I. definition

Project Overview

Bicycle sharing platforms provides renting and returning bicycles services as needed basis. The platforms allow users to rent and return bicycle through a network of kiosk locations accessible over the entire city. These services are now available with a mobile phone application, meaning that user can locate the nearest available bicycle and rent it with a couple of tap on mobile phone application.

Mobile devices records information such as the duration of travel, departure location, arrival location, and time elapsed. These allow monitoring the condition of bicycles(if wheels wear out), studying mobility in a city, predicting the demand of bicycle during the time of a day.[1,2] Therefore, the data can be utilized to help improving company's services towards users, such as meeting the demand of usage by supplying optimal amount of bicycles, reducing possible accident due to the faulty brake system or worn out wheels, and setting up new service locations to cater the services to region where it was historical unavailable. Above problems can be addressed via getting insightful information by analyzing the existing data. To develop a prototype model, I took the bike sharing data from a Kaggle competition(Bike Sharing Demand, https://www.kaggle.com/c/bike-sharing-demand) for this work. This project will provide insights on the factors that affect the demand of rental bike in Washington, D.C.

Problem Statement

The project aims to utilize historical usage patterns with weather data to forecast bike rental demand in the Capital Bikeshare program in Washington, D.C. The inputs consist categorical (e.g. weather, season, etc) and numerical data (e.g. temperature, windspeed, etc). The output, count of rental bike, is a numerical output, therefore the project is a regression problem.

Metrics

Root Mean Squared Logarithmic Error (RMSLE) will be used to evaluate this project. The RMSLE is calculated as

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(p_i+1)-\log(a_i+1))^2}$$

Where:

- n is the number of hours in the test set
- p_i is the predicted count
- a_i is the actual count
- $\log(x)$ is the natural logarithm

RMSLE is basically the logarithmic version of mean squared error. In this case, it provides a mean squared error of the logarithmic scale of the output. This provides errors relatively to the order of magnitude. Hence, the error represents the mismatch on a relatively same scale.

II. Analysis

Data Exploration & Visualization

The data is obtained from Kaggle competition site. Hourly rental data spanning two years are provided; the training set is comprised of the first 19 days of each month, while the test set is the 20th to the end of the month. Below are the data fields provided:

- datetime - hourly date + timestamp
- season - 1 = spring, 2 = summer, 3 = fall, 4 = winter
- holiday - whether the day is considered a holiday
- workingday - whether the day is neither a weekend nor holiday
- weather –

  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

4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

- temp - temperature in Celsius
- atemp - "feels like" temperature in Celsius
- humidity - relative humidity
- windspeed - wind speed
- casual - number of non-registered user rentals initiated
- registered - number of registered user rentals initiated
- count - number of total rentals

Since the data is recorded in hourly basis, the "datetime" can be splited to "date" and "hour". We also derive other useful features such as "weekday", "month", "year", "day".

First, we explore the (mean) daily rental count of training data(Figure 1). It appears that there are higher demands around June to August, while a lower demand around December to February. Overall, the counts are higher in year 2012 than 2011, suggesting that there is a growth in bike sharing business.

Next, we explore the hourly demand on weekday basis(Figure 2). On weekdays (Monday to Friday), there are two slight peaks around 7-8 and 17-18 hours, a feature that is not observed on the weekends (Saturday to Sunday). This corresponds to the starting and ending of office and school hours, suggesting that the high amount of bike rental in these hours are like due to mobilities of students and office workers from home to their respective destinations (schools, offices), or vice versa. On weekends, the bike demand features a hump from around 8 to 21 hours; the same trend can be roughly seen in the weekdays between both peaks (around 7-8 and 17-18 hours). As expected, this suggests that most bike rental users are getting active during the day time. Besides, there is also a noticeable demand in the midnight hours (0-2 hours), in which we can relate to the users that work on night shift in convenient shops etc.

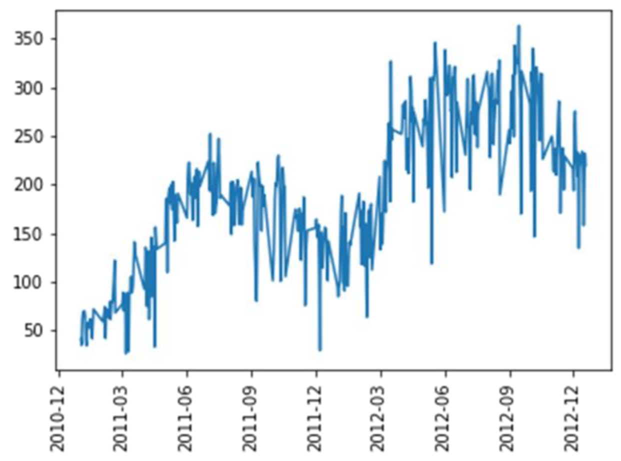

Figure 1: Mean of rental count of training data set.

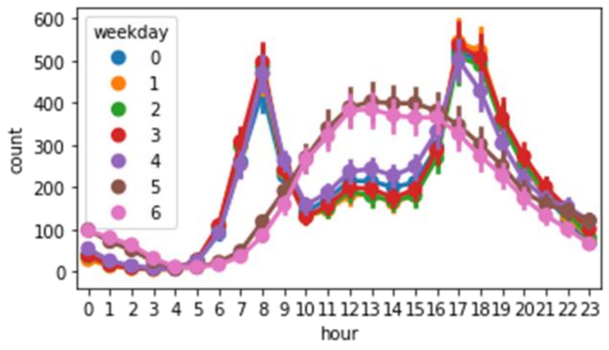

Figure 2: Hourly demand of bike on weekdays. 0-6 represent Monday-Sunday.

