CSE4021
Machine Learning
J Component

Bike Sharing Count Prediction Using Machine Learning

F1 Slot

Submitted by:

Kashish Mittal 18BCE0919

Saakshi Mittal 18BCE0985

Submitted to:

Prof. GOPINATH MP

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

TABLE OF CONTENTS

S.No.	Topic	Page No.
1.	Abstract	3
2.	Introduction	3
3.	Literature Survey	4
4.	Methodology	6
5.	Architectural Models	7
6.	Results and Experiments	8
7.	Ensemble Learning Method Used- Bagging	18
8.	Conclusion And Future Work	20
9.	Code	20
10.	References	35

1. Abstract

Bike Renting Systems are the latest trend that are introduced into the market lately replacing the old way renting bikes. These systems made it easy for a customer to rent a bike within seconds and ease the services where pick up and dropping them is very easy. These bike sharing systems are very useful as they accurately record the travel duration, arrival and departure positions when compared to other transport services which is done by using virtual sensor network using which we can sense the location of the vehicle anywhere in the city. In this project, based on certain attribute such as weather, specific hour and all the day information we predict the hourly bike count which is the main motto of this paper. Here it is a step by step process where we initially do the Descriptive Analysis, then based on it we do Missing Value Analysis, then we do Outlier Analysis, then Correlation Analysis, then the most important step where for the given data we do a Model Selection where in our case we get that the Random Forest Regression is the best for the given data set and scenario. Then finally we perform Random Forest Model and at last we plot a graph of Feature importance and conclude the necessary attributes.

2. Introduction

Now a days we are living in a world with advanced technology. We can see the growth of technology in each and every sector. In hugely populated cities we see the biggest problem arising these days is traffic. This traffic problem we can even see in our country India as we are the second highly populated country in the world. In cities like Bangalore, Delhi, Chennai, Hyderabad, etc, we see that government is trying to rectify this traffic problem by introducing various modes of public transport, most recently the Metro.

These things need more and more investment and takes time for establishment for long years. Even though these solutions do not rectify the problem entirely. If you see the most of these cities are with IT employees and only some of them use these metro and other public transport if their destination is very far. But if their destination is a little bit nearer they might need to use their vehicle itself.

But most of these IT workers are either from middle class families, they cannot afford to buy and maintain their vehicle and upon this the increase in petrol rates further creates problems. So to rectify this kind of problems, we use this Bike Sharing as a solution.

Where this idea of bike sharing is completely different as now a day we are living in a automated world where this new generation of bike sharing is introduced in which all the work starting from the membership, taking the bike for rent and returning it back all are made automatic by which we save time, the availability is also known and the services got better in which we don’t need any third party vendor and even the payment is also made online which is a completely trustable and efficient way of renting bikes.

3. Literature Survey

[1] Analysis of Dublin bike data can be used in providing some insights into bike usage patterns. In this paper, they have performed clustering analysis and identified interesting clusters at the busiest and quietest bike stations. Examining the number of available bikes every 10 minutes, the results showed that at the stations analysed there can be significant differences in bike usage at weekends and weekdays, during business hours and after work. The different patterns observed from the results can be explained based on geographical knowledge regarding the locations. Testing with new data suggests pattern consistency.

[2] In this papers an analysis is made on the social practices which will emerge as a part of the Bike Rental Systems which they have introduced as an alternate urban mobility. Here their main aim is to help the pedestrians in converting into cyclists using this pivotal bike sharing service as a device that will be an intermediate between the pedestrian and the bike sharing services. This system also makes sure that standardization in a technical manner is there and also reveals the differences between the people who are familiar and the people who are not familiar with this bike rental systems simultaneously. The main aspects that they discussed here are the Self service bicycles, Intermodality, Transportation, Bike Rental System and the Modal Shift.

[3] As we know for any prediction of the bike rental system, land is the most important factor that has the highest influence in predicting its model. In this paper, they have predicted the appropriate model as follows. Initially they gathered the data of the walking distance by riders from a rental station for returning the public bicycle to their destinations was collected on which they considered 85 percentage as a statistical value which was used as a factor of influence to those stations. Later a relationship model was constructed. Finally they compared the old rental model with the new one by testing both the models. Where the final result is showing the daily rental demand.

[4] As we see in most of the developed countries and some developing countries we see the increase of the transportation using bicycles sharing systems, with this increase there are many challenges that are faced by the bike sharing systems such as dock or parking shortages and unavailability of bicycles. in this paper, the main objective is to develop a prediction model based on the increasing demands of the customers using the rental data that will balance the needs and the bicycle operations. Initially various methods that can be used to process and collect the required data are proposed. Later for both the station based and the trip based aspects, bicycle usage patterns were studied to provide some user demand prediction guidance. Later, using back propagation neural network which is used for creating demand prediction models for all the services of various stations, cluster analysis which is used to identify various services proposed in different stations and comparative analysis which is performed to find whether the accuracy is improved for the prediction models by various factors such as working or non working days,distinction between stations,. All three were combined to predict the customer's demand. Finally we find the performance for all the proposed methodologies by conducting a case study for evaluation. The results conclude that the factors, working and non working days, distinction between stations will improve the accuracy for all the prediction models.

[5] In this paper, the implementation of the Bike Sharing and the Bike Rental application is created where they divided their objective into two categories Bike Sharing and Bike Rentals where they used PIECES Analysis to sort out their major problems faced during creation of this application where it involves, System Performance Analysis, Information Analysis, Economic Analysis, Control Analysis, Efficiency Analysis and Service Analysis. Later on these basis they developed a Spiral Model involving Communication, Planning, Modelling, Construction and Development. Using this application, visitors can go to any tourist places and rent a bike comfortably and quickly by using this web based technology. Using this application we can even ease the payment method safely and quickly without even using any cash. Further this application can display a report from the data of the bicycle very accurately and quickly with its financial statement recording for the further needs of the company.

[6] In this paper, they have taken one of the challenges that a public bike renting application that they created which is facing a problem in the effective planning of the usage of resources. Here they analyzed the Dublin bike renting scheme by using some statistical and data mining methods. They initially collected the data, later they done an exploration analysis where they find the bike stations and the usage of the rentals in that particular area by plotting these areas on a map and then finally analyze the bike patterns with respect to time and then performing cluster analysis. The obtained result is used as an initialization for developing a good prediction analysis.

[7] In this paper, they have created an application that will help people to in moving to any place by using a bike sharing system. This application helps us to find the two best nearer bike stations where the user can pick up the city bike or just to leave it. This even further helps in making the probability of finding the available bikes maximum and minimize the walking distance, etc. They have used Spatio temporal prediction algorithm in developing this application which further helps in estimating the availability of the bikes and outperforming all the already developed solutions. Here using this prediction they built an spatial underlying network between various bike stations which also consists of various temporal patterns. Here their application was tested with the Dublin Bike Sharing System which is a real time dataset.

[8] Predicting the bike sharing demand is very useful as it will further help us in finding the location of the bikes and also makes sure that there is a more movement of the bikes for the users. In this paper, they have introduced a real time method which will help in predicting the bike returning and renting in various places in a city. Considering the data collected on a certain period based on time, weather and history as its features using which we construct a network of the trips taken by the bike, and from that data we will do a community detection where we take two communities from it that contains the data of various stations. They used LSTM model for training with two layers using which we predict the bike sharing. Further they used the gating mechanism to process the sequential data of the neural recurring network. Using this they found the Root Mean Squared Error and predict a proposed model that outperforms other deep learning models by just comparing their RMSE.

[9] In this paper they have introduces a predictive model and analyze the usage of bike using a bike sharing system.Here they have used deep learning where they specifically used the convolutional neural network to predict the daily use of bike sharing at both the station and city level. This convolutional neural networks also related with the parameters nearest neighbourhood station, patch size, temporal window, and the learning ratio. By using all these things they predict the bike sharing system.

[10] In this paper, they used the visualized data using visualization technology and find out all the possible factors that have an impact on the number of users that are using the bike sharing service. Initially they analyze the data where they take the feeling temperature, season, wind speed, weather sit and humidity as the important factors that has a great impact o the number of users directly. Later, they use the NN model, DELM model, Regression Model and ELM model using which we predict the possible number of users of the bike sharing system.

