PROJECT REPORT

BIKE SHARING SYSTEMS

Submitted towards partial fulfillment of the criteria

for award of PGPDSE by Great Lakes Institute of Management

**Submitted By**

**Group No. 3 [Batch: September 2018]**

**Group Members**

1. **Abhilash**
2. **Arshdeep Kaur Ahuja**
3. **Ashish Subudhi Tumula**
4. **Sai Ramya Machavarapu**

**Supervisor**

**Mr. Ankush Bansal**



**Great Lakes Institute of Management**

# ABSTRACT

Bike sharing systems are new generation of traditional bike rentals where whole process from membership, rental and return has become automatic. Through these systems, users can easily rent a bike from one location and return at another location.

Bike-sharing rental process is highly correlated to the environmental and seasonal settings. For instance, weather conditions, precipitation, days of week, seasons, hours of the day, etc. can affect the rental behavior Today, there exists great interest in these systems due to their important role in traffic, environmental and health issues. most of important events in the city could be detected via monitoring bike sharing systems. Hence analyzing such data and building Machine learning algorithms on this data can benefit the bike rental companies in improving their business by allowing them to keep track of demand for the number of bikes which can be rented based on the environmental, seasonal settings and specific hours of the day.

# ACKNOWLEDGEMENT

We certify that the work done by us for conceptualizing and completing this project is original and authentic. We thank our mentor Mr. Ankush Bansal for his constant support and guidance throughout the project.

**Date: 17/02/2019**

**Place: Hyderabad**

# CERTIFICATE OF COMPLETION

I hereby certify that the project titled Bike Sharing systems for case resolution was undertaken and completed under my supervision by Mr. Ankush Bansal of Post Graduate Program in Data Science and Engineering (PGP – DSE).

Ankush Bansal

**Date: 17/2/2019**

**Place: Hyderabad**

Table of Contents

[ABSTRACT 2](#_Toc1318444)

[ACKNOWLEDGEMENT 3](#_Toc1318445)

[CERTIFICATE OF COMPLETION 4](#_Toc1318446)

[ABBREVIATIONS 5](#_Toc1318447)

[CHAPTER 1 5](#_Toc1318448)

[INTRODUCTION 5](#_Toc1318449)

[PURPOSE OF STUDY 5](#_Toc1318450)

[PROBLEM STATEMENT 7](#_Toc1318451)

[DATA SOURCES 7](#_Toc1318452)

[TECHNIQUES AND TOOLS 7](#_Toc1318453)

[LIMITATIONS 7](#_Toc1318454)

[CHAPTER 2 8](#_Toc1318455)

[PROJECT FLOW 8](#_Toc1318456)

[DATA DESCRIPTION 9](#_Toc1318457)

[ATTRIBUTE INFORMATION 9](#_Toc1318458)

[DATA PREPROCESSING 10](#_Toc1318459)

[FEATURE ENGINEERING 10](#_Toc1318460)

[EXPLORATORY DATA ANALYSIS 11](#_Toc1318461)

[TABLEAU VISUALIZATIONS 15](#_Toc1318462)

[INFERENCES FROM EDA 19](#_Toc1318463)

[MODEL BUILDING 20](#_Toc1318464)

[NORMALIZATION 20](#_Toc1318465)

[TEST TRAIN SPLIT 20](#_Toc1318466)

[LINEAR REGRESSION 20](#_Toc1318467)

[DECISION TREE REGRESSION 22](#_Toc1318468)

[RANDOM FOREST REGRESSION 22](#_Toc1318469)

[GRADIENT BOOSTING REGRESSOR 23](#_Toc1318470)

[BAGGING REGRESSOR 23](#_Toc1318471)

[LASSO AND RIDGE 24](#_Toc1318472)

[CHAPTER4 26](#_Toc1318473)

[RECOMMENDATIONS 26](#_Toc1318474)

[MODEL COMPARISION 26](#_Toc1318475)

[CONCLUSION 27](#_Toc1318476)

[BIBILOGRAPHY 27](#_Toc1318477)

## ABBREVIATIONS

* EDA: Exploratory Data Analysis
* ML: Machine Learning
* OLS: Ordinary Least Square
* RMSE: Root Mean Square Error
* MAPE: Mean Absolute Percentage Error.

## CHAPTER 1

### INTRODUCTION

Bike sharing systems are new generation of traditional bike rentals where whole process from membership, rental and return has become automatic. Through these systems, users can easily rent a bike from one location and return at another location. Currently, there are about over 500 bike-sharing programs around the world which is composed of over 500 thousands bicycles. Today, there exists great interest in these systems due to their important role in traffic, environmental and health issues.

### PURPOSE OF STUDY

Apart from interesting real-world applications of bike sharing systems, the characteristics of data being generated by these systems make them attractive for the research. Opposed to other transport services such as bus or subway, the duration of travel, departure and arrival position is explicitly recorded in these systems. This feature turns bike sharing system into a virtual sensor network that can be used for sensing mobility in the city. Hence, it is expected that most of important events in the city could be detected via monitoring these data.

Hence analyzing such data and building Machine learning algorithms on this data can benefit the bike rental companies in improving their business by allowing them to keep track of demand for the number of bikes which can be rented based on the environmental, seasonal settings and specific hours of the day.