Figure 3 shows the hourly demand on holidays. It is notable that on April 15-16 (DC Emancipation Day), October 8-10(Columbus Day), and November 11-12 (Veterans Day), there are two slight peaks around 7-8 and 17-18 hours, similar to the bike demand in normal working day(Figure 2, Monday-Friday). This might suggest that these three holidays might not be heavily celebrated by everyone, some businesses (restaurants, retail shops, etc) might operate as usual and some employees might have to work. It should be noted that on National Day(July 4th), the bike rental has high demand over a wider time period (9am -10pm hour) as compared to weekends(11am-5pm) and other holidays(9am-7pm). On average, the counts are higher in year 2012 than 2011, suggesting that there is a growth in bike sharing business, a trend that was also observed in Figure 1.

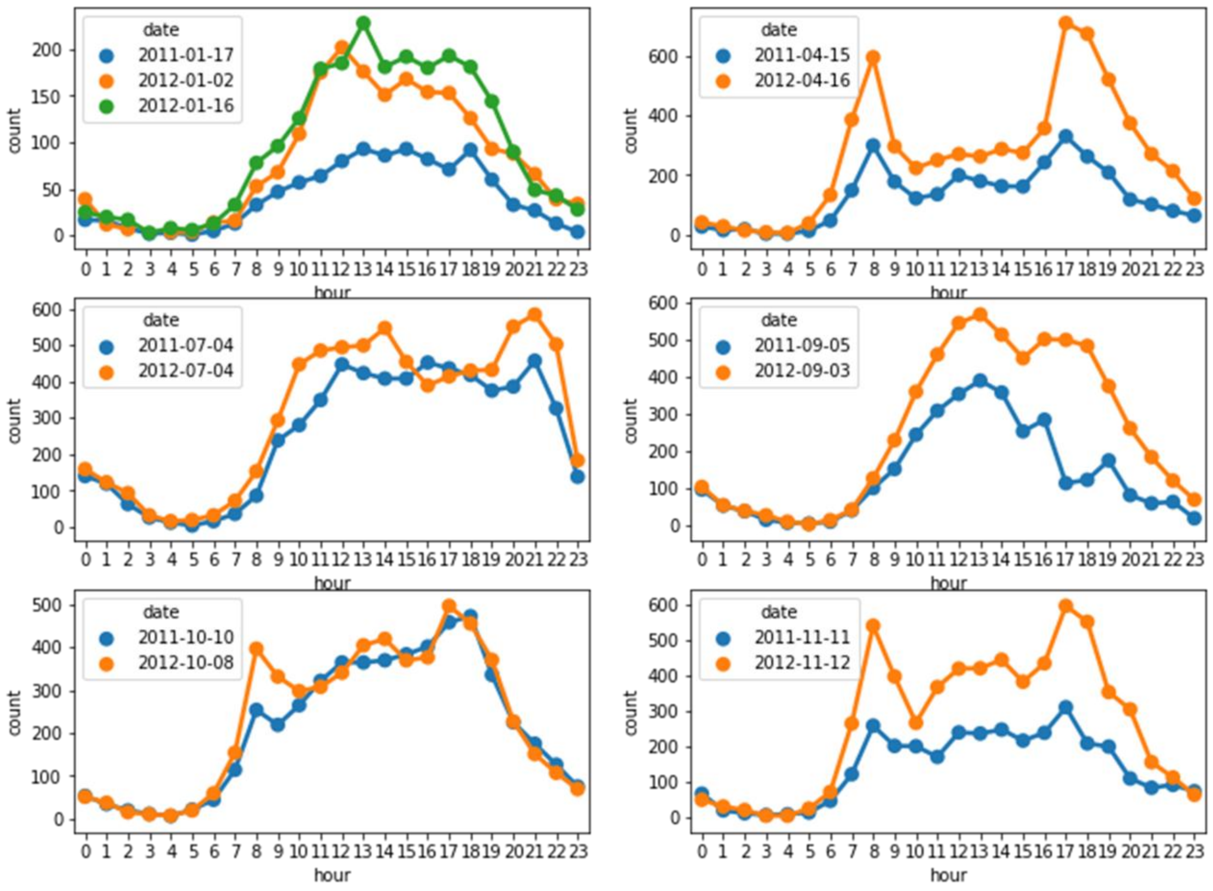

Figure 3: Hourly demand of bike rental on holiday.