4. Methodology

Initially for the given data set we do Descriptive Analysis where we split the data set for validation, training and testing. Later we will check whether there are any NULL values in the data which is known as Missing Value Analysis. Then we do the Outlier Analysis where we create Box Plots and Remove all the outliers from the data. After that we do the Correlation Analysis. Based on the result of the Correlation Analysis we dismiss some attributes and later then, we do a Model Selection and later the most important step in which we perform Random Forest Regression,based on the output from the Model Selection. This Random Forest Regression can be done using many ways, but here in our case we use two methods to divide the data for Random Forest Regression. The first method is we implement Random Forest Regression on the Training, Validation and Testing Datasets, similarly we can also use the K Fold Cross Validation method where in our scenario we took K as 3, hence Three Fold Cross Validation can be done using which we split the data into three, and later implement this Random Forest Regression on each split. Finally we check the Feature importance of the attributes for both the methods and conclude the important and prominent attributes that effect the target attribute or Response Variable.

Dataset used: https://archive.ics.uci.edu/ml/datasets/bike+sharing+dataset

Sample Data Set:

instant	dteday	season	yr	mnth	holiday	weekday	workingda	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
1	01-01-2011	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
2	02-01-2011	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
3	03-01-2011	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
4	04-01-2011	1	0	1	0	2	1	1	0.2	0.212122	0.590435	0.160296	108	1454	1562
5	05-01-2011	1	0	1	0	3	1	1	0.226957	0.22927	0.436957	0.1869	82	1518	1600
6	06-01-2011	1	0	1	0	4	1	1	0.204348	0.233209	0.518261	0.089565	88	1518	1606
7	07-01-2011	1	0	1	0	5	1	2	0.196522	0.208839	0.498696	0.168726	148	1362	1510
8	08-01-2011	1	0	1	0	6	0	2	0.165	0.162254	0.535833	0.266804	68	891	959
9	09-01-2011	1	0	1	0	0	0	1	0.138333	0.116175	0.434167	0.36195	54	768	822
10	10-01-2011	1	0	1	0	1	1	1	0.150833	0.150888	0.482917	0.223267	41	1280	1321

5. Architectural Model

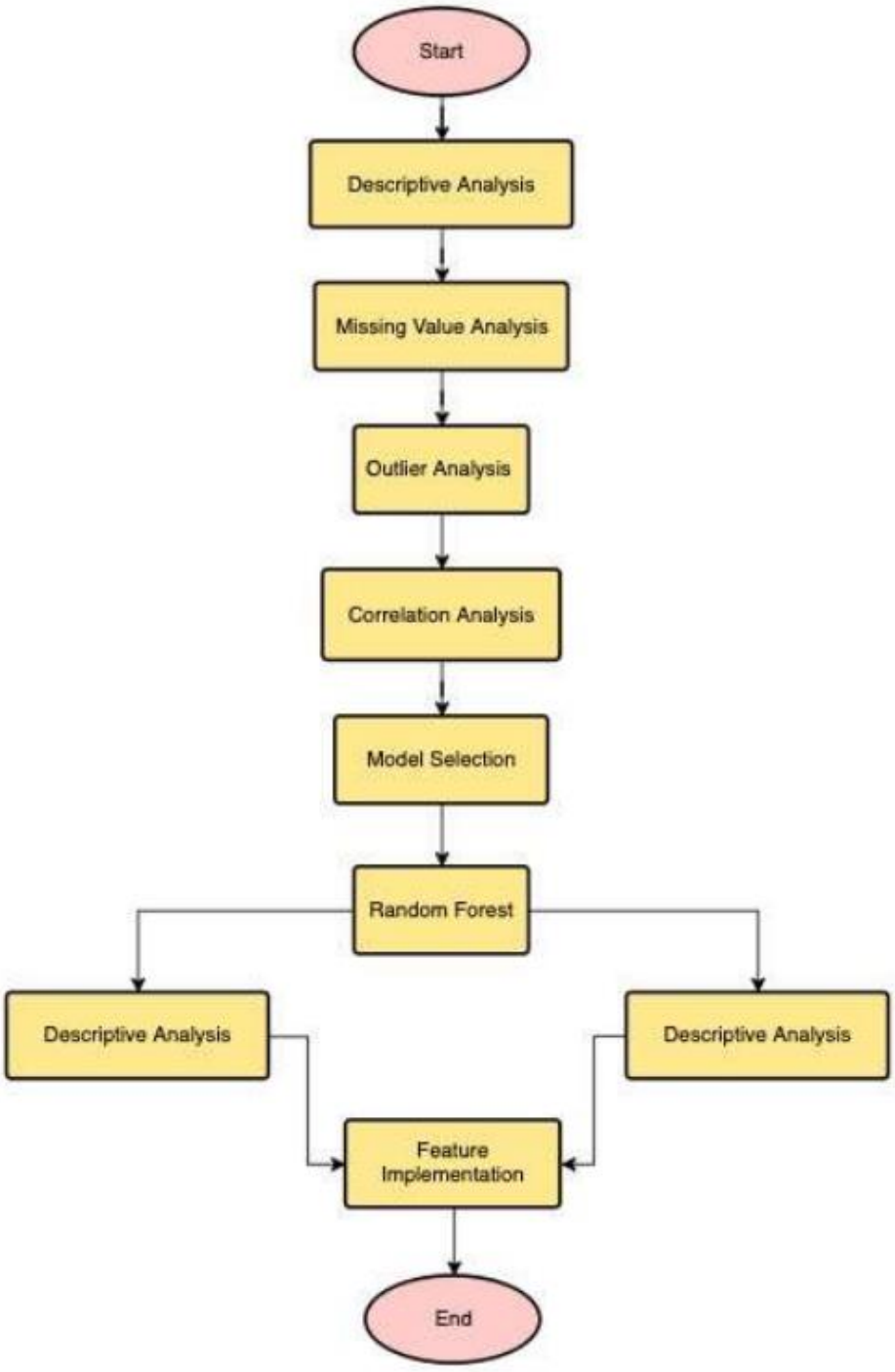


Figure 1: Architecture

6. Results and Experiments

a. Descriptive Analysis

Here initially we load the csv file and divide the attributes based on their feature such as categorical attributes and integer attributes. This spitting will further be helpful for validation, testing and training. Later we find all the statistical information for each integer column. Later for each column of the categorical attributes we find the variables like the count, unique values, top value and the frequency.

	temp	atemp	hum	windspeed
count	17379.000000	17379.000000	17379.000000	17379.000000
mean	0.496987	0.475775	0.627229	0.190098
std	0.192556	0.171850	0.192930	0.122340
min	0.020000	0.000000	0.000000	0.000000
25%	0.340000	0.333300	0.480000	0.104500
50%	0.500000	0.484800	0.630000	0.194000
75%	0.660000	0.621200	0.780000	0.253700
max	1.000000	1.000000	1.000000	0.850700

Figure 2: Data Statistics for each column

	season	holiday	mnth	hr	weekday	workingday	weathersit
count	17379	17379	17379	17379	17379	17379	17379
unique	4	2	12	24	7	2	4
top	3	0	5	17	6	1	1
freq	4496	16879	1488	730	2512	11865	11413

Figure 3: Describe values for Categorical Attributes

b. Missing Value Analysis

This Missing Value analysis will check whether the data set is having the null values (not a number kind of values) or not. If there are any null values for further processing, they need to be replaced. Where in our case, the given data attributes does not have any null values.

```
instant       False
dteday        False
season        False
yr            False
mnth          False
hr            False
holiday       False
weekday       False
workingday    False
weathersit    False
temp          False
atemp         False
hum           False
windspeed     False
casual        False
registered    False
cnt           False
dtype: bool
```

Figure 4: Missing Value Analysis

c. Outlier Analysis

i. Box Plots

Here we create Box plots based on cont where on y-axis we show the count and on x-axis we show the other data such as Month, Weather Situation, Working Day, Hour of the Day and Temperature.

