

**Background :** People are able to rent a bike from one location and return it to a different location on an as-needed basis. Data like duration of travel, departure and arrival locations, time elapsed etc was recorded -> hence acts as sensor data to study the mobility in a city. Ask of the Competition - Combine Historical usage patterns with the weather data to forecast bike rental demand in Washington D.C.




**Kaggle Link:** <https://www.kaggle.com/c/bike-sharing-demand/data>

- 3,242 Teams
- 3,559 Competitors
- 32,809 Entries

This competition did not award **ranking points**. This competition did not count towards **tiers**

Note that InClass, Playground, and Getting Started competitions typically do not award medals. Many users watch the public leaderboard closely, as breakthroughs in the competition are announced by score gains in the leaderboard.

The private leaderboard, by contrast, tracks model performance on data unseen by participants. The private leaderboard thus has final say on whose models are best, and hence, who the winners and losers of the Competition will be.

#	Δpub	Team Name	Notebook	Team Members	Score ?	B
1	—	Bolaka Mukherjee			0.33756	
2	—	Logical Guess			0.34821	
3	—	Louis Martin			0.34834	

**Problem Summary:** Hourly rental data spanning two years have been provided.

**Training Set:** comprised of the first 19 days of each month

**Testing Set:** Comprised the data from 20th to the end of the month

**Goal:** Predict the total count of bikes rented during each hour covered by the test set, using only information available prior to the rental period.

**Sample Output:**

	datetime	count
0	2011-01-20 00:00:00	12.237781
1	2011-01-20 01:00:00	6.016530
2	2011-01-20 02:00:00	4.656349
3	2011-01-20 03:00:00	2.871552
4	2011-01-20 04:00:00	2.592699

## Data Understanding and Exploration:

- **Columns description:**

1. datetime — hourly date + timestamp
2. season — 1 = spring, 2 = summer, 3 = fall, 4 = winter
3. holiday — whether the day is considered a holiday
4. workingday — whether the day is neither a weekend nor holiday
5. weather —
  - 5.1: Clear, Few clouds, Partly cloudy, Partly cloudy
  - 5.2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
  - 5.3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
  - 5.4: Heavy Rain + Ice Pellets + Thunderstorm + Mist, Snow + Fog
6. temp — temperature in Celsius
7. atemp — “feels like” temperature in Celsius
8. humidity — relative humidity
9. windspeed — wind speed
10. casual — number of non-registered user rentals initiated
11. registered — number of registered user rentals initiated
12. count — number of total rentals

- **Independent Variables**

1. datetime: date and hour in "mm/dd/yyyy hh:mm" format
2. season: Four categories-> 1 = spring, 2 = summer, 3 = fall, 4 = winter
3. holiday: whether the day is a holiday or not (1/0)
4. workingday: whether the day is neither a weekend nor holiday (1/0)
5. weather: Four Categories of weather
  - 1-> Clear, Few clouds, Partly cloudy, Partly cloudy
  - 2-> Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
  - 3-> Light Snow and Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
  - 4-> Heavy Rain + Ice Pellets + Thunderstorm + Mist, Snow + Fog
6. temp: hourly temperature in Celsius
7. atemp: "feels like" temperature in Celsius
8. humidity: relative humidity
9. windspeed: wind speed