### PROBLEM STATEMENT

The objective of the project is to predict the hourly bike rental count in a bike sharing system based on the environmental and seasonal settings.

### 

### DATA SOURCES

Original Data Source: <http://capitalbikeshare.com/system-data>

Weather Information: <http://www.freemeteo.com>

Holiday Information: <http://dchr.dc.gov/page/holiday-schedule>

### TECHNIQUES AND TOOLS

TOOLS: Python, Tableau

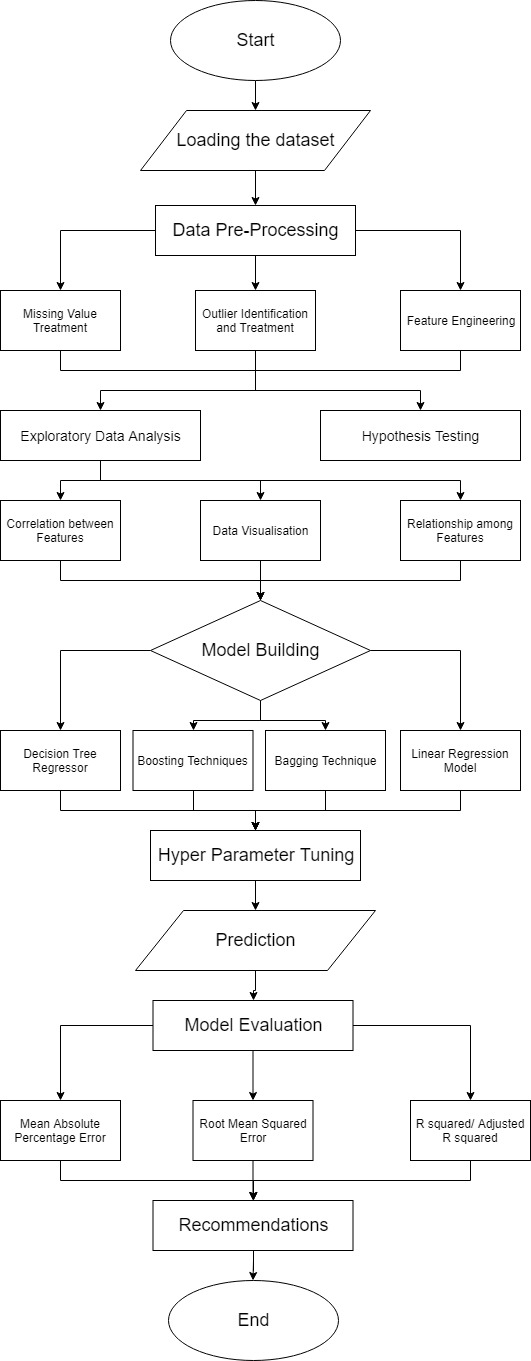
TECHNIQUES: Machine Learning Algorithms, Statistics

DOMAIN: Bike Sharing Systems

### LIMITATIONS

## CHAPTER 2

### PROJECT FLOW



### DATA DESCRIPTION

Bike sharing dataset contains the hourly count of rental bikes between years 2011 and 2012 in Capital bikeshare system with the corresponding weather and seasonal information.

The data set consists of 17379 rows and 17 Attributes. Each row in the dataset gives information about the number of bikes rented hourly with the corresponding weather and seasonal information during that hour

### ATTRIBUTE INFORMATION

* **instant**: record index(int64)
* **dteday**: date(object)
* **season**: season (1: springer, 2: summer, 3: fall, 4: winter) (int64)
* **yr**: year (0: 2011, 1:2012) (int64)
* **mnth**: month (1 to 12) (int64)
* **hr**: hour (0 to 23) (int64)
* **holiday**: weather day is holiday or not (int64)
* **weekday**: day of the week (int64)
* **working day**: if day is neither weekend nor holiday is 1, otherwise is 0. (int64)
* **weathersit**:

1. 1: Clear, Few clouds, Partly cloudy, Partly cloudy (int64)
2. 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist (int64)
3. 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds (int64)
4. 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog (int64)

* **temp**: Normalized temperature in Celsius. The values are derived via (t-t\_min)/(t\_max-t\_min), t\_min=-8, t\_max=+39 (only in hourly scale) (float64)
* **atemp**: Normalized feeling temperature in Celsius. The values are derived via (t-t\_min)/(t\_max-t\_min), t\_min=-16, t\_max=+50 (only in hourly scale) (float64)
* **hum**: Normalized humidity. The values are divided to 100 (max) (float64)
* **windspeed**: Normalized wind speed. The values are divided to 67 (max) (float64)
* **casual**: count of casual users (int64)
* **registered**: count of registered users (int64)
* **cnt:** (Target) count of total rental bikes including both casual and registered (int64)

### DATA PREPROCESSING

* **Data Type Conversion**

Variables like season, yr, mnth, hr, holiday, weekday, working day, weathersit are interpreted as numerical. We changed them to categorical.

We have a datetime object, so it's better to break it into day, month, year and hr.

* **Dropping Unwanted Columns**

Instant, casual count and registered count have been dropped. Instant is the

* **Missing values**

There are no missing values in the data.