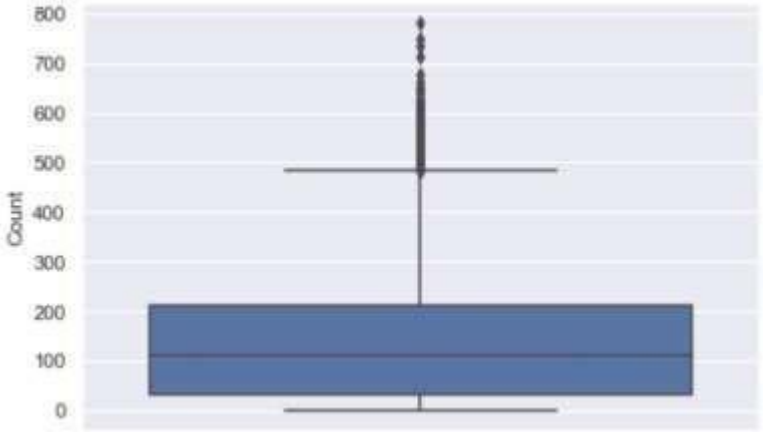


Figure 5: Box Plot On Count

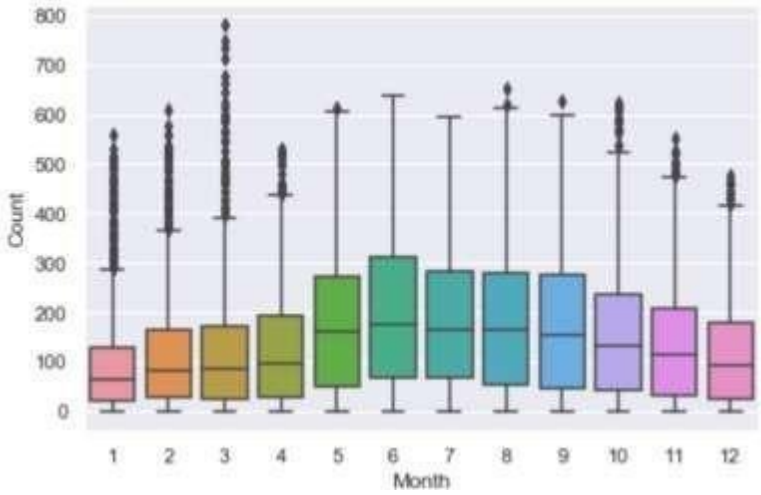


Figure 6: Box Plot On Count Across Months

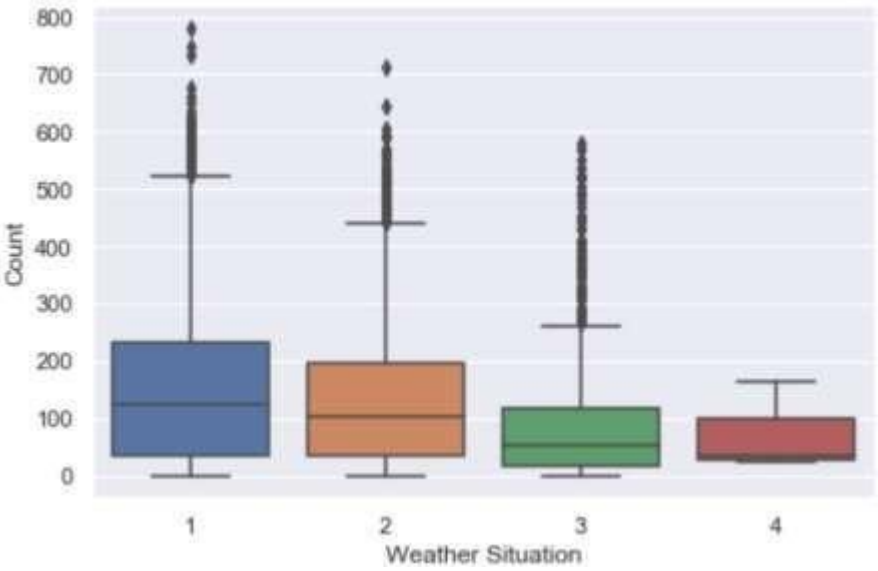


Figure 7: Box Plot On Count Across Weather Situations

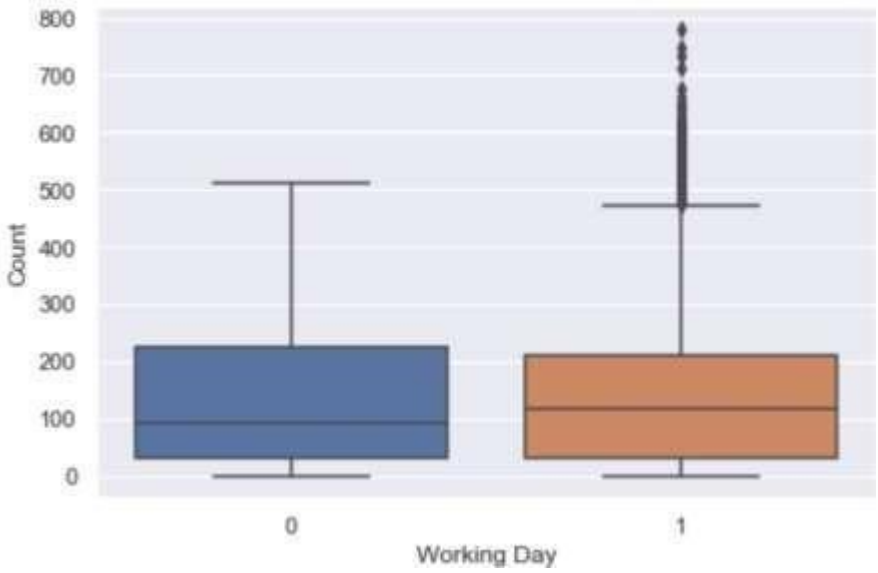


Figure 8: Box Plot On Count Across Working Day

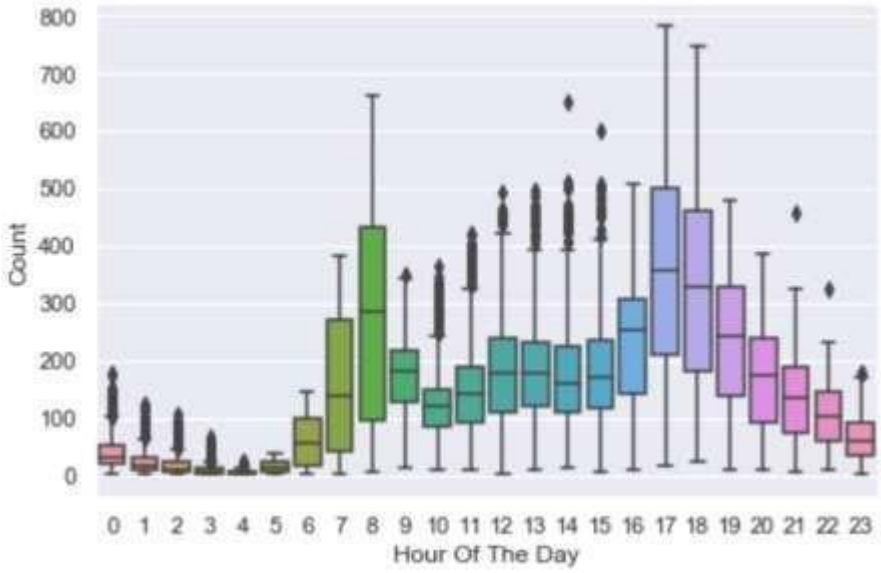


Figure 9: Box Plot On Count Across Hour Of The Day

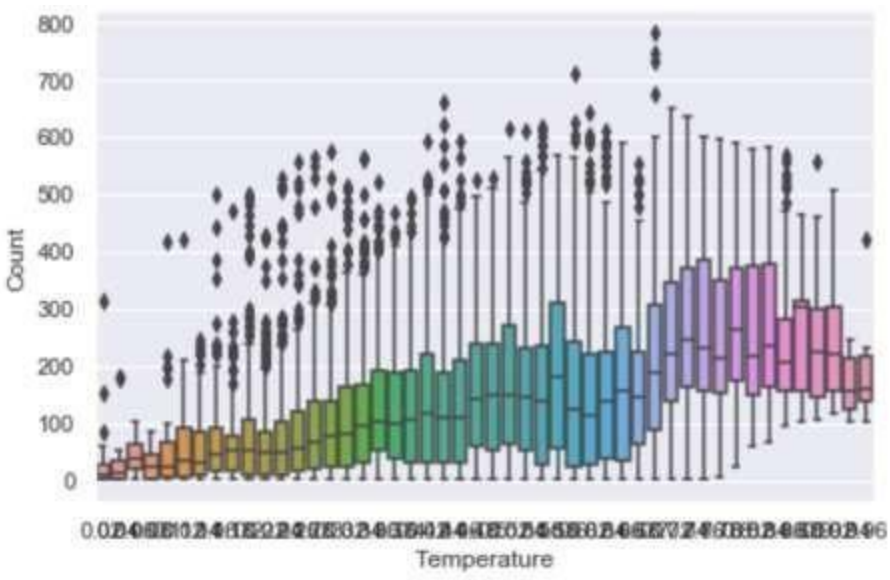


Figure 10: Box Plot On Count Across Temperature

In the above box plots if you consider the holiday and the working day box plots we see that the use of bikes is more during working days when compared to the weekends and holidays. If you check the hourly box plot we see that there are two local maximums at 8 AM and 5 PM which concludes that the customers are using more of the bike rental services at those . If you check the other factor, that is temperature which is very important as we can see that whenever the temperature is high, we see more number of bike rents whereas when the temperature is low, not only the less average number of bike rents but also more outliers in the data.

ii. Removal of outliers from data