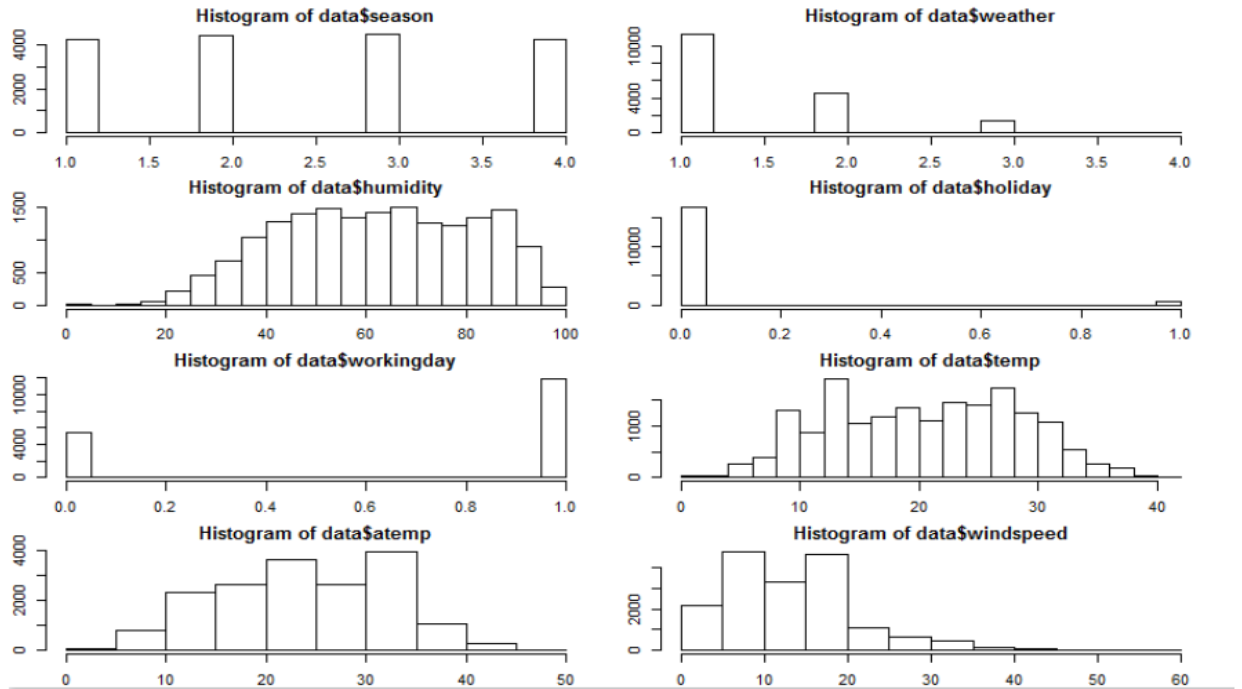
- **Dependent Variables**

1. registered: number of registered user
2. casual: number of non-registered user
3. count: number of total rentals (registered + casual)

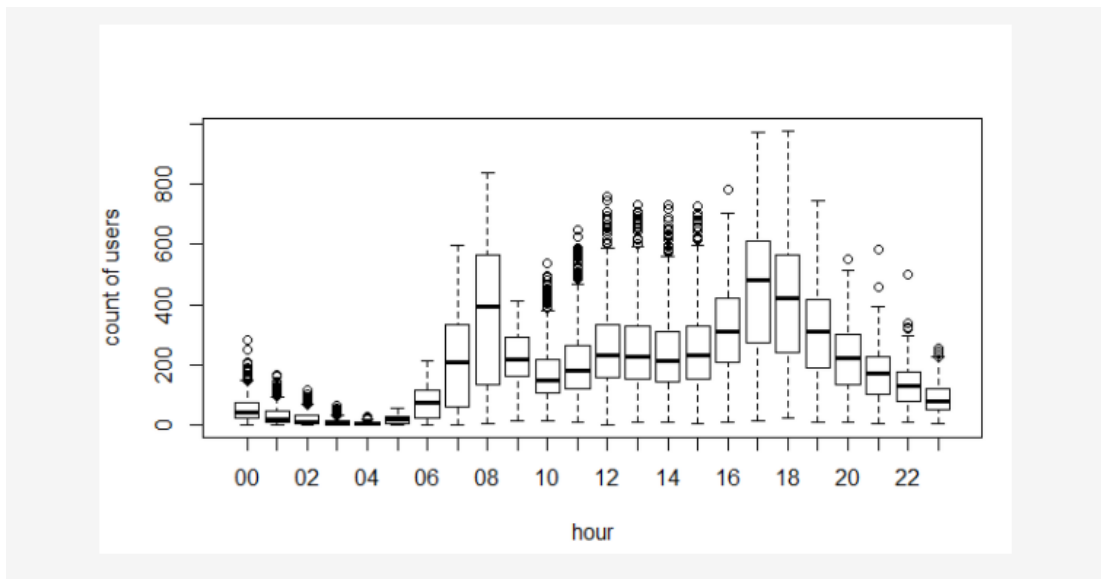
- **Variable Type Identification:**

```
str(data)
'data.frame': 17379 obs. of 12 variables:
 $ datetime : Factor w/ 17379 levels "2011-01-01 00:00:00",...: 1 2 3 4 5 6 7 8 9 10 ...
 $ season   : int  1 1 1 1 1 1 1 1 1 1 ...
 $ holiday  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ workingday: int  0 0 0 0 0 0 0 0 0 0 ...
 $ weather  : int  1 1 1 1 1 2 1 1 1 1 ...
 $ temp     : num  9.84 9.02 9.02 9.84 9.84 ...
 $ atemp    : num  14.4 13.6 13.6 14.4 14.4 ...
 $ humidity : int  81 80 80 75 75 75 80 86 75 76 ...
 $ windspeed: num  0 0 0 0 0 ...
 $ casual   : num  3 8 5 3 0 0 2 1 1 8 ...
 $ registered: num  13 32 27 10 1 1 0 2 7 6 ...
 $ count    : num  16 40 32 13 1 1 2 3 8 14 ...
```