To explore the effect of each features on bike demand, we performed correlation analysis on the training data set (Figure 4). It should be noted that the sum of “casual” and “registered” is equal to “count” in the dataset, and both features (“casual” and “registered”) are highly correlated to the bike demand (0.69 and 0.97, respectively). Besides, both features are also not available in the test dataset, and this means that these features should be removed when the dateset is being used to train any machine learning models. The “atemp” and “season” features are highly correlated to “temp” and “month” features, respectively. We choose to remove “atemp” and “season” for the training of machine learning models, as “temp” and “month” can accurately capture the nuances of the factors that can affect the bike demand in Washington D.C.. It should be noted that “workingday” is highly negatively correlated to “weekday” (correlation factor of -0.7). Here, we chose not to remove either of the features as they are important features to finely predict the demand of

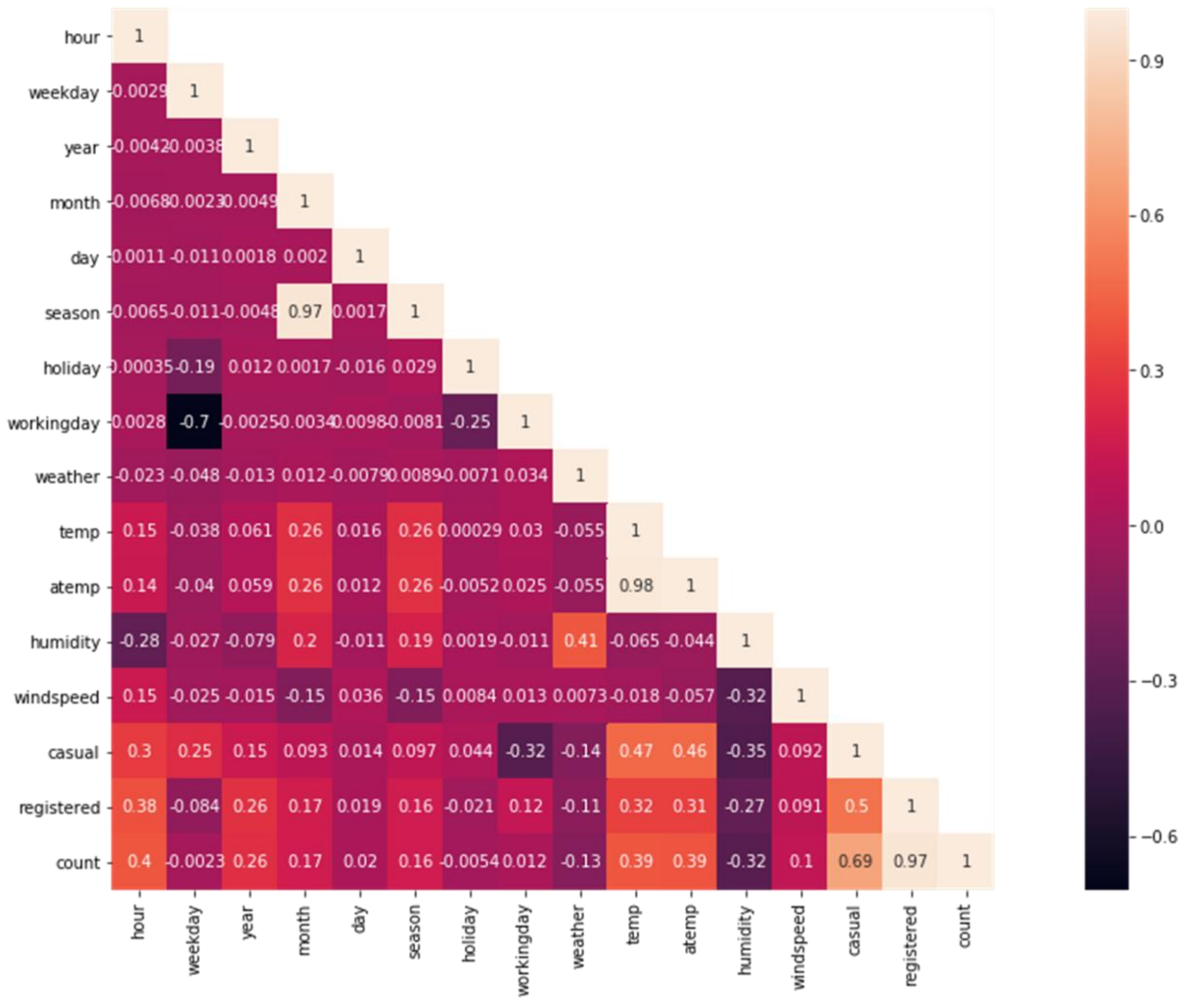

Figure 4: Correlation analysis of features.

the bike rental.

While both “casual” and “registered” will not be considered for training of machine learning models, they both represents interesting features for bike rental company for understanding the rental usage of their customer base. Correlation analysis between “casual”, “registered”, and “count” on working days and non-working days are shown in Figure 5. On working days, the “registered” has much high correlation factor(0.99) to “count” as compared to “casual” (0.7). However, the difference in correlation factors (0.94 and 0.97 for “casual” and “registered”, respectively) is smaller on non-working day. An hourly bike demand by “registered” and “casual”, shown in Figure 6, reveals that the “registered” users dominated the bike demand on both working and weekend, while a

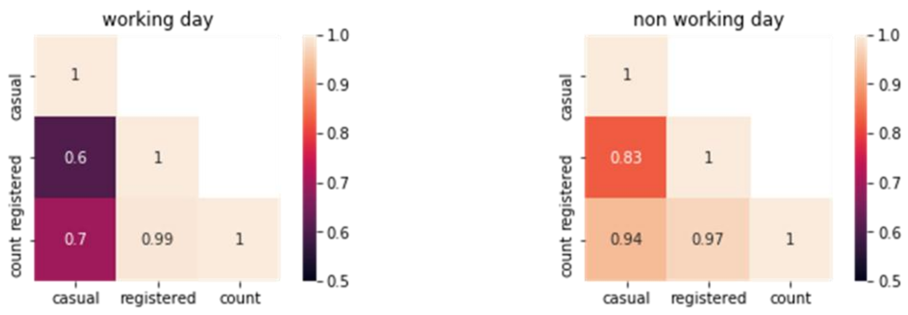

Figure 5: Correlation analysis of casual, registered, and count on working day (left) and non-working day (right).