Here initially we consider the distribution plot of all the count values where these values are compared with its corresponding normal distribution we do not find any match.

Using IQR(Interquartile Range) and median we remove these outliers of the count values as we see these count values are not fitting into the normal distribution. We can also transform the target values into a normal distribution by using standard deviation and mean which can be an alternative process. Here in the data set we can see the reduce of samples from 10151 to 10427.

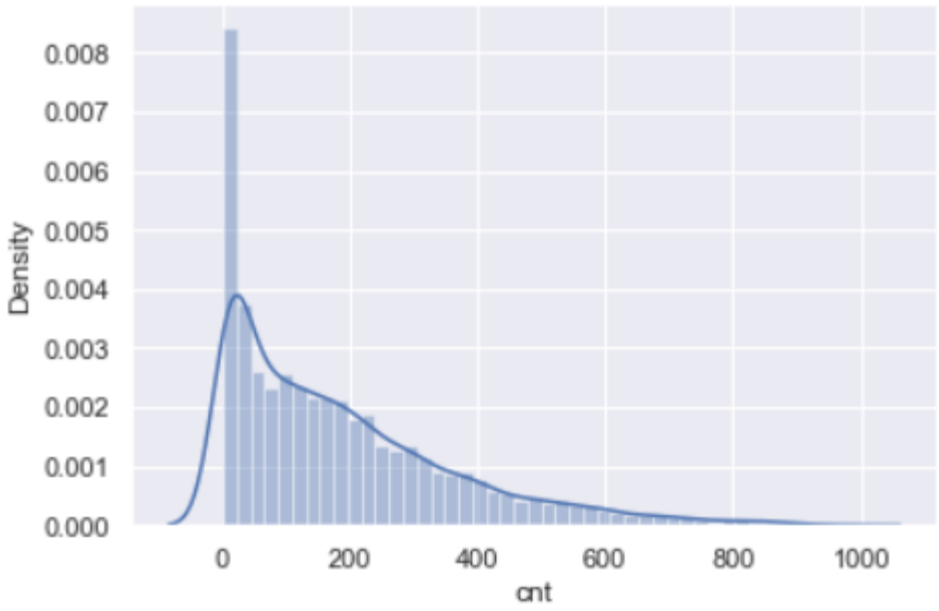


Figure 11: Data With Outliers

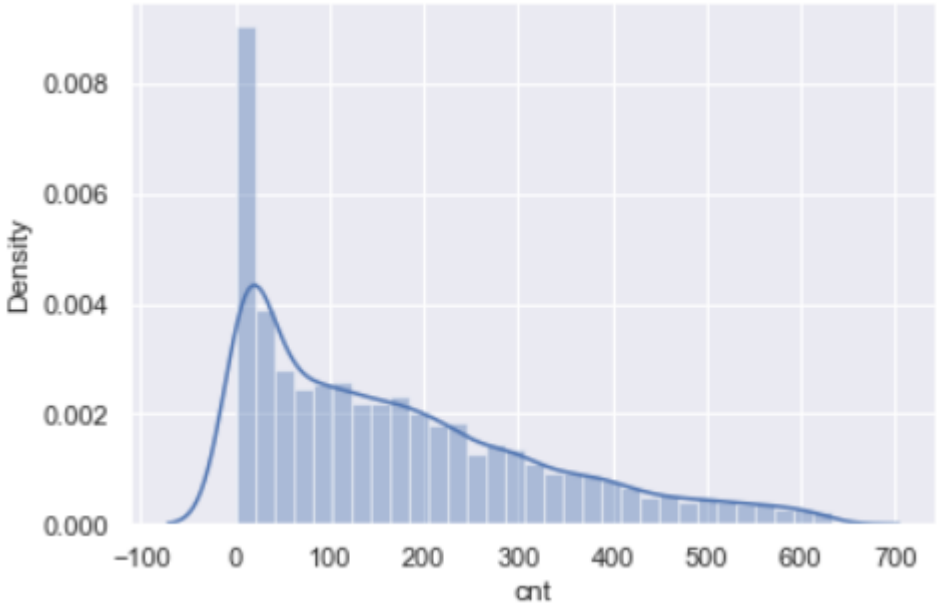


Figure 12: Data Without Outliers

iii. Correlation Analysis

Below we can see a correlation matrix for the numerical features of the data set of bike sharing where we see the most promising variables as temperature("temp") hour("hr"). We can also see the value of correlation is very strong for both feeling temperature("atemp") and temperature("temp"). To avoid collinearity and to reduce the dimensionality and model complexity of the predictive model we dismiss the variable feeling temperature("atemp").