- **Understanding the distribution of numerical variables and generating a frequency table for numeric variables**



- **Hypothesis Analysis: Season Based Bike Demand**



- Use a metric Analyzer for training Accuracy using Regression model - Linear Regression, Gradient Boosting Regressor, Random Forest Regressor and Decision Tree to find the best model. The best model is Gradient Boosting Regressor.

- Use sklearn cross\_validation(K=10) during training the data, tree, svm, linear\_model, preprocessing, neighbors, ensemble for data analysis

## Comparison[Top Tip and Tricks]:

Winning Leaderboard	Second Leaderboard	Third Leaderboard
<b>Bolaka Mukherjee : 0.33756</b>	<b>Logical Guess : 0.34821</b>	<b>YoungMin Kim : 0.05942</b> <b>Joined last year</b>
<a href="https://github.com/Bolaka/bike-sharing">https://github.com/Bolaka/bike-sharing</a>	<a href="https://github.com/logicalguess/kaggle-bike-sharing-demand">https://github.com/logicalguess/kaggle-bike-sharing-demand</a>	<a href="https://www.kaggle.com/winston1214/bike-sharing-third-kaggle">https://www.kaggle.com/winston1214/bike-sharing-third-kaggle</a>
<ul style="list-style-type: none"> <li>• With feature importance</li> <li>• With cross validation using classification model.</li> <li>• Ensemble Method Classification and Regression</li> <li>• Compare all Regression Metric Scores</li> <li>• Explored and captured the effect of seasonality and weather and by demand.</li> </ul>	<ul style="list-style-type: none"> <li>• Only two regressor model</li> <li>• No Feature Importance/feature engineering</li> <li>• Model used are RandomForest Regressor and GradientBoosting Regressor</li> </ul>	<ul style="list-style-type: none"> <li>• Did detailed data exploration</li> <li>• Cross validation using different CV metric regularization to optimize hyperparameter</li> <li>• Ensemble method for regression and used LGBRegression which gives better result</li> <li>• Compared the RMLSE score for all the models</li> </ul>

## What is the Strategy?

**Leaderboard leader :** Bolaka Mukherjee got a lowest RMSLE (Root Mean Squared Logarithmic Error) score of 0.33756 on this competition.

## What did they do to win the competition :

1. Prepare the data from the file and convert it to dataframe.
2. Extract and get the best features to be used in the dataset.
3. Prepare the target values and inspect the feature importance.
4. Did a **cross validation** using k = 10 fold leaving some specific weeks data ( use 10 weeks to test ).Used classification to find the accuracy of target prediction for Xtest sample and compare it with different classification models.
  - ADABOOST Classifier

- Bagging Classifier
  - ExtraTrees Classifier
  - Random Forest Classifier
  - Gradient Boosting Classifier
  - KNeighborsClassifier
  - Train the classifier on all data, predict all test data including the casual and registered user prediction.
5. Similarly used regression modeling to compute the error score of the training model with hyperparameter tuning.
    - Linear Regression
    - Decision Tree Regressor
    - Gradient Boosting Regressor
    - Random Forest Regressor
  6. Train the labels casual and registered users and check if the model can predict the train dataset including the negative values. Plotted the predicted vs true values for random week.
  7. Add prediction on casual and registered user with error scores RMSLE
  8. Compare all the average cross-validation score AVERAGE RMSLE.
  9. Computed all the metric score errors for comparison RMSLE, RMSE, MSE, MSLE, SE, SLE, MAPE

## **Data Preparation/Cleanup**

1. Using 'datetime' column to get two more features 'time' and 'month'
2. Dropping unnecessary columns.
3. One hot encoding on categorical columns
4. Imputing null values with the median of each respective column.