* **Outliers**

Count, windspeed and humidity have outliers.

* **Outlier Treatment**

Log transformation is used to treat outliers in count and windspeed.

Humidity has only one outlier in it so that is dropped.

### FEATURE ENGINEERING

* **Mean Temperature:**

We have temp attribute which indicates actual temperature of the air and atemp attribute which indicates feel temperature that is temperature felt on the skin, which can be affected by the wind and humidity in the air. We can created a new column which is the mean temperature of temp and atemp variables.

* **Hour Buckets:**

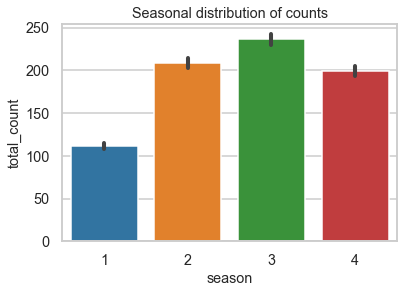
We created various buckets based on demand for bikes at various hours of the day. These buckets are:

low = [0, 1, 2, 3, 4, 5, 6, 23]

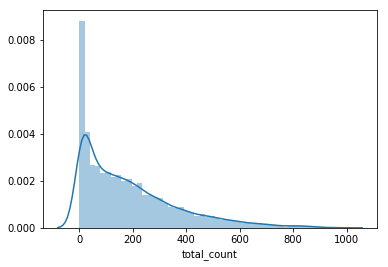
med = [10, 11, 12, 13, 14, 15, 21, 22]

hig = [7, 8, 9, 16, 17, 18, 19, 20]

### EXPLORATORY DATA ANALYSIS

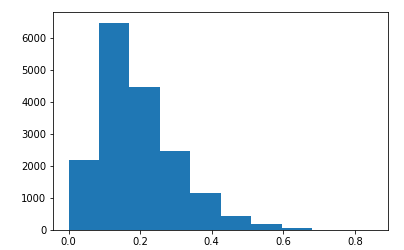
** Distribution of continuous variables**

**Total Count Distribution**



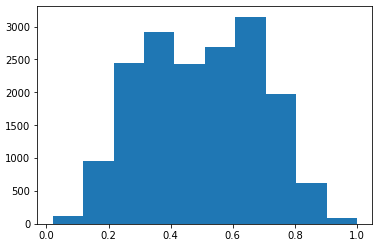
Total count is right skewed, which indicates outliers in the count. Applying a log transformation technique can make it close to normal distribution.

**Windspeed Distribution**



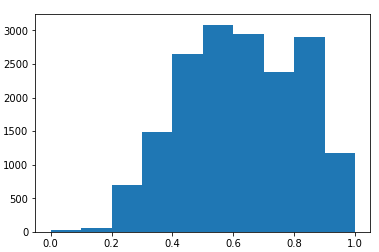
Windspeed distribution is also right skewed

**Temperature distribution**

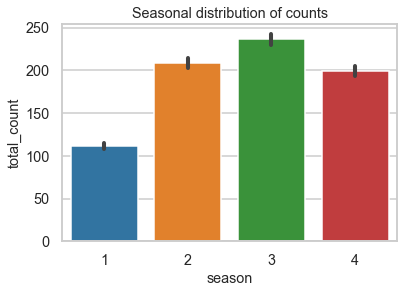


Temperature follow an approximate normal distribution.

**Humidity Distribution**



Humidity is slightly left skewed.



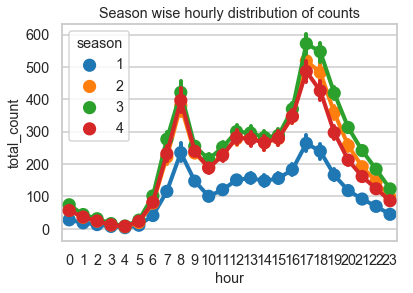
Season1: Spring

Season 2: Summer

Season 3: Fall

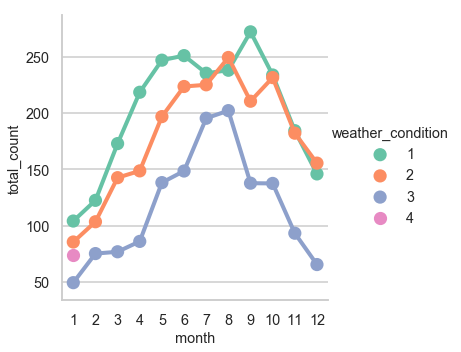
Season 4: Winter

Bike rental count is lowest in spring and highest in fall, hence, we can conclude that the number of customers renting bikes in cold months is very less.



During any season the hourly customer count is almost similar during the peak hours i.e. 8 -9 AM and 5-7 PM. Also, the hourly trend followed in all the seasons is approximately the same.