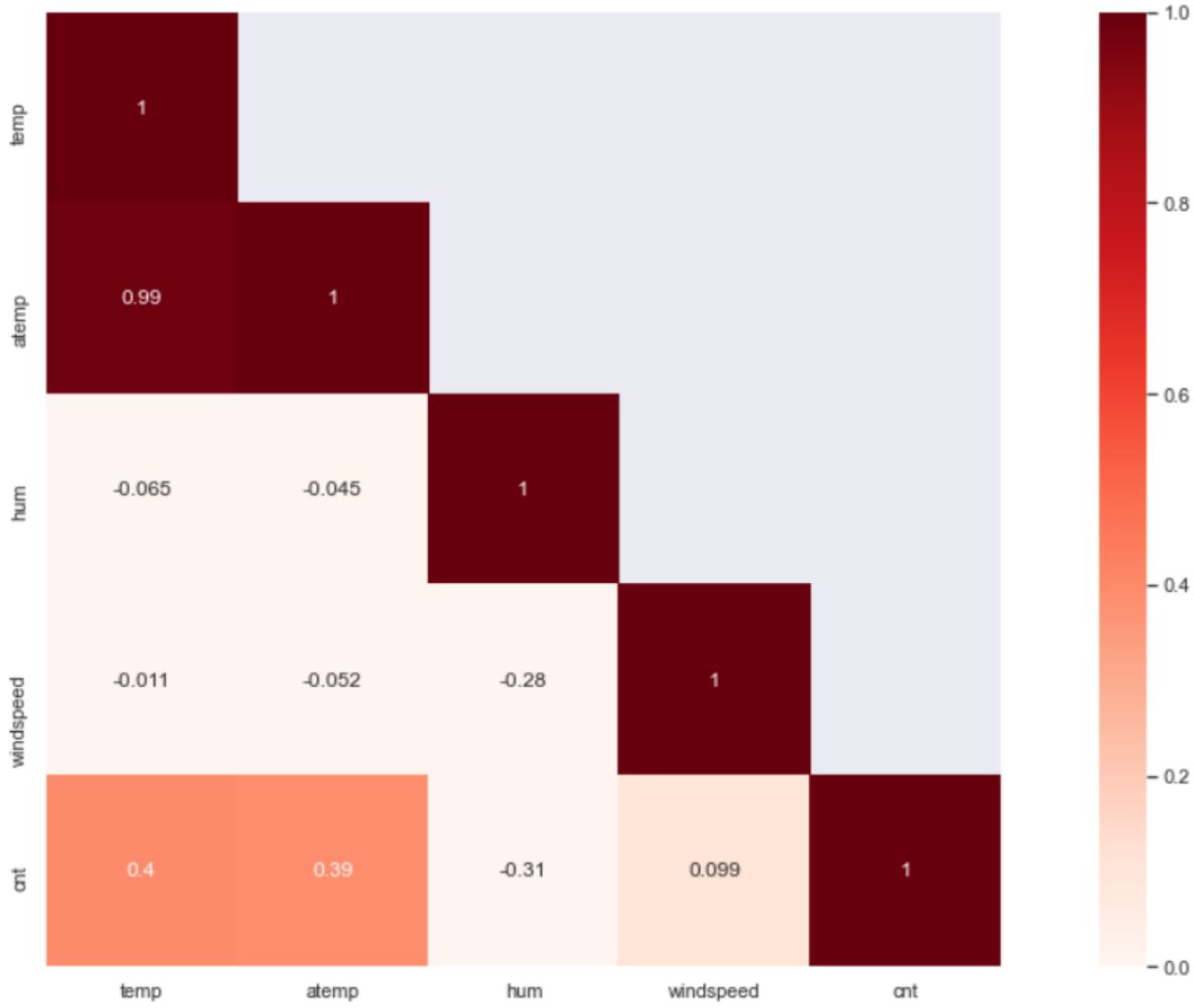


Figure 13: Correlation Matrix For Numerical Feature

d. Model Selection

We need a regression algorithm to predict the count value based on the categorical and numerical attributes. Here the data set is considered small as we see there are less than 20 thousand samples where by considering all the analysis steps we can finally conclude that some of the variables or attributes are particularly significant. Considering all the characteristics we can select the appropriate algorithms where, to find the appropriate algorithm we first consider the algorithms as Support Vector Regression with different types of kernels, Lasso, Random Forest, Elastic Net and Ridge Regression where we find Mean Square Error and the R square score. Where they are defined as follows:

Mean Squared Error (MSE):

$$MSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2}$$

R Square Score:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} e_i^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Figure 14: R Square Score

Here from the above figure 13, if you compare all the values of the Mean Square Error, we see the error is least for the Random Forest Regression whereas if you compare the R square Score it is highest for the Random Forest Regression. Thus, in our scenario, Random Forest Regression is the best regression model for analysis.

e. Random Forest Regression

Random forest regression is an advanced type algorithm using which we generate many number of individual decision trees where for each and every decision tree we feed the observations and after which we will get an output. Here in our case this random forest regression algorithm create around 200 decision trees that are trained using different different sub samples from our data set. Here in below we have done Random Forest Regression for various data sets by defining RandomForestRegression function where we find the values of Mean Square Error, Mean Absolute Error, R Square Score and Root Mean Squared Logarithmic Error where these RMSLE and MAE are defines as follows:

Root Mean Squared Logarithmic Error (RMSLE):

$$RMSLE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\log(x_i) - \log(y_i))^2}$$

Mean Absolute Error (MAE):

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

Now for all the three data sets: Training, Validation and Testing, we perform Random Forest Regression and Find out the MSE, MAE, R Square Score and RMSLE values.

Model	Dataset	MSE	MAE	RMSLE	R^2 score
RandomForestRegressor	training	291.74	10.80	0.21	0.98
RandomForestRegressor	validation	18910.27	96.29	0.47	0.59
RandomForestRegressor	testing	19852.06	96.14	0.51	0.59

Figure 15: Random Forest Regression for Training, Validation and Testing Datasets

In the above Figure 14, we have the random forest regression analysis where we find the MSE, MAE, RMSLE and R Square Score for Training, Validation and Testing Datasets. Where if you see the value of MSE, MAE, RMSLE are small and R Square score is larger for Training Data Set which concludes that the training Dataset is the best as it has more accuracy when compared to Validation and Testing datasets.

f. Random Forest Regression Using K-Fold Cross Validation

There is another way of solving Random Forest Regression using K Fold Cross Validation where here let us consider K as 3. Here we split the dataset into 3 parts where for each split, we perform the Training, Validation and Testing and find MSE, MAE, RMSLE and R Square Score for each split. To improve the accuracy of prediction we consider averaging and this averaging we also implement to control over fitting which here we name it as Mean.

Model	Split	Mean Squared Error	Mean Absolute Error	RMSLE	R^2 score
RandomForestRegressor	1	4489.95	43.72	0.41	0.86
RandomForestRegressor	2	4636.60	44.60	0.41	0.86
RandomForestRegressor	3	4691.67	44.57	0.41	0.86
RandomForestRegressor	Mean	4606.07	44.30	0.41	0.86

Figure 16: Random Forest Regression for 3 Fold Classification Dataset

In the above Figure 15, if you see the final model has a Mean Absolute Error of 44.30 which is an average of all the three splits that we derived from the Three Fold Cross Validation. And if you want to find the best split, in this case it is split 1.

g. Feature Importance

Feature importance is a process where we initially give feature rankings to all the features or attributes except 'count' where here also there are two parts as if you consider the process of Random Forest Regression for Training, Validation and Testing, we consider the validation dataset to finding the feature rankings and using it we plot the feature importance as a bar graph and the feature rankings and feature importance are as follows:

```
Feature ranking:
1. feature hr (0.612987)
2. feature temp (0.153037)
3. feature hum (0.061663)
4. feature workingday (0.044289)
5. feature windspeed (0.031500)
6. feature weathersit (0.024915)
7. feature mnth (0.023815)
8. feature weekday (0.022891)
9. feature season (0.021279)
10. feature holiday (0.003622)
```