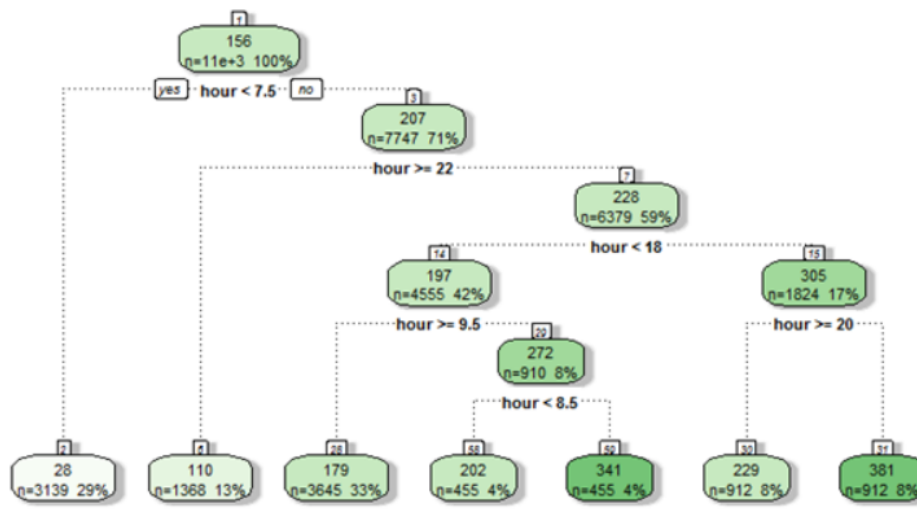
## **Feature Engineering:**

1. Feature Importance selection using RandomForest Regressor.
2. This list decides which features are going to be used, the rest is filtered out.

```
features = ['day_since_begin', 'hour_of_day_cos', 'hour_of_day_sin', 'workingday',
            'temp', 'weather1', 'weather3', 'holiday_external', 'almost_holiday',
            'weekday0', 'weekday1', 'weekday2', 'weekday3', 'weekday4', 'weekday5',
            'weekday6', 'humidity', 'windspeed', 'rush_hour', 'holiday']
```



**Hour Bins:** To categorize the hour into three categories using Decision Tree for casual and registered users



## Model Building:

1. Linear Regression
2. Decision Tree Regressor
3. Gradient Boosting Regressor
4. Random Forest Regression

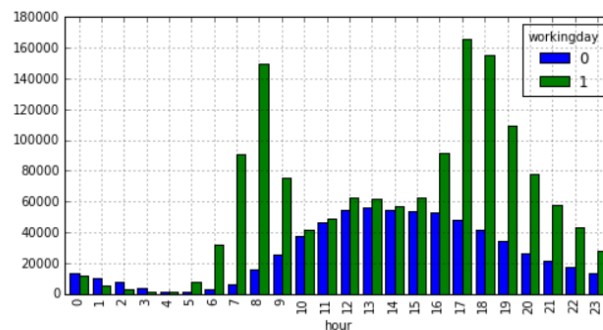
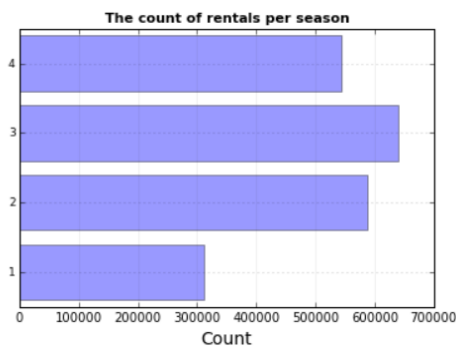
**Best Performing Model:** GBR( $n_{\text{estimators}}=200$ )

Gradient Boosting for regression : GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage a regression tree is fit on the negative gradient of the given loss function.

Takeaways:

1. Do Feature engineering and Extract Important Features
2. Use ensemble methods for classification/regression models and perform hyperparameter optimization.
3. Get the best metric score for each Regression model.
4. Found the best model GBR with lowest RMSLE score **0.3375**.

**Leaderboard 2nd leader** : logicalgues got the second lowest RMSLE (Root Mean Squared Logarithmic Error) score of 0.34821 on this competition.



On workingdays, any deducible effect of humidity, by any chance...?  
(taking the bike to work no matter what, but dry if back home...?)



