the challenge of understanding an appropriate RMSE score. A guideline that was used in assessing whether the RMSE is acceptable or not was done by comparing it against the standard deviation of the dependent variable, 188.31. As it can be seen from table 3., all RMSEs are below the 188.31.

The final variables in model II's regressors are year, hour, month, working day, temperature, humidity, and weather condition. The results are shown for both the 5-fold cross-validation and the validation set on Table 3.

Comparing the training scores achieved from model II regressors against model I regressors, the model II Decision Tree performance scores are within the range of the scores achieved in model I Decision Tree. On the other hand, the RMSE scores for Random Forest and Gradient Boosting in model II are lower than the respective scores achieved on model I and fall outside the range of variation; the explained variance scores for the two regressors are also outside the range of variation in comparison to those achieved in model I, and are higher. What this indicates is that aside from the Decision Tree regressor, Random Forest and Gradient Boosting performance scores are not comparable as they are outside of the range of variation. There is also a large decrease in RMSE and increase in the Explained Variance performance scores, which supports the importance of fine-tuning the model parameters. Using less features and tuned hyper-parameters, the models are within the range of variation or performed slightly better than their baseline counterparts. When comparing the regressors in model II cross-validated scores against each other, with the exception of the Decision Tree regressor's RMSE, the RMSE and explained variance performance scores are comparable.

The trained model is applied to a separate validation set of 20% using the parsimonious model II. From table 3., the RMSE scores for the Decision Tree Regressor, Random Forest Regressor, and Gradient Boosting Regressor are 55.93, 46.50, and 50.78, respectively; the explained variance scores are 0.91, 0.94, and 0.92, respectively. The scores achieved on the validation fall within the range of variation of the regressors seen in model II 5-fold cross-validation. What the scores from the regressors in model II for both the cross-validation and validation set indicate is that comparable or slightly better performance results can be achieved with fewer features to produce a parsimonious model. Moreover, the scores between the regressors indicate that while Random Forest and Gradient Boosting are comparable for both performance scores, higher complexity of technique does not indicate that it will outperform a "simpler" model. While performance scores are an important part in comparing different machine learning techniques, there is a caveat of computation efficiency. Even with comparable performance results between Random Forest and Gradient Boosting, the runtime when performing 5-fold cross-validation demonstrates that there is a difference in computation efficiency. Model II's Random Forest runtime took 295.05516 seconds, which roughly translates to five minutes compared to the runtime of 24.59064 seconds for Gradient Boosting. While five minutes may not be that significant, the increase in runtime will be amplified with the size of the dataset. The takeaway from this project is that of the three tree-based methods, Random Forest and Gradient Boosting were comparable in both performance scores and complexity of the technique does not necessarily indicate higher performance results. Additionally, runtime is an important component to take into consideration as it can be seen from Table 3.

5. Limitations and Future Work

A limitation that was faced during the project was the hourly aggregation of the entire dataset and the lack of bike docking-station information. Overall, there was a lack of granularity with the dataset to extrapolate pattern on ridership and the demand for bike usage.

For future work, using a historical trip dataset with the origin and destination of trip data can be used to conduct deeper analysis on the origin-destination pairs. The inclusion of features such

as land-use, special events, proximity to other transportation systems, bike path, GPS data, and neighborhood/population density can also provide further insight into what drives bike demand and where there is heavier bike usage. By performing bike demand prediction and a deeper analysis using additional features, future research on bike rebalancing and new location development can be done more efficiently.

6. Conclusion

The objective of this project was to identify key features to predict the number of bikes that are going to be rented every hour using three tree-based regressor: Decision Tree, Random Forest, and Gradient Boosting. What this project has shown is that with less features comparable results can be achieved against a full feature model. And while there were some mixed results with the Decision Tree and the other regressors in the model II, Random Forest and Gradient Boosting performed comparably in both performance scores. However, there was a tradeoff when factoring in computation efficiency and computational power is an important component of machine learning that needs to be taken into consideration.

Overall, the process of doing this project provided insight into the tree-based techniques that were explored and why trying different approaches is beneficial when looking to solve any machine learning problem, whether it be a regression method or a classification method. Because the "belief" of one technique being superior may not hold true. The key takeaway with this project is the important of exploring the available techniques at our disposal and factor in how computation efficiency when choosing one algorithm over another.

As far bike sharing programs and studies related to the topic, with the abundance of data being generated from the program, deeper analysis on trends should be evaluated to see how they can help with predicting demand. There is still a lot of work that needs to be done and will continue to evolve as users increase and people look for alternative means of transportation. An interesting study that can be done regarding bike sharing programs is whether these programs truly observe any health benefits from the users as researchers claim in their studies.

References

Buck, D., & Buehler, R. (2012, January). Bike lanes and other determinants of capital bikeshare trips. In *91st Transportation research board annual meeting*.

Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [https://archive.ics.uci.edu/ml/datasets/bike+sharing+dataset]. Irvine, CA: University of California, School of Information and Computer Science.

Faghih-Imani, A., Eluru, N., El-Geneidy, A. M., Rabbat, M., & Haq, U. (2014). How land-use and urban form impact bicycle flows: evidence from the bicycle-sharing system (BIXI) in Montreal. *Journal of Transport Geography*, *41*, 306-314.

Gebhart, K., & Noland, R. B. (2014). The impact of weather conditions on bikeshare trips in Washington, DC. *Transportation*, *41*(6), 1205-1225.

O'Mahony, E., & Shmoys, D. B. (2015, February). Data analysis and optimization for (citi) bike sharing. In *Twenty-ninth AAAI conference on artificial intelligence*.

Singhvi, D., Singhvi, S., Frazier, P. I., Henderson, S. G., O'Mahony, E., Shmoys, D. B., & Woodard,
D. B. (2015, April). Predicting bike usage for new york city's bike sharing system. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*.