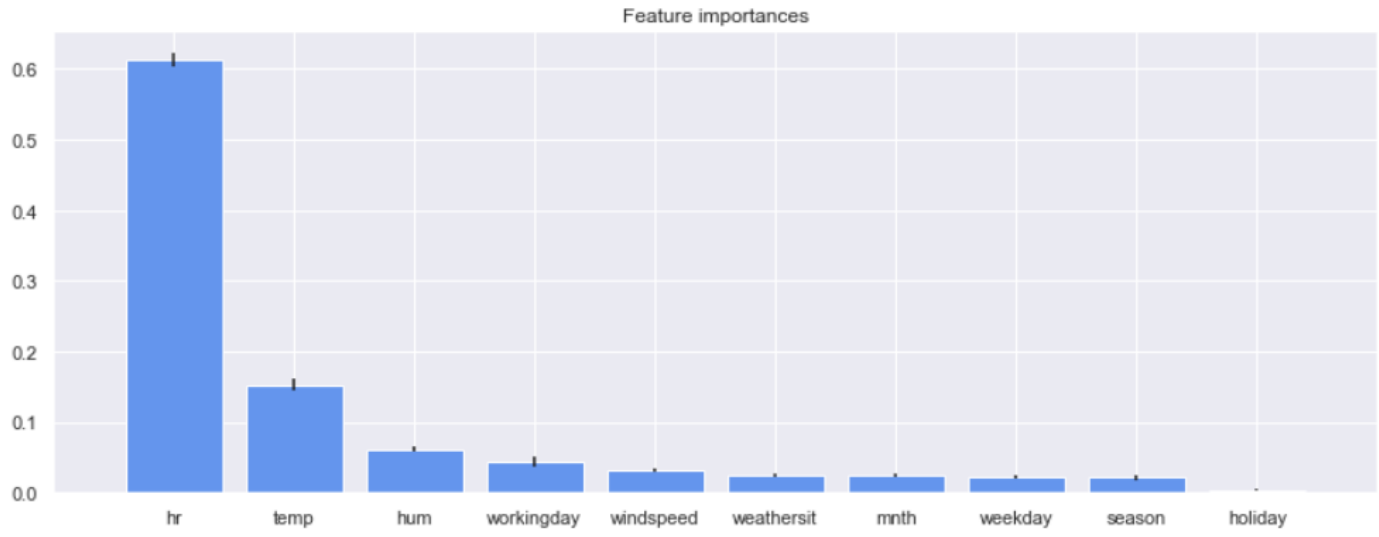


Figure 17: Feature Importance For Validation Dataset

7.Ensemble Learning Method Used- Bagging

Random Forest is one of the most popular and most powerful machine learning algorithms. It is a type of ensemble machine learning algorithm called Bootstrap Aggregation or bagging. The bootstrap is a powerful statistical method for estimating a quantity from a data sample. This is easiest to understand if the quantity is a descriptive statistic such as a mean or a standard deviation. Bootstrap Aggregation (or Bagging for short), is a simple and very powerful ensemble method. An ensemble method is a technique that combines the predictions from multiple machine learning algorithms together to make more accurate predictions than any individual model. Bootstrap Aggregation is a general procedure that can be used to reduce the variance for those algorithm that have high variance. An algorithm that has high variance are decision trees, like classification and regression trees (CART). Random Forests are an improvement over bagged decision trees.

A problem with decision trees like CART is that they are greedy. They choose which variable to split on using a greedy algorithm that minimizes error. As such, even with Bagging, the decision trees can have a lot of structural similarities and in turn have high correlation in their predictions. Combining predictions from multiple models in ensembles works better if the predictions from the sub-models are uncorrelated or at best weakly correlated. Random forest changes the algorithm for the way that the sub-trees are learned so that the resulting predictions from all of the subtrees have less correlation. It is a simple tweak. In CART, when selecting a split point, the learning algorithm is allowed to look through all variables and all variable values in order to select the most optimal split- point. The random forest algorithm changes this procedure so that the learning algorithm is limited to a random sample of features of which to search.

For each bootstrap sample taken from the training data, there will be samples left behind that were not included. These samples are called Out-Of-Bag samples or OOB. The performance of each model on its left out samples when averaged can provide an estimated accuracy of the bagged models. This estimated performance is often called the OOB estimate of performance. These performance measures are reliable test error estimate and correlate well with cross validation estimates.

As the Bagged decision trees are constructed, we can calculate how much the error function drops for a variable at each split point. In regression problems this may be the drop in sum squared error and in classification this might be the Gini score. These drops in error can be averaged across all decision trees and output to provide an estimate of the importance of each input variable. The greater the drop when the variable was chosen, the greater the importance. These outputs can help identify subsets of input variables that may be most or least relevant to the problem and suggest at possible feature selection experiments you could perform where some features are removed from the dataset.

Model	Mean Squared Error	R? score
SGDRegressor	28375.79	0.18
Lasso	23139.53	0.33
ElasticNet	27747.25	0.20
Ridge	23114.58	0.33
SVR	26233.71	0.24
SVR	19323.25	0.44
NuSVR	19334.33	0.44
RandomForestRegressor	5841.47	0.83

Figure 19-Model selection

We compare all the models and take the more contributing models and give them more preference while using bagging.

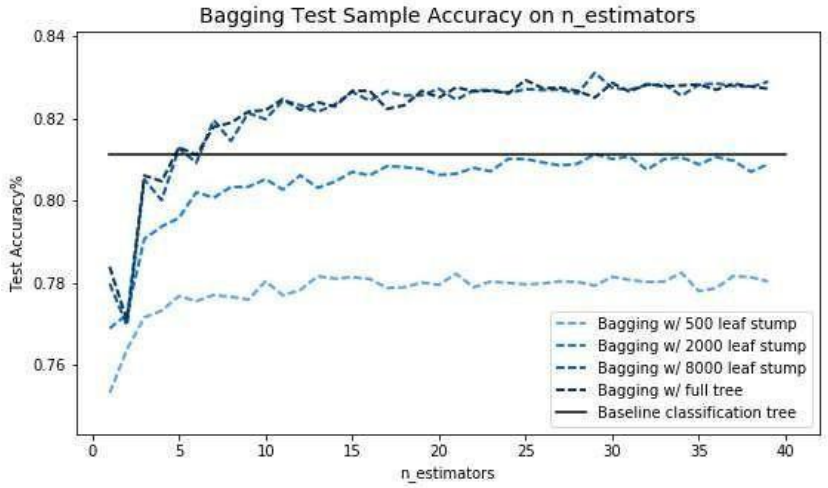


Figure 20: Test Accuracy% vs n_estimators

Here we consider n_estimators to be 10 because if we see figure 20 we can clearly see it gives more accuracy and also by increasing the n_estimators the accuracy approximately remained constant. Also we are suggesting bagging w/2000 leaf stump for more accuracy.

8. Conclusion And Future Work

Finally we can predict that the attributes hour and temperature variables are the factors that influence the most of the bike sharing count data set using which we know their prominence. We can also state that the use of Random Forest for this kind of data set is the best way to accurately predict the influence of which attributes are more and which are less. Similarly if you consider the method where we use the Training, Validation and Testing, we can conclude that the Training Dataset is always the best Dataset. Whereas we find the Mean of all the three splits for the MSE, MAE, RMSLE and R Square Score whereas in all the Means we consider Mean of the MAE as the factor to analyze the Random Forest Regression Model.

For the large data sets (>10 Mio. samples), the implementation of the Random Forest Regression using sklearn will slow down the process which is due to the more computation cost and also because the storage of the data in the main memory can not be completely stored. In certain senarios this sklearn leads to the worst implementation that crashes and thus not advised for large data sets.

There are many other alternatives for this problem, but the best one will be the python woodly implementation in which we use the top tree pre classification and that also distribites the samples to the bottom of the Random Forests that are implemented using C and this is also a highly optimised solution. We can use other machine learning frameworks such as Apache Spark ML for highly optimised distributed computation which will help in the utilization of computer clustering. This Spark ML can run Kubernetes, Apache Mesos, Hadloop etc, which will access the data from various popular Apache databases such as Apache Cassandra which is 100 times faster than any classic algorithms.

To improve the performance of the data model, we adjust the distribution of the target variable that is, if we see some of the predictive models, they assume the normal distribution of the target variable which can be improved by simply implementing a transformation in the data pre-processing which further improves the performance of the such methods.

9. Code

```
#Importing libraries nad packages
import pandas as pd
# from dataloader
# import dataloader
import seaborn as sns
import matplotlib.pyplot as plt
from prettytable import PrettyTable
import numpy as np
# Sklearn model delection
from sklearn.model_selection import RandomizedSearchCV
# Sklearn metrics
from sklearn.metrics import mean_squared_error,mean_absolute_error,mean_squared_log_error
# Sklearn models
```

```
from sklearn.linear_model import Lasso, ElasticNet, Ridge, SGDRegressor
from sklearn.svm import SVR, NuSVR
from sklearn.ensemble import BaggingRegressor, RandomForestRegressor
from sklearn.neighbors import KNeighborsClassifier
from sklearn.cluster import KMeans
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.linear_model import LinearRegression
import random
%matplotlib inline
# Make results reproducible
random.seed(100)
```

#Descriptive Analysis

#Provide data set splits for training, validation, and testing:

```
data = pd.read_csv(r'C:\Users\HP\Desktop\ML project\hour.csv')
category_features = ['season', 'holiday', 'mnth', 'hr', 'weekday', 'workingday', 'weathersit']
number_features = ['temp', 'atemp', 'hum', 'windspeed']

features= category_features + number_features
target = ['cnt']
```

data

Out[115]:

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0000	3	13	16
1	2	2011-01-01	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0000	8	32	40
2	3	2011-01-01	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0000	5	27	32
3	4	2011-01-01	1	0	1	3	0	6	0	1	0.24	0.2879	0.75	0.0000	3	10	13
4	5	2011-01-01	1	0	1	4	0	6	0	1	0.24	0.2879	0.75	0.0000	0	1	1