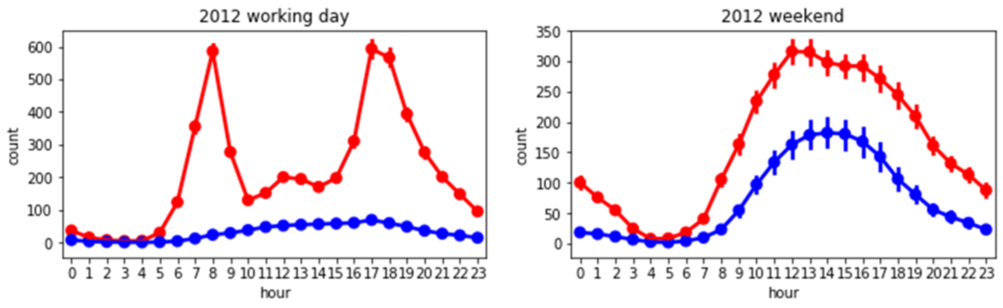

Figure 6: The count of registered (red) and casual (blue) rental on working day and weekend day in year 2012.

higher ratio of "casual" users are observed on the weekend.

Before we proceed to the machine learning analysis, we performed outlier analysis on the output "count" of the dataset(Figure 7). The linear scale of boxplot (top left and right) shows a few distinct characteristics: (1) the median, and the 25 to 75 percentile are lower than of the mean; (2) the outlier points are all on the upper part of the plot. Meanwhile, the logarithmic scale of the boxplot (bottom left and right) shows the opposites features: (1) the median, and the 25 to 75 percentile are upper than of the mean; (2) the outlier points are all on the lower part of the plot. The distribution of the count are shown in Figure 8. While examining the data, we notice that the daily rental has higher distribution of lower counts since there are lesser users from 10pm to 6am. While the higher counts are mostly coming from the peaks hour during the working days (around 7-8 and 17-18 hours). By taking the logarithmic scale of the "count" (Figure 8), the data renormalized to smaller range (about 0-7) from a larger range (about 0-1000). This also renormalized the "count" values, hence the outlier are on the low part of "count" value. A smaller range of the output