**Visualizing Monthly Rental Count across Weather Conditions**



Weather condition 1: Clear,

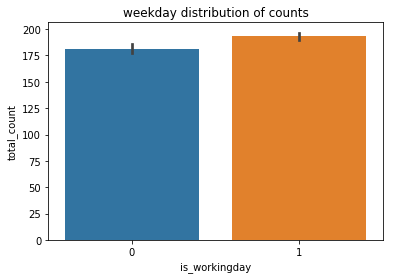
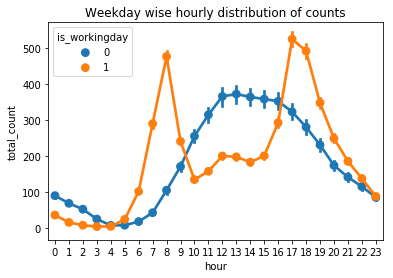
Weather condition 2: Mist

Weather condition 3: Light Snow Light Rain

Weather condition 4: Heavy Rain and Thunderstorms

People enjoy renting and riding bikes during summer when the weather is very clear.

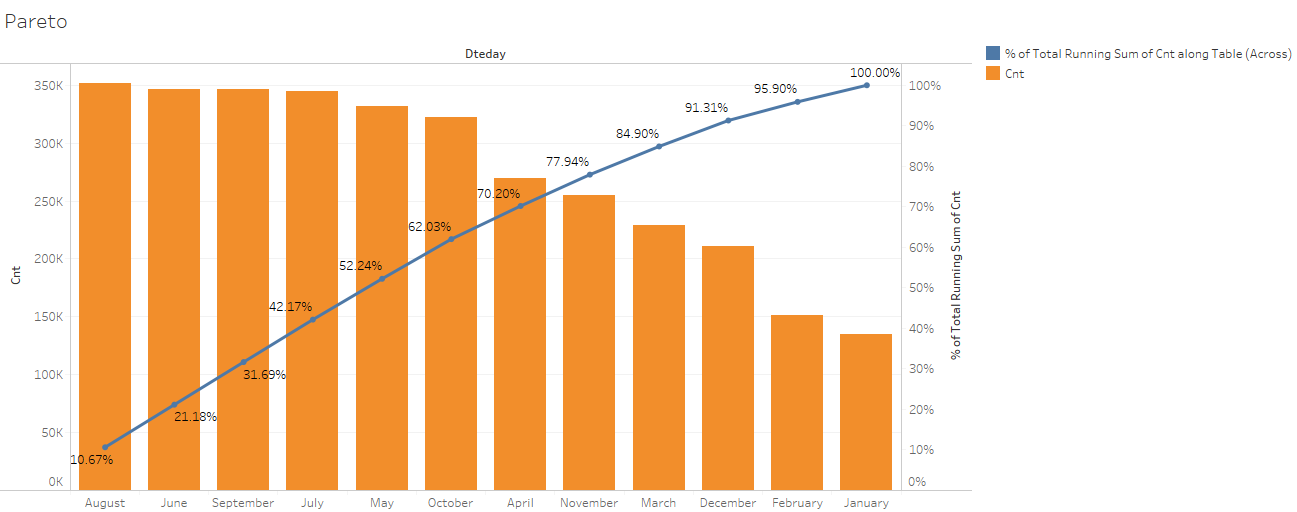
**Visualizing Bike Counts during Working Days**

Bike rental demand is slightly higher during working days as opposed to holidays and weekends only during peak hours i.e. 5-9AM and in the evening after 5PM, this might be due to the surge prices on weekends.

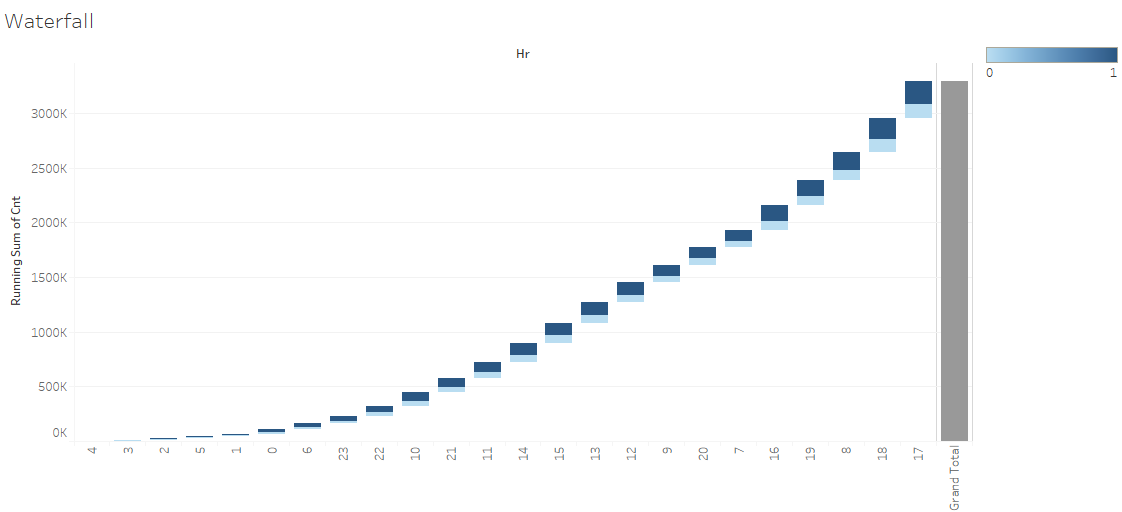
### TABLEAU VISUALIZATIONS

**Variation in Bike Rental Count during various Months**

****

70% of the total bike rental count was happening during August, June, September, July, May, October and April i.e. mainly during fall. This might be due to the clear weather conditions during these months which are convenient for the bikers to ride.