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
17374	17375	2012-12-31	1	1	12	19	0	1	1	2	0.26	0.2576	0.60	0.1642	11	108	119
17375	17376	2012-12-31	1	1	12	20	0	1	1	2	0.26	0.2576	0.60	0.1642	8	81	89
17376	17377	2012-12-31	1	1	12	21	0	1	1	1	0.26	0.2576	0.60	0.1642	7	83	90
17377	17378	2012-12-31	1	1	12	22	0	1	1	1	0.26	0.2727	0.56	0.1343	13	48	61
17378	17379	2012-12-31	1	1	12	23	0	1	1	1	0.26	0.2727	0.65	0.1343	12	37	49

17379 rows × 17 columns

```
from sklearn.model_selection import train_test_split
# train,test = train_test_split(data, test_size=0.33, random_state=42)
train, val, test = np.split(df.sample(frac=1, random_state=42),[int(.6*len(df)), int(.8*len(df))])
train
```

Out[118]:

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
12830	12831	2012-06-23	3	1	6	19	0	6	0	1	0.80	0.6970	0.27	0.1940	185	240	425
8688	8689	2012-01-02	1	1	1	20	1	1	0	1	0.24	0.2273	0.41	0.2239	5	83	88
7091	7092	2011-10-28	4	0	10	2	0	5	1	1	0.32	0.3030	0.66	0.2836	1	3	4
12230	12231	2012-05-29	2	1	5	19	0	2	1	1	0.78	0.7121	0.52	0.3582	69	457	526
431	432	2011	1	0	1	0	0	4	1	1	0.2	0.22	0.5	0.3881	5	8	13

	instant	dtedday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
		-01-20									6	73	6				
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6947	6948	2011-10-22	4	0	10	2	0	6	0	1	0.40	0.4091	0.62	0.2537	6	25	31
4657	4658	2011-07-17	3	0	7	23	0	0	0	1	0.70	0.6667	0.74	0.1343	39	54	93
6414	6415	2011-09-29	4	0	9	20	0	4	1	1	0.60	0.6212	0.53	0.0896	39	234	273
15609	15610	2012-10-17	4	1	10	14	0	3	1	2	0.56	0.5303	0.43	0.1642	82	188	270
3150	3151	2011-05-16	2	0	5	4	0	1	1	1	0.50	0.4848	1.00	0.1343	1	5	6

10427 rows × 17 columns

test

Out[119]:

	instant	dtedday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
3321	3322	2011-05-23	2	0	5	7	0	1	1	2	0.56	0.5303	0.94	0.1045	13	223	236
14062	14063	2012-08-14	3	1	8	3	0	2	1	2	0.68	0.6364	0.83	0.1940	0	3	3
6246	6247	2011-09-22	3	0	9	20	0	4	1	2	0.62	0.5455	0.94	0.1343	35	250	285
11564	11565	2012-05-02	2	1	5	1	0	3	1	1	0.56	0.5303	0.83	0.0000	8	7	15
604	6049	2011	3	0	9	1	0	3	1	1	0.7	0.71	0.5	0.2239	19	105	12

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
8		-09-14				4					8	21	2				4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
11284	11285	2012-04-20	2	1	4	9	0	5	1	1	0.46	0.4545	0.88	0.0896	30	329	359
11964	11965	2012-05-18	2	1	5	17	0	5	1	1	0.66	0.6212	0.34	0.1343	124	688	812
5390	5391	2011-08-17	3	0	8	12	0	3	1	1	0.80	0.7273	0.43	0.2836	26	163	189
860	861	2011-02-08	1	0	2	7	0	2	1	1	0.24	0.1970	0.65	0.4179	3	97	100
15795	15796	2012-10-25	4	1	10	8	0	4	1	2	0.52	0.5000	0.83	0.1642	33	746	779

3476 rows × 17 columns

#Get column names of the pandas data frame:

```
print(list(data.columns))
```

```
['instant', 'dteday', 'season', 'yr', 'mnth', 'hr', 'holiday', 'weekday', 'workingday',
'weathersit', 'temp', 'atemp', 'hum', 'windspeed', 'casual', 'registered', 'cnt']
```

#Print the first two samples of the dataset to explore the data:

```
print(data.head(2))
```

```
   instant      dteday  season  yr  mnth  hr  holiday  weekday  workingday  \
0        1  2011-01-01       1   0     1   0        0        6           0
1        2  2011-01-01       1   0     1   1        0        6           0

   weathersit  temp   atemp   hum  windspeed  casual  registered  cnt
0           1  0.24  0.2879  0.81        0.0       3          13   16
1           1  0.22  0.2727  0.80        0.0       8          32   40
```

```
fullData = data
#Get data statistics for each column:
```

```
                temp          atemp            hum      windspeed
count  17379.000000  17379.000000  17379.000000  17379.000000
mean       0.496987      0.475775      0.627229      0.190098
std        0.192556      0.171850      0.192930      0.122340
min        0.020000      0.000000      0.000000      0.000000
25%        0.340000      0.333300      0.480000      0.104500
50%        0.500000      0.484800      0.630000      0.194000
75%        0.660000      0.621200      0.780000      0.253700
max        1.000000      1.000000      1.000000      0.850700
```

```
for col in category_features:
  fullData[col] = fullData[col].astype('category')
print(fullData[category_features].describe())
```

```
        season  holiday   mnth     hr  weekday  workingday  weathersit
count    17379    17379  17379  17379    17379       17379       17379
unique       4        2     12     24        7           2           4
top          3        0      5     17        6           1           1
freq      4496    16879   1488    730     2512       11865       11413
```

#Missing Value Analysis

#Check any NULL values in data:

```
print(fullData.isnull().any())
```

```
instant       False
dteday        False
season        False
yr            False
mnth          False
hr            False
holiday       False
weekday       False
workingday    False
weathersit    False
temp          False
atemp         False
hum           False
windspeed     False
```

```
casual         False
registered     False
cnt            False
dtype: bool
```