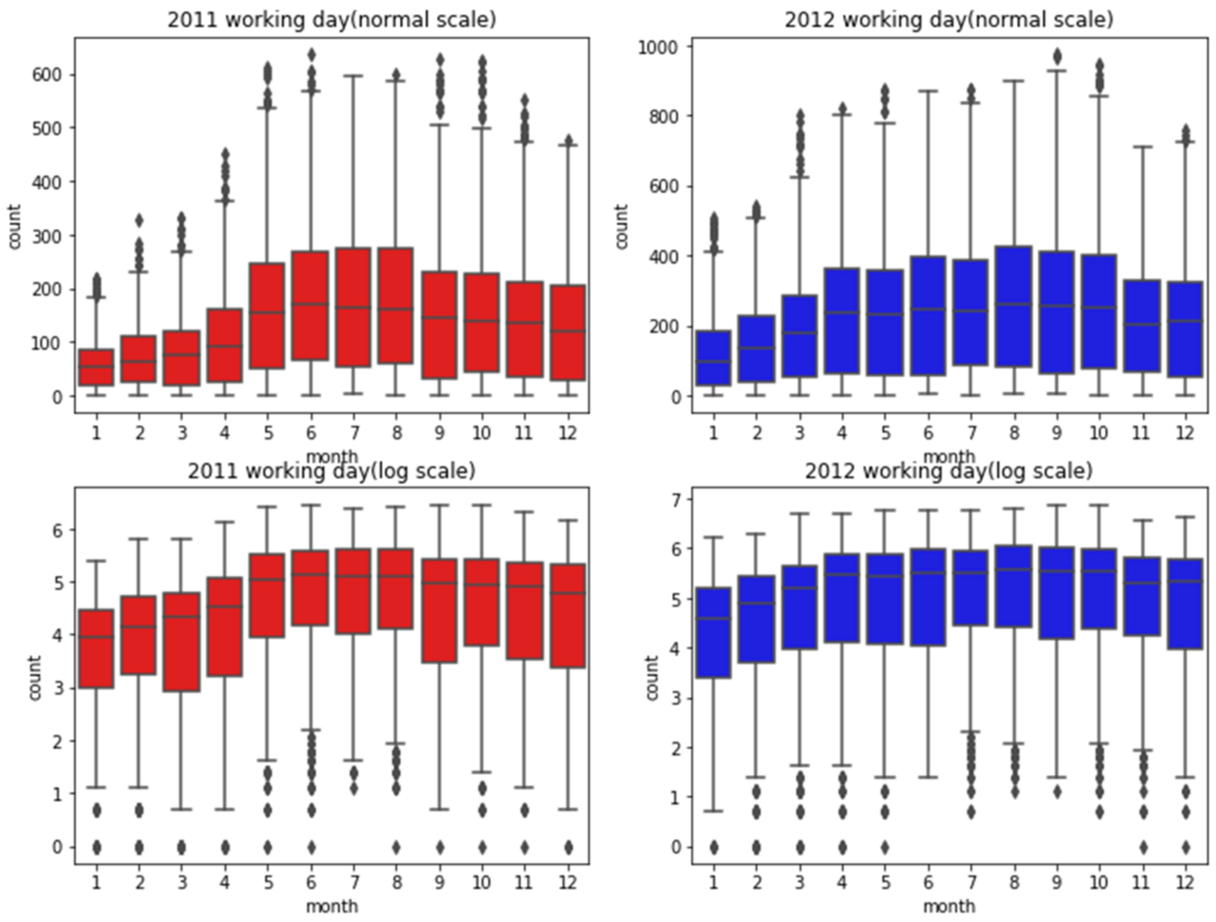

Figure 7: Box plot of count as a function of month in 2011 and 2012. Top left and right show the plot in linear scale on working days in year 2011 and 2012, respectively. Bottom left and right shows the show the plot in logarithmic scale on working days in year 2011 and 2012, respectively.

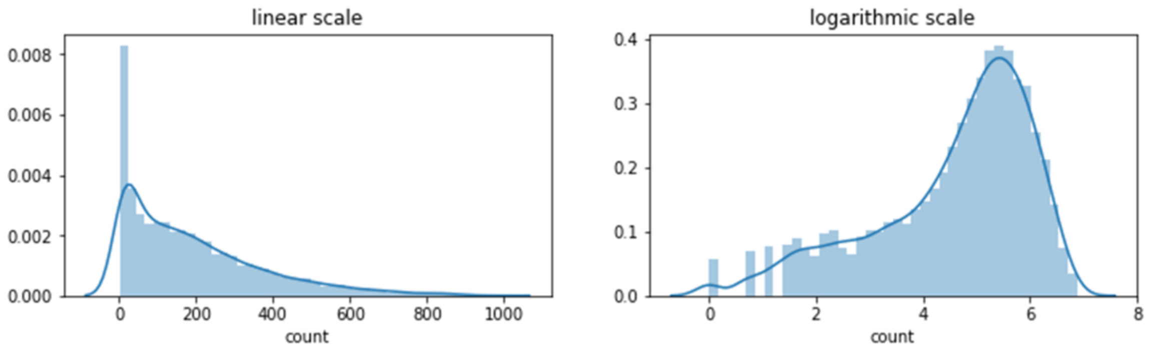

Figure 8: Distribution of count in linear (left) and logarithmic scale (right).

value improve the speed of machine learning model training as the model parameter only has to fit to a smaller range of output.

Algorithms and Techniques

As mentioned in the previous section, the bike demand problem is a regression problem. Given historical usage patterns with weather data, we are tasked to predict the number of bike rental demand. We explored boosting method as a solution to the problems. Boosting method is one of the ensemble models that use boosting to improve the performance of the model. It trains and improves the subsequent learner based on the previous learner. In this case, we can use a decision tree model as the base model for training. The decision tree works by answering a series of questions that enables better purity in the leaves. The model makes predictions by calculating the weighted average of all the predictions from all learners that were trained. This reduces the biases and variances from each individual learner, thus giving better and more generalized predictions.

In this work, we explore AdaBoost and Gradient Boosting Regressor to predict the hourly bike demand. AdaBoost uses short decision trees as the base learner.[3] The algorithm improved the performance of each training instance by weighing the correctly predicted instances with a higher weight while the subsequent base learner focused on learning from wrongly predicted instances that are given less weight. Trees are built on top of each other and the weights are updated subsequently. The model has also been applied to different application such as Face Detection.[3] The model is suitable for large dataset and is returning feature importances. It reduces bias and variance. Compare to Random Forrest, the training time can be very long. The model can overfits as well if data sample is too small.

Gradient Boosting Regressor is one of the ensemble model that uses boosting to improve the performance of the model. Boosting method trains and improves the subsequent learner based on the previous learner.[4-6] For gradient boosting method, the model started by predicting the mean of all variable. The subsequent decision tree model will be trained to minimize the residual, a.k.a the difference between prediction and actual label, hence improving the previous model by a gradual step. The model then update/minimize the errors of the assignment on the subsequent learner by using gradient descent. When the error minimized, a weighted average of the mean of all learner are then used for making the final model and predictions. When training the model, stopping criteria must be tuned to avoid overfitting of the training data. Similar to AdaBoost, the model also return feature importances, and tends to reduce bias and variance. Compare to Random Forrest, the training time can be very long. The model can overfits as well.

Benchmark

According to competition, the result should be benchmarked with Random Forrest Model. Random Forrest Model is an ensemble method that combining several weak learner(i.e. decision tree) to enhance and improve the overall prediction accuracy. Compare to most ensemble model, Random Forrest offers relatively high performance with shorter training time; the model can also handle missing values; it can be modeled with categorical inputs; given enough trees, the model tends not to overfit.

The random forest model is commonly used as standard machine learning model and it has been wide utilized for different application such as Machine Fault Diagnosis.[7] Compare to most ensemble model, Random Forrest offers relatively high performance with shorter training time; the model can also handle missing values. However, the model can become so large due to high number of tree; it tends to underfit when number of features are very low. The model handles the categorical data input and returns feature importances which help in identify the major features affecting the outcomes. Random Forrest uses decision tree as classifier, which allows easy interpretation of the model outcomes. Besides, this ensemble of decision tree method helps reduces the variance of the model, giving a more generalized prediction.