**Variation in Bike Rental Count during a day**

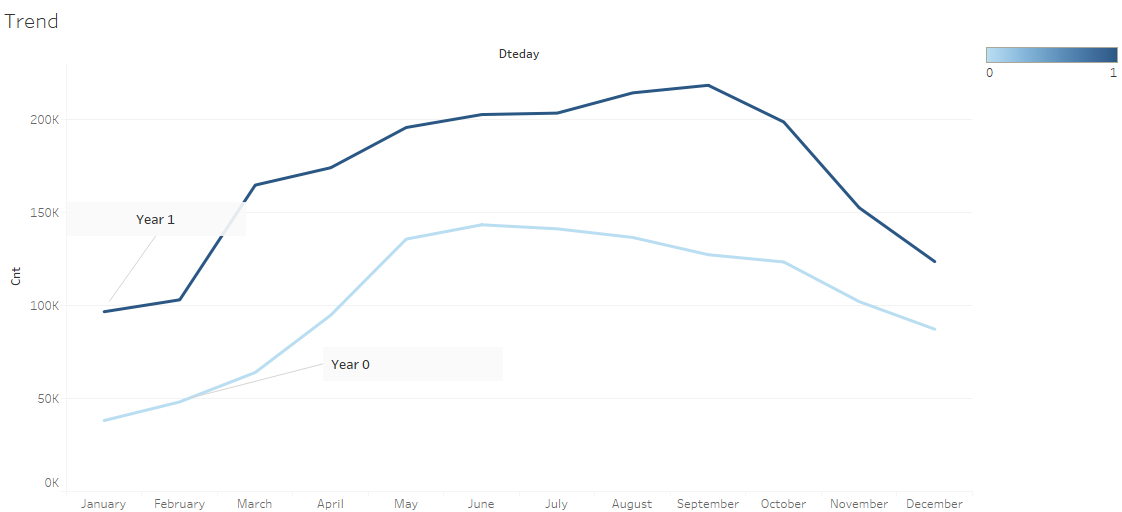


Bike rental count is low during the hour bucket [0,1,2,3,4,5,6,23]

Bike rental count is medium during the hour bucket [10,11,12,13,14,15,21,22]

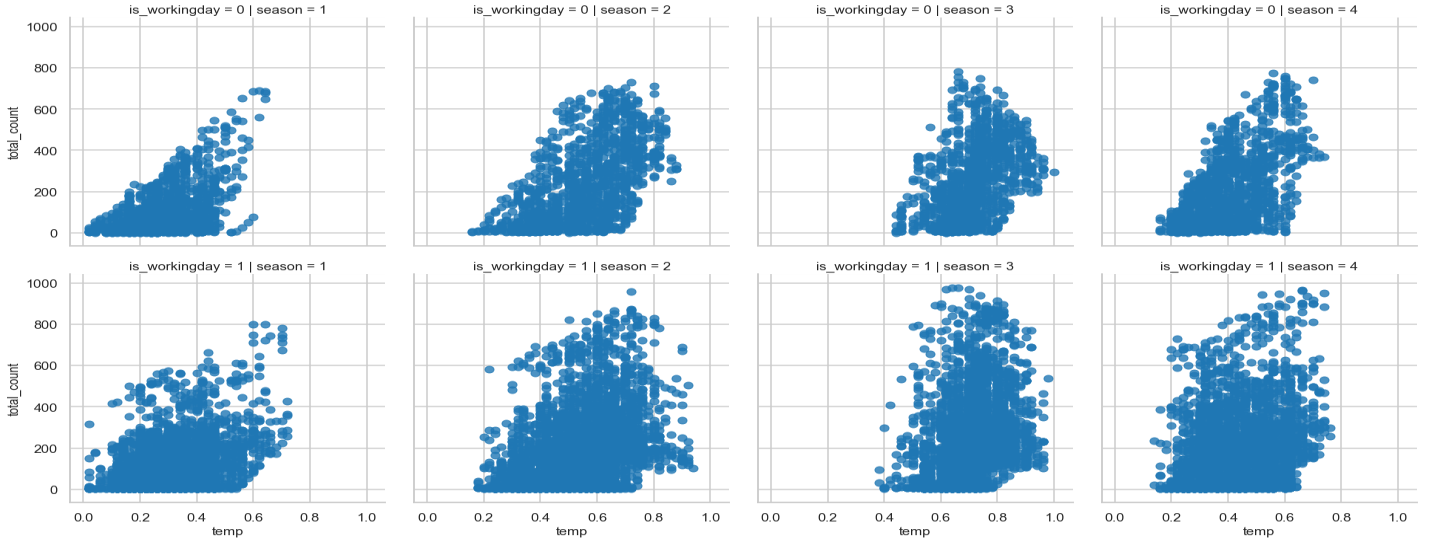
Bike rental count is very high during the hour bucket [7,8,9,16,17,18,19,20]