**What did they do to win the competition :**

1. Predicted the counts of rentals in a city per season, count by hour on a working day and non working day and finally the counts by hour on working day depending on the weather and humidity.
2. Used GBR ( Gradient Boosting Regressor) to find the accuracy of target prediction for



- Xtest sample and compared it with RFR (Random Forest Regressor).
3. Calculated the RMSLE (Root Mean Squared Logarithmic Error ) score on the validation set which is the validation score of the competition to decide the winner.

### Data Prep:

1. Loaded the test and train datasets into dataframes.
2. Interpolate weather, atemp, humidity. Interpolation is a convenient method to create a function based on fixed data points, which can be evaluated anywhere within the domain defined by the given data using linear interpolation.
3. Used logarithmic transformation of 'dependent columns' and added 1 in the beginning to all the values so the null values do not become -inf after the transformation step.
4. He then extracted the columns of features and grouped them by 'date', 'day', 'month', 'year', 'hour', 'dayofweek', 'weekofyear'.

```
cols: base_cols + []  
rmse: 1.351102939
```

```
cols: base_cols + ['hour']  
rmse: 0.471594968034
```

```
cols: base_cols + ['hour', 'dow']  
rmse: 0.455313354995
```

```
cols: base_cols + ['hour', 'dow', 'year']  
rmse: 0.362092301557
```

```
cols: base_cols + ['hour', 'dow', 'year', 'month']  
rmse: 0.334982551858
```

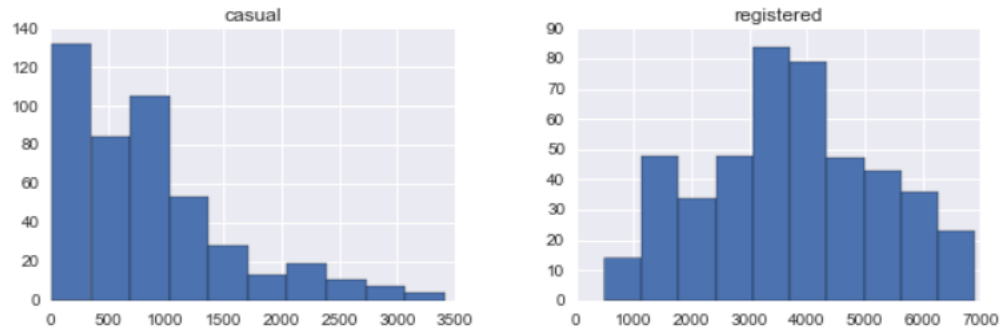
```
cols: base_cols + ['hour', 'dow', 'year', 'month', 'weeks_since_start']  
rmse: 0.337482905295
```

```
cols: base_cols + ['hour', 'dow', 'year', 'month', 'weeks_since_start', 'day']  
rmse: 0.342750167966
```

5. Selected the one with lowest rmse which is the hour, date of week, year and month.
6. Then this feature set was held constant and tuned the model parameters to further improve accuracy using Random Forest.
7. Define a custom train and test split where if the date is less than or equal to the cutoff date which is the 19th of every month, the rows fall into the train and otherwise test.

### Model Building:

1. Used Regression modeling to compute the error score of the training model.
  - a. Gradient Boosting Regressor (GBR)
  - b. Random Forest Regressor (RFR)
2. Combined both casual and registered users as input and then fed into the model for training.



3. Instead of randomly splitting our training data for cross validation, he constructed a framework that's more in line with how the data is divided up for this competition.
4. (Given first 19 days of each month, what is demand for remaining days)
5. Splitted the training data into 2 time contiguous datasets for fitting and validating the model (days 1-14 vs. days 15-19).
6. Also, since submissions are evaluated based on the root mean squared logarithmic error (RMSLE), he replicated that computation as he tested and tuned the model.
7. Predicted the count of rentals per season in a city and calculated the RMSLE score on the validation set.

### Takeaways:

1. Prepared data by grouping and extracted the counts of each group.
2. Used RMSE to check which set of features/parameters are giving out the best results on the model.
3. Used regression models and compared the results.
4. Found the best model GBR with lowest RMSLE score 0.34821.