While random forest regressor is suggested, there is tutorial or related benchmarked value to achieve. Hence, we will also implement and optimize the random forest model as a benchmark for our analysis.

III. Methodology

Data Preprocessing

As the dataset is given in CSV format, we used csv_read function of Pandas dataframe to import the data file. The given features are datetime, season, holiday, workingday, weather, temperature, "feel like" temperature, humidity, windspeed, casual, registered, count. Noted that count is to be predicted, while registered and casual is to be removed as they are not available in test dataset and they are also highly correlated to the count (see Figure 4 and 5).

We started by extracting date, hour, weekday, year, month, and day from "datetime". We then set season, holiday, workingday, weather, weekday, month, year, hour as categorical features, and leaves the rest of the features as numerical features (see data_preprocess2 function in the Jupyter notebook).

We drop datetime, atemp, season, date. Datetime is being dropped since the information has been extracted into hour, weekday, year, month, and day. Atemp, which represents the "feel like" temperature are dropped since it has high correlation with temperature, in which it might give super collinearity in the subsequent training. For the same reason, we drop the feature "season", since the feature "month" can better capture the nuances of the problem than "season". Since we have the "weekday"(Monday, Tuesday, etc) as feature, we dropped "day" as it provides little information about the bike demands. The final features used for training are 'holiday', 'workingday', 'weather', 'temp', 'humidity', 'windspeed', 'count', 'hour', 'weekday', 'year', 'month', 'day'.

Implementation & Refinement

We defined our RMSLE as it is not a standard error function in sklearn modules. All the regressor (i.e. Random Forest Regressor, AdaBoost Regressor, and Gradient Boosting Regressor) models can be imported from sklearn.ensemble.

We used two validation methods for the training the dataset. For the first method, we split the data with date before 15th of the month as training data, while the rest is used as validation dataset. This method will later be referred as "train/valid" method. For the second method, we use cross-validation method with number of split k=4. We can use cross-validation method as the "day" has been removed and the data is no longer being seen as in a chronological order; hence there should not be any data leakage from the data of later days.

Other than setting the random seed to 0, we used the default parameters for training the model. Refinement of the model can be achieved by using the GridSearchCV from sklearn.grid_search. In this case, grid parameter and a scorer function to be defined for fine tuning the model parameter.

IV. Results

Model Evaluation and Validation

We started by evaluating the benchmarked model (i.e. Random Forest Regressor) on the problem. Other than the random seed is fixed to zero, the default setting were used. Using the train/valid split method, the default Random Forest Regressor gives RMSLE score of 0.1272, 0.3249, 0.48294 for the train, validation, and test, respectively. When GridSearch_CV is used to fine tune the model parameter, we can get RMSLE score of 0.1489, 0.3065, 0.47029 for the train, valid, and test, respectively. By using all of training dataset for training, a cross validation method is to fine tune the parameter. The results are RMSLE score of 0.1034 and 0.45905 for train and test dataset , respectively. The RMSLE scores are summarized in Table 1. We noted that all the RMSLE score of train dataset is lower than of validation/test dataset. This indicates that the model overfits the training dataset, hence it gives a prediction with larger error on "new dataset". The feature importances of the optimized Random Forest Regressor model obtained with gridSearch_CV method are shown in Figure 9. The "hour" feature is the most important one as hourly prediction is tasked for the problem.

We explore boosting methods (AdaBoost and Gradient Boosting) to improve the prediction accuracy. All RMSLE scores of AdaBoost method(>0.63) are higher than of Random Forest (<0.50). However the train, validation, and test scores are closer to each other, indicating the model does not overfit the data, since it is able to give a generalized prediction on both the old and new dataset. High RMSLE scores of all datasets nevertheless indicates that the model underfits, meaning the complexity of the model fails to capture the nuances from the datasets. This is likely due to the simplicity of AdaBoost model that uses a very shallow