**Variation in Bike Rental Count over 2 years  
0: 2011, 1:2012**



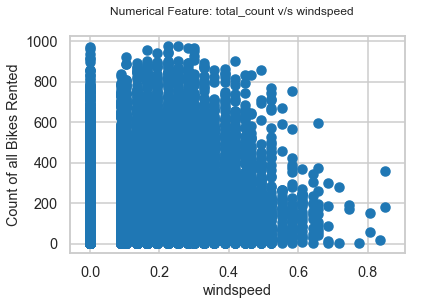
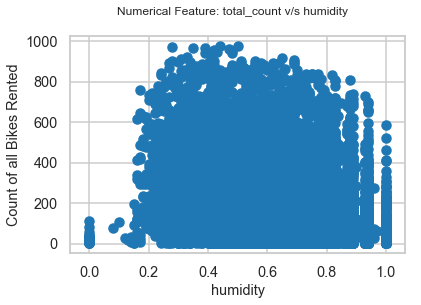
There is rapid increase in bike rental count in 2102, highest during September

**Distribution of Count with respect to Temperature and Working days**



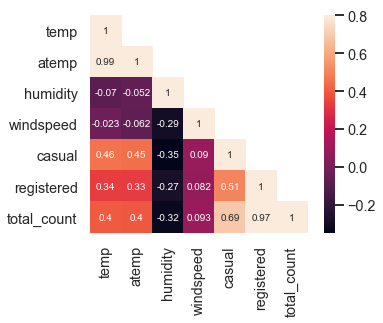
So, people prefer renting bikes as the day get hotter and during working days

**Distribution of Count with respect to Humidity and Wind speed**

Humidity and windspeed are not very significant to determine the bike rental count

**Correlation Plot**



There is no strong correlation between total count and temperature, humidity, windspeed.

There is multicollinearity in the data

Atemp and temp are highly correlated. There is correlation between actual temperature and feel temperature

Casual and registered columns are also correlated to total count

**HYPOTHESIS TESTING**

**To check if the bike rental count is dependent on the season**

**Null Hypothesis**: There is no dependency between bike rental count and season

**Alternate Hypothesis:** There is dependency between bike rental count and season

Testing with 1-way ANOVA and by observing p-value 7.4e-2 (< 0.05) we can reject null hypothesis so there is significant dependency between season & count

**To check if the bike rental count is dependent on the Weather condition**

**Null Hypothesis**: There is no dependency between bike rental count with respect to weather.

**Alternate Hypothesis**: There is dependency between bike rental count with respect to weather

Testing with 1-way ANOVA and by observing p-value 5.791e-65 (< 0.05), There is significant dependency between Weather condition and bike count

**To check if the bike rental count is dependent on the weekday and month**

**Null Hypothesis**: There is no dependency between bike rental count with respect to weekday and month

**Alternate Hypothesis**: There is dependency between bike rental count with respect to weekday and month

Testing with 1-way ANOVA and by observing p-value 9.391e-05 (< 0.05), 2-way ANOVA With the change in weekday and month there is significant change in bike rental count

Among the Categorical variables season, weather conditions, month and weekday are significant in determining the bike rental count

### INFERENCES FROM EDA

* Log transformation can be used to make the distributions close to normal and to reduce the outliers
* Among the numerical features datetime, casual and register, rec\_id is not significant in determining the total count
* Bike rentals are independent of the windspeed and the humidity, because they are almost constant over the months
* As a conclusion we can say, that the amount of bike rentals depends mainly on the weather, real and feel temperature, season and holidays.
* The analysis shows that there is a positive relationship between the amount of bike rentals and temperature.
* The mean amount of bike rentals increases and decreases with the temperature. So, people mainly rent bikes on nice days and nice temperature. This could be important of planning new bike rental stations.

### MODEL BUILDING

The following ML models are used in predicting total rental bike count

Linear Regression

DecisionTreeRegressor

RandomForestRegressor,

GradientBoostingRegressor,

BaggingRegressor

Lasso

Ridge

Evaluation Metrics: R Square/Adjusted Square

Root Mean Square Error

Mean Absolute Percentage error

### NORMALIZATION

Normalization is done change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information.

Numerical columns in this data set are normalized using Zscores

### TEST TRAIN SPLIT

X = data [['season', 'yr', 'mnth', 'hr', 'holiday', 'weekday', 'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed']]

y = data['cnt']

train, test = train\_test\_split (data, test\_size = 0.3, random\_state = 1)

X\_train\_ed = train.drop (['cnt', 'hr', 'temp', 'atemp'], axis = 1)

X\_test\_ed = test.drop(['cnt', 'hr', 'temp', 'atemp'], axis = 1)

### LINEAR REGRESSION

Linear Regression establishes a relationship between **dependent variable (Y)** and one or more **independent variables (X)** using a **best fit straight line** (also known as regression line).

It is represented by an equation **Y=a+b\*X + e**, where a is intercept, b is slope of the line and e is error term. This equation can be used to predict the value of target variable based on given predictor variable(s).

Best fit Line can be obtained by Least Square Method. It is the most common method used for fitting a regression line. It calculates the best-fit line for the observed data by minimizing the sum of the squares of the vertical deviations from each data point to the line. Because the deviations are first squared, when added, there is no cancelling out between positive and negative values.

In case of multiple independent variables, we can go with **forward selection**, **backward elimination** and **step wise approach** for selection of most significant independent variables.