---

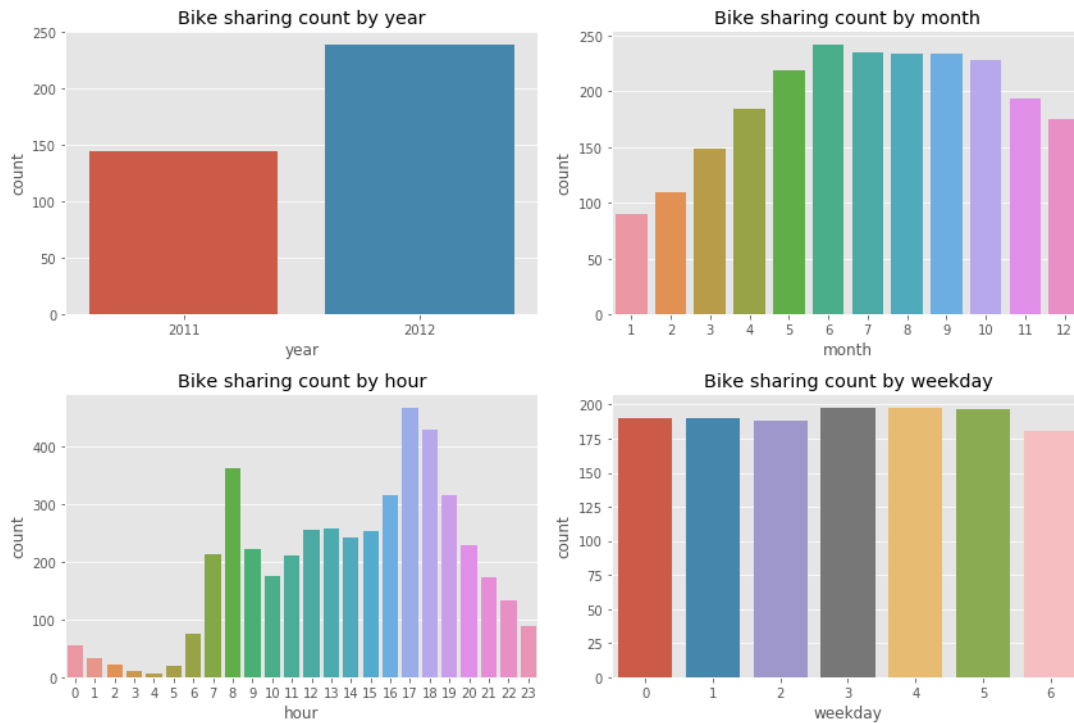
**Leaderboard 3rd leader** : YoungMin Kim got the highest lowest RMSLE (Root Mean Squared Logarithmic Error) score of **0.05942** on this competition but joined after the competition was finished. Our team considered his work in this competition.

### What did they do to win the competition :

1. Loaded the test and train datasets into dataframes.
2. Did extensive exploratory data analysis to come up with various plots like histograms, boxplot of bike sharing count based on different time metrics, seasons; found the distribution of weather attributes like temp, humidity, wind speed etc.
3. Performed Correlation analysis for this features

`"temp", "atemp", "casual", "registered", "humidity", "windspeed", "count"`

4. Compute the count distribution and probability plot
5. Normalize the dependent variable for regression, so  $\log(1+p)$  is used to normalize it. RMSLE



### Data Prep:

Did extensive exploratory data analysis to come up with various plots like histograms, boxplot of bike sharing count based on different time metrics, seasons