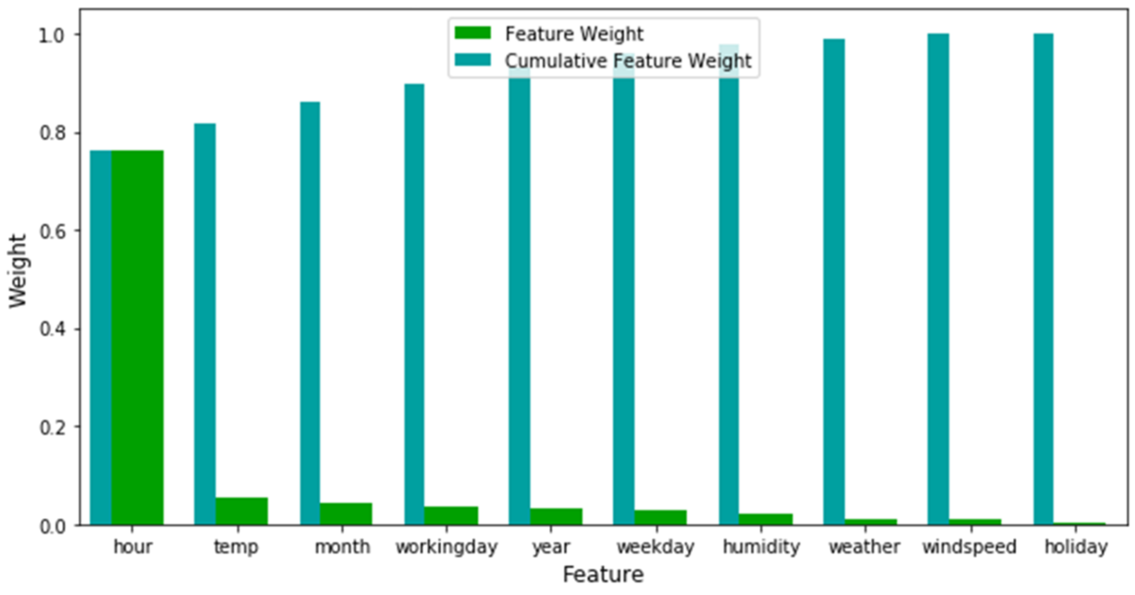

Figure 9: Feature importances of the optimized Random Forest Regressor model obtained with gridSearch_CV method.

decision tree as the base learner.

Next, we explore the Gradient Boosting Regressor to improve the prediction. The default model gives RMSLE scores of 0.3515, 0.3645, and 0.4915 for the train, validation, and test

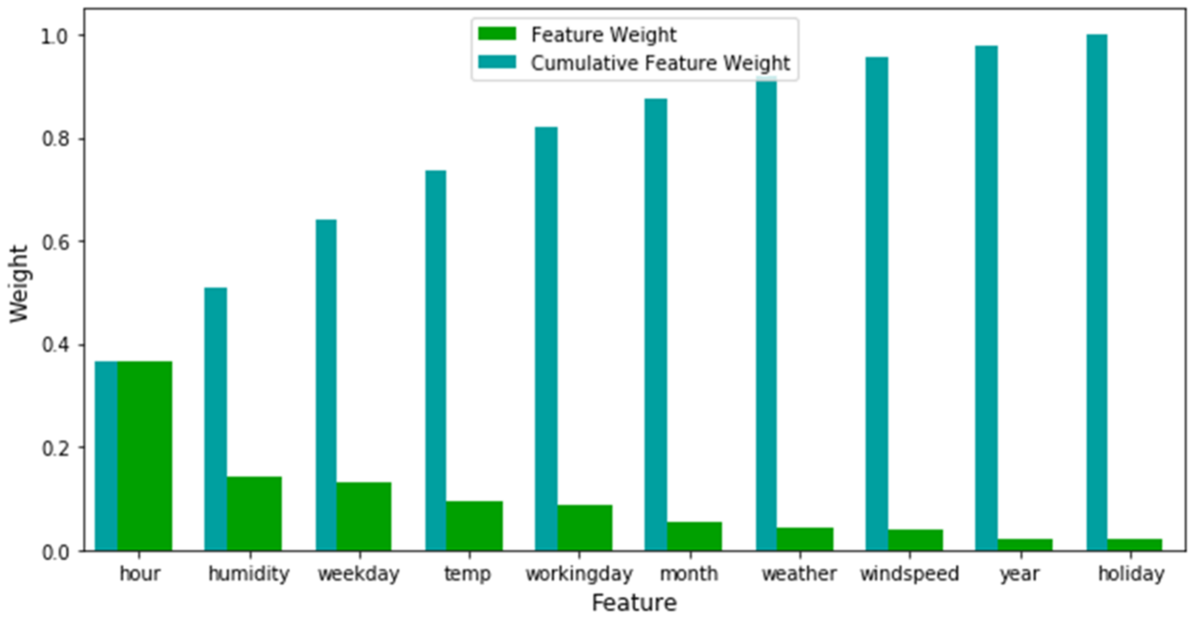

Figure 10: Feature importances of the optimized Gradient Boosting Regressor model obtained with gridSearch_CV method.

Machine Learning Model	valid/train split			Cross-Validation	
	train	valid	test	train	test
Random Forest (default)	0.1272	0.3249	0.48294	-	-
Random Forest (optimized)	0.1489	0.3065	0.47029	0.1034	0.45905
AdaBoost (default)	0.6398	0.6459	0.69390	-	-
AdaBoost (optimized)	0.7080	0.7215	0.76112	0.6986	0.75198
Gradient Boosting (default)	0.3515	0.3645	0.4915	-	-
Gradient Boosting (optimized)	0.2429	0.2913	0.43593	0.2429	0.42583

Table 1: Summary of RMSLE scores obtained from models and data splitting methods. The optimized models uses Grid_Search_CV to fine tune the model parameters. Cross-validation method uses k-fold value of 4 for grid-searching the best model parameters.

dataset, respectively, when using the train/valid splitting method. The grid search further improves the RMSLE scores to 0.2429, 0.2913, and 0.43593 for the train, validation, and test dataset, respectively. When using the cross validation methods, the RMSLE scores of train and test dataset are 0.2429 and 0.42583, respectively. It should be noted that the optimized models have a lower RMSLE scores than the benchmarked Random Forest model, meaning the Gradient Boosting model gives a better accuracy prediction on the new dataset. We also observed that the RMSLE scores of train dataset are higher than of validation and test dataset, suggesting that the model overfits the training dataset. The feature importances of the best Gradient Boosting Regressor model obtained with gridSearch_CV method are shown in Figure 10. When compared to Figure 9, the "hour" feature is still the most important as expected, although the weight is lower. We noted that the weather factors (temperature, humidity, month), working day, and weekday strongly influences the prediction. "Holiday" does not provide a predictive power as the feature is covered as a non-working day in "working day" feature.