Assumptions:

* There is linear relationship between independent and dependent variables
* There is little or no multicollinearity between the independent variables
* There is no autocorrelation and heteroskedasticity
* There exists Multivariate normality

Building OLS, Using Stepwise selection, it is found that the variables which are significant to determine the total count are **'atemp', 'hum', 'hr', 'yr', 'season', 'weekday', 'windspeed'**

atemp has p-value 0.0

hum has p-value 0.0

hr. has p-value 0.0

yr has p-value 7.62851e-94

season has p-value 5.35531e-65

weekday has p-value 7.2361e-05

windspeed has with p-value 0.000328987

Evaluation metrics:

**Mean Absolute Error**: 0.3214019119553735

**R-Squared**: 0.814

**Root Mean Squared Error (RMSE**): 0.43454933194950063

**Adj. R-squared:**0.814

Building the OLS model with hum, hr\_bkt, mean\_temp, yr, season, weekday, holiday

Workingday

**R-squared**:0.688

**Adj. R-squared**:0.687

Since the adjusted-r square value we are getting on using the data with engineered features is

quite lower as compared to when we are using the original features, we will now use just the

orginal features, i.e. the X\_train\_org and the X\_test\_org.

After Checking for the Multicolinearity and VIF values the significant variables in determining

the total bike rental count are:

**'atemp', 'hum', 'hr', 'yr', 'season', 'weekday', 'windspeed'**

### DECISION TREE REGRESSION

Decision tree builds regression models in the form of a tree structure It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The result is a tree with **decision nodes** and **leaf nodes**. A decision node has two or

more branches each representing values for the attribute tested. Leaf node represents a decision

on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.

Building Decision Tree Regressor on Xtrain and ytrain, and testing on the Xtest, ytest following

are the values of evaluation metrics

**Mean Absolute Error**: 0.19272202180522513

**R-Squared**: 0.9128626542808281

**Root Mean Squared Error (RMSE**): 0.295190355057837

**Accuracy Score**: 0.9128626542808281

### RANDOM FOREST REGRESSION

The Random Forest is ab Ensemble model which is one of the most effective machine learning

models for predictive analytics.

The **random forest** model is a type of additive model that makes predictions by combining decisions from a sequence of base models. More formally we can write this class of models as:

g(x)=f0(x)+f1(x)+f2(x)+...

where the final model g is the sum of simple base models fi. Here, each base classifier is a simple [decision tree](https://turi.com/learn/userguide/supervised-learning/decision_tree_regression.html). This broad technique of using multiple models to obtain better predictive performance is called an Ensemble model. In random forests, all the base models are constructed independently using **a different subsample** of the data.

Building Random Forest Regressor on Xtrain and ytrain, and testing on the Xtest, ytest following

are the values of evaluation metrics

**Mean Absolute Error**: 0.1625426689305493

**R-Squared**: 0.9355434065302716

**Root Mean Squared Error (RMSE):** 0.2538830310787399

**Accuracy Score**: 0.9355434065302716

### GRADIENT BOOSTING REGRESSOR

Gradient Boosting for regression 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.

The objective of this algorithm is to minimize the loss of the model by adding weak learners using a gradient descent like procedure. This class of algorithms was described as a stage-wise additive model. This is because one new weak learner is added at a time and existing weak learners in

the model is frozen and left unchanged.

The loss function used depends on the type of problem being solved. It must be differentiable. Regression may use squared error. Decision trees are used as the weak learner in gradient boosting.

Specifically, regression trees that output real values for splits and whose output can be added together are used, allowing subsequent models outputs to be added and “correct” the residuals in the predictions. Trees are constructed in a greedy manner, choosing the best split points based on purity scores.

Trees are added one at a time, and existing trees in the model are not changed. A gradient descent procedure is used to minimize the loss when adding trees. After calculating error or loss, the weights are updated to minimize that error.

When a new test data is given to the model, each model in the ensemble makes prediction and averaging is used to predict the test data

Building Gradient boosting Regressor on Xtrain and ytrain, and testing on the Xtest, ytest following

are the values of evaluation

metrics

**Mean Absolute Error**: 0.14783687302458953

**R-Squared**: 0.9469873274194586

**Root Mean Squared Error (RMSE**): 0.23024481010555142

**Accuracy Score**: 0.9469873274194586

### BAGGING REGRESSOR

A Bagging regressor is an ensemble estimator that fits base regressors each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. Such subsets of original data are called bootstrap samples Such a meta-estimator can typically be used to reduce the variance of a black-box estimator (e.g., a decision tree), by introducing randomization into its construction procedure and then making an ensemble out of it.

When random subsets of the dataset are drawn as random subsets of the samples, then this algorithm is known as Pasting. If samples are drawn with replacement, then the method is known as Bagging.

Building Bagging Regressor on Xtrain and ytrain, and testing on the Xtest, ytest following

are the values of evaluation metrics

**Mean Absolute Error**: 0.15377205529354623

**R-Squared**: 0.9416919429378585

**Root Mean Squared Error (RMSE**): 0.24147061324753671

**Accuracy Score**: 0.9416919429378586

### LASSO AND RIDGE

**RIDGE**

Removing predictors from the model can be seen as settings their coefficients to zero. Instead of forcing them to be exactly zero, penalizing them if they are too far from zero, thus enforcing them to be small in a continuous way. This way model complexity is decreased while keeping all variables in the model. This, basically, is what Ridge Regression does.