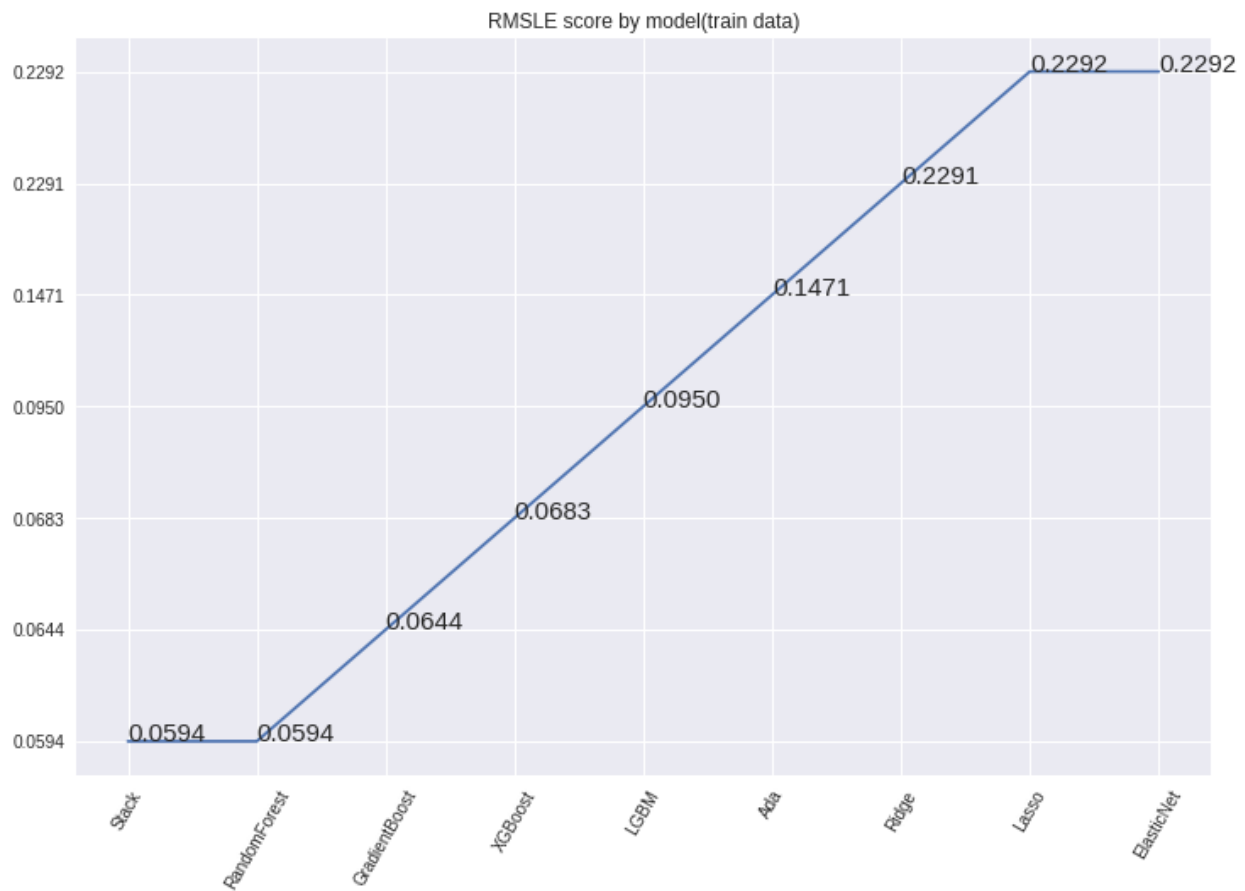
### Feature Engineering:

Considered [season, holiday, workingday, weather, weekday, year] as categorical features.

### Model Building:

1. Created a pipeline by combining RobustScaler
2. Used different type of cross validation method RidgeCV, LassoCV, ElasticNetCV, GridSearchCV to automatically select the best hyperparameter and minimize overfitting
3. Implemented RandomForestRegressor, GradientBoostingRegressor, AdaBoostRegressor, LGBMRegressor. RandomForest was the best with an RMLSE score of 0.0594.
4. Stacked the results of all the models.

	model	rmsle
3	RandomForest	0.0594215
4	GradientBoost	0.0643663
5	XGBoost	0.0682606
6	LGBM	0.0950058
7	Ada	0.147061
1	Ridge	0.229134
0	Lasso	0.229198
2	ElasticNet	0.229198



### Takeaways:

1. Created a pipeline adding RobustScaler to RidgeCV, LassoCV, ElasticNetCV, GridSearchCV to solve overfitting problem
2. Different method of Cross validation using to optimize hyperparameter
3. Feature Importance Implementation
4. Ensemble method for regression analysis
5. Compared the RMLSE score for all the models and found that RandomForest was the best with

RMLSE score of 0.0594

## References:

1. <https://medium.com/analytics-vidhya/root-mean-square-log-error-rmse-vs-rmlse-935c6cc1802a>
2. <https://www.mastersindatascience.org/resources/getting-started-with-kaggle-competitions/>
3. <https://www.kaggle.com/getting-started/44088>
4. <https://www.kaggle.com/getting-started/78482>
5. <https://www.analyticsvidhya.com/blog/2015/06/solution-kaggle-competition-bike-sharing-demand/>
6. <https://towardsdatascience.com/end-to-end-case-study-bike-sharing-demand-dataset-53201926c8db>
7. <https://github.com/brendansudol/kaggle-bike-share/blob/master/ipy-notebooks/main.ipynb>