Justification

The Gradient Boosting Method is found to be better than the benchmarked Random Forest model. When comparing the optimized models of the two, Gradient Boosting Regressor consistently gives a better (i.e. lower) scores of validation and test dataset. This suggests that a generalized model can be obtained with Gradient Boosting Regressor, and it provides a better accuracy prediction on unseen dataset. While the training and prediction duration are longer for the case of Gradient Boosting Regressor than Random Forest, a better prediction accuracy of the two is desirable as the problems does not require real-time update and prediction. Therefore, the optimized Gradient Boosting model can be deployed for actual bike demand prediction.

V. Conclusion

In summary, we studied the bike rental demands in Washington D.C. with various ensemble machine learning methods. We started by exploring the training data to visualize the average demand across the years, and hourly demands on different seasons and weekdays. The features of the training data can be summarized as below:

- The overall demands are higher across the years, suggesting that there is a growth in demand or in business;
- Characteristic peaks around 7-8 and 17-18 hours can be seen on working days;
- Holidays are celebrated differently. Characteristic peaks around 7-8 and 17-18 hours seen on certain holiday suggests that these holidays might not be heavily celebrated;
- Characteristic peaks around 7-8 and 17-18 hours are likely to be highly related to registered users;
- On linear scale, the dataset has a higher distribution of the low “count”. The distribution is skewed towards to low or high count depending on the linear scale and logarithmic scale.
- The “hour” is taken as the most importance features as the problem required to predict hourly bike demand. The “holiday” feature is however the least important since the feature has been covered as non-working day in “working day” feature.

Reflection

We used boosting algorithm to improve the prediction of bike demand as compared to the benchmarked Random Forest model. The bike demand varies across seasons and weather, and it generally increases across years. There are factors/features that we can derive from the existing dataset, such as day, year, month, hour, and weekday, to improve the model prediction. One of features that are highly correlated to each other (such as season vs month, atemp vs temp) should be removed to reduce the collinearity in the training process; the feature that can better capture the nuances shall be used for model training.

The most challenging part of the problem is that the data is in chorological order. We were provided the first 19 days of the month for training, and it posts the problem of data leakage of the future data to the historical data. To reduce the data leakage, we introduce the weekday (Monday, Tuesday, etc) to replace the "day" feature as it better represents the nuances of the problem. This allows random train-validation data splitting and the use of cross-validation method to train on the entire dataset; the result gives a more accurate prediction on test dataset (unseen dataset).

AdaBoost and Gradient Boosting regressor are being tested against the benchmarked Random Forest. AdaBoost, which uses a shallow decision tree as base learner, is proven to be able to make generalized prediction of across train, validation, and test dataset, but the model underfits since all of the RMSLE are higher. While Gradient Boosting regressor gives better prediction on validation and test dataset, a distinct difference between the RMSLE scores of train and validation/test dataset shows that the model overfits.

As all of the models used decision tree as base learner, the models return feature importances. Features importances inform the key features of the problems, and provide insights on future strategies to approach the problems, as well as building a simplified model. It is noted that "hour", weather info (month, temp, humidity), and working days are among the most important features in predicting bike demand, while the "holiday" feature is the least important since its' nuances has been covered in "working day".

Improvement

To improve the accuracy of the prediction, there are a few strategies that we can employ:

- Feature engineering: We approach and solve the problem with only 10 features. To obtain more features, we can derive the polynomial features, which account of the interaction between features (such as combination of working day + weekday), to allow the model to capture a finer nuances of the problems. Besides, we can expand the holiday features to certain types of holiday (e.g.: Is the holiday a festive celebration(Christmas etc)? Is the holiday a national day?). Different holidays are celebrated differently (e.g. people might take leaves for family reunion/longer vacation.) and it affects the overall demands on the bike rental. Other factors, such as holiday falls on Monday or Friday, may have to be considered as this might encourage local population to have a short vacation in nearby cities.

- Better algorithm: Light GBM[8] and XGBoost[9] are widely used as a replacement for Random Forest model to give better prediction.
- A combination of embedding matrix and deep neural network might also be able to give a better prediction with minimal feature engineering. This model may provide a better prediction in a distance future as it can provide a continuous numerical prediction as compared to the decision tree based ensemble methods.

Reference:

1. Ashqar, Huthaifa I., et al. "Modeling bike availability in a bike-sharing system using machine learning." Models and Technologies for Intelligent Transportation Systems (MT-ITS), 2017 5th IEEE International Conference on. IEEE, 2017.

2. Datta, Arnab Kumar. Predicting bike-share usage patterns with machine learning. MS thesis. 2014.

3. https://www.analyticsvidhya.com/blog/2015/05/boosting-algorithms-simplified/

4. https://medium.com/mlreview/gradient-boosting-from-scratch-1e317ae4587d

5. https://machinelearningmastery.com/gentle-introduction-gradient-boosting-algorithm-machine-learning/

6. https://www.analyticsvidhya.com/blog/2015/05/boosting-algorithms-simplified/

7. https://link.springer.com/chapter/10.1007%2F978-1-84628-814-2_82

8. http://lightgbm.readthedocs.io/en/latest/Python-Intro.html

9. https://xgboost.readthedocs.io/en/latest/index.html