In Ridge Regression, the OLS loss function is augmented in such a way that we not only minimize the sum of squared residuals but also penalize the size of parameter estimates, in order to shrink them towards zero:

https://res.cloudinary.com/dyd911kmh/image/upload/f_auto,q_auto:best/v1543418449/eq7_ylxudw.png

Solving this for β^β^ gives the t ridge regression estimates β^ridge=(X′X+λI)−1(X′Y)β^ridge=(X′X+λI)−1(X′Y), where I denotes the identity matrix.

The λ parameter is the regularization penalty

* As λ→0,β^ridge→β^OLSλ→0,β^ridge→β^OLS;
* As λ→∞,β^ridge→0λ→∞,β^ridge→0.

So, setting *λ* to 0 is the same as using the OLS, while the larger its value, the stronger is the coefficients' size penalized.

Building Ridge on Xtrain and ytrain, and testing on the Xtest, ytest following

are the values of evaluation

metrics

**Mean Absolute Error**: 0.5687040083812681  
**R-Squared**: 0.47437241889724635

**Root Mean Squared Error (RMSE**): 0.7250017800686792

**Accuracy Score**: 0.47437241889724635

**LASSO**

Lasso, or Least Absolute Shrinkage and Selection Operator, is quite similar to ridge regression. It also adds a penalty for non-zero coefficients, but unlike ridge regression which penalizes sum of squared coefficients (the so-called L2 penalty), lasso penalizes the sum of their absolute values (L1 penalty). As a result, for high values of λ, many coefficients are exactly zeroed under lasso, which is never the case in ridge regression. The only difference in ridge and lasso loss functions is in the penalty terms. Under lasso, the loss is defined as:

https://res.cloudinary.com/dyd911kmh/image/upload/f_auto,q_auto:best/v1543418448/eq11_ij4mms.png

Building Lasso on Xtrain and ytrain, and testing on the Xtest, ytest following

are the values of evaluation

metrics

**Mean Absolute Error**: 0.603887808041848

**R-Squared**: 0.42628176539936935

**Root Mean Squared Error (RMSE**): 0.7574419017988314

**Accuracy Score**: 0.42628176539936

## CHAPTER4

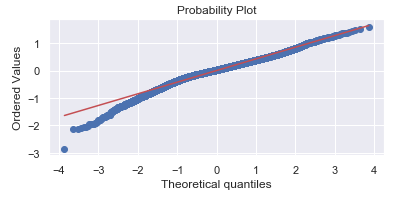
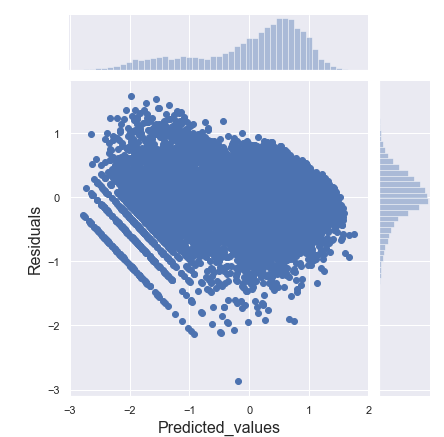
### RECOMMENDATIONS

### MODEL COMPARISION

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MODEL** | **R-SQUARE** |  | **RMSE** | **MAE** |
| **Linear**  **Regression** | **81.4** |  | **0.43** | **0.32** |
| **Decision Tree**  **Regression** | **91.28** |  | **0.29** | **0.19** |
| **Random**  **Forest** | **93.7** |  | **0.25** | **0.15** |
| **Gradient**  **Boosting** | **94.7** |  | **0.22** | **0.14** |
| **Bagging** | **94.1** |  | **0.24** | **0.15** |
| **Lasso** | **42.6** |  | **0.75** | **0.6** |
| **Ridge** | **47.4** |  | **0.72** | **0.56** |

Comparing various Regression models Gradient boosting is giving beast accuracy and least RMSE, MAE values, hence this model can used in predicting the total bike rental count on any given day based on the environment and seasonal settings

Linear Regression-Residual and QQ plot



Checking the residual plot and QQ plot, we can see that the residuals have a pattern, and are not normally distributed, which means the linear model doesn’t fit the data so well.

## CONCLUSION

Amount of bike rentals depends mainly on the weather and temperature. The EDA analysis shows that there is a positive relationship between the amount of bike rentals and temperature. So, people mainly rent bikes on nice days especially during Fall and nice temperature. This could be important for Bike sharing on planning new bike rental stations.

Bike rental count is very high during the hour bucket [7,8,9,16,17,18,19,20], hence these bike sharing systems can increase the rental bike availability during these hours and charge their customers accordingly

## BIBILOGRAPHY

Various References used:

<https://www.analyticsvidhya.com>

<https://www.datacamp.com/community/tutorials/tutorial-ridge-lasso-elastic-net>

<https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/normalize-data>

<https://rstudio-pubs-static.s3.amazonaws.com/98475_c0697af6a9b045239a367f9190fdb12d.html>