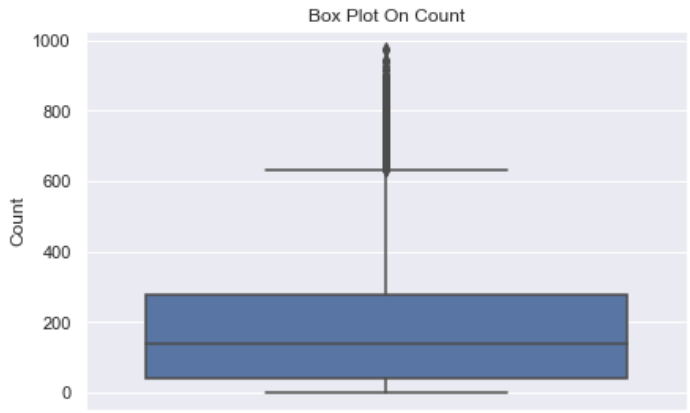
#Outlier Analysis

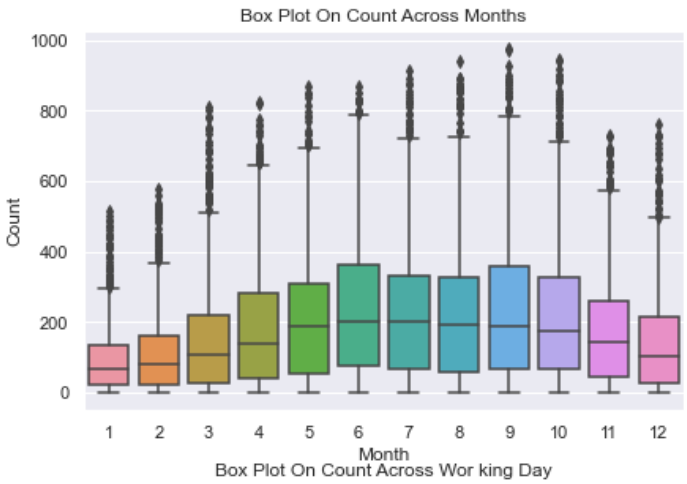
#Box plots

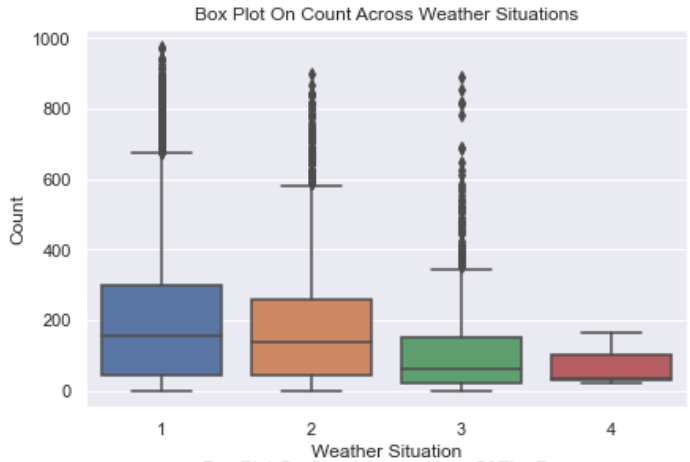
```
sns.set(font_scale=1.0)
fig,axes=plt.subplots(nrows=3,ncols=2)
fig.set_size_inches(15, 15)
sns.boxplot(data=train,y="cnt",orient="v",ax=axes[0][0])
sns.boxplot(data=train,y="cnt",x="mnth",orient="v",ax=axes[0][1])
sns.boxplot(data=train,y="cnt",x="weathersit",orient="v",ax=axes[1][0])
sns.boxplot(data=train,y="cnt",x="workingday",orient="v",ax=axes[1][1])
sns.boxplot(data=train,y="cnt",x="hr",orient="v",ax=axes[2][0])
sns.boxplot(data=train,y="cnt",x="temp",orient="v",ax=axes[2][1])

axes[0][0].set(ylabel='Count',title="Box Plot On Count")
axes[0][1].set(xlabel='Month', ylabel='Count',title="Box Plot On Count Across Months")
axes[1][0].set(xlabel='Weather Situation', ylabel='Count',title="Box Plot On Count Across Weather Situations")
axes[1][1].set(xlabel='Working Day', ylabel='Count',title="Box Plot On Count Across Wor king Day")
axes[2][0].set(xlabel='Hour Of The Day', ylabel='Count',title="Box Plot On Count Across Hour Of The Day")
axes[2][1].set(xlabel='Temperature', ylabel='Count',title="Box Plot On Count Across Temperature")
```

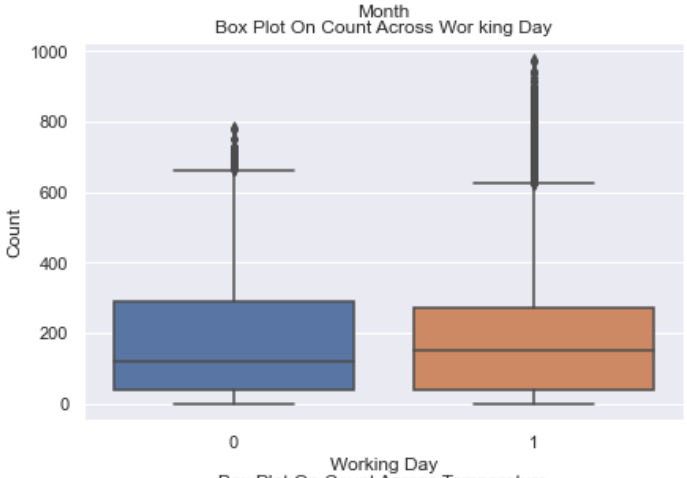
Out[126]:

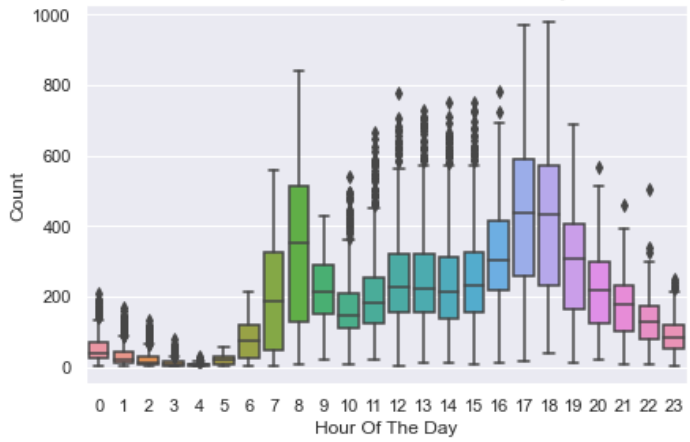
```
[Text(0.5, 0, 'Temperature'),
 Text(0, 0.5, 'Count'),
 Text(0.5, 1.0, 'Box Plot On Count Across Temperature')]
```

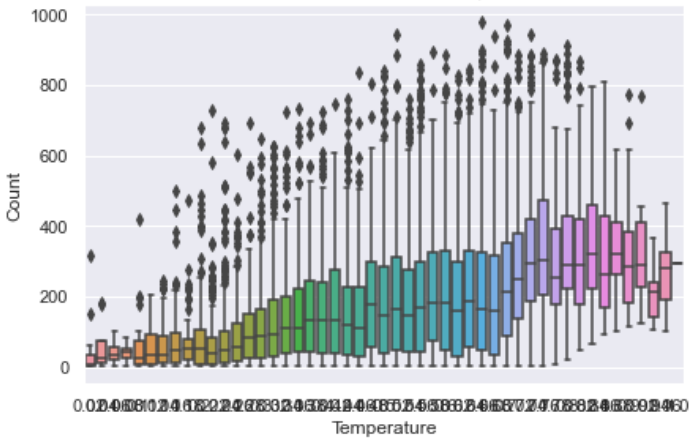










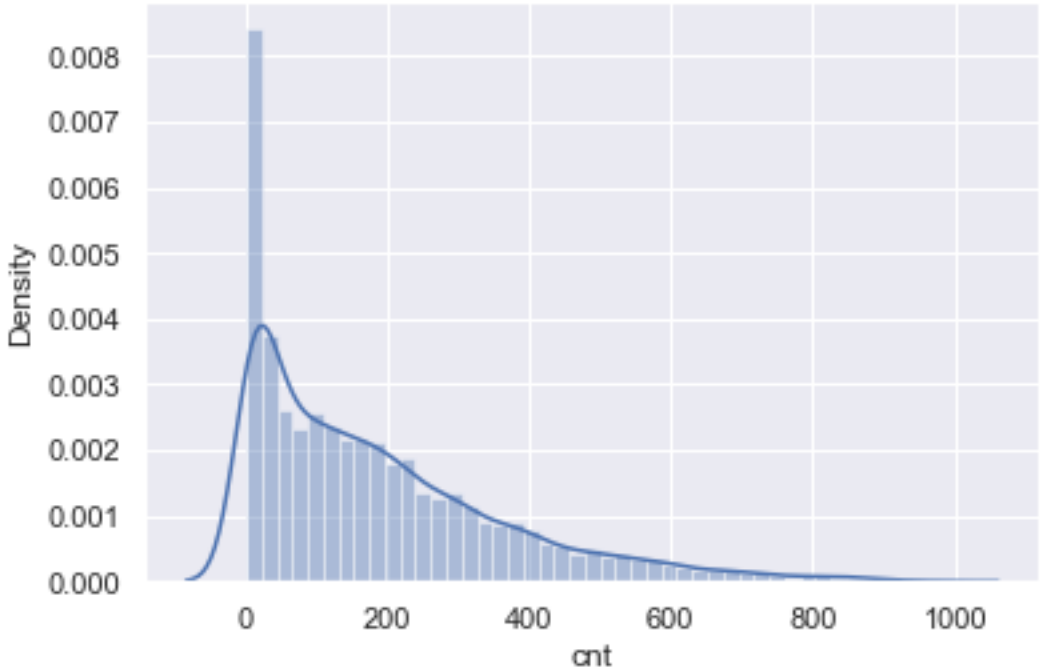


#Remove outliers from data

```
sns.distplot(train[target[-1]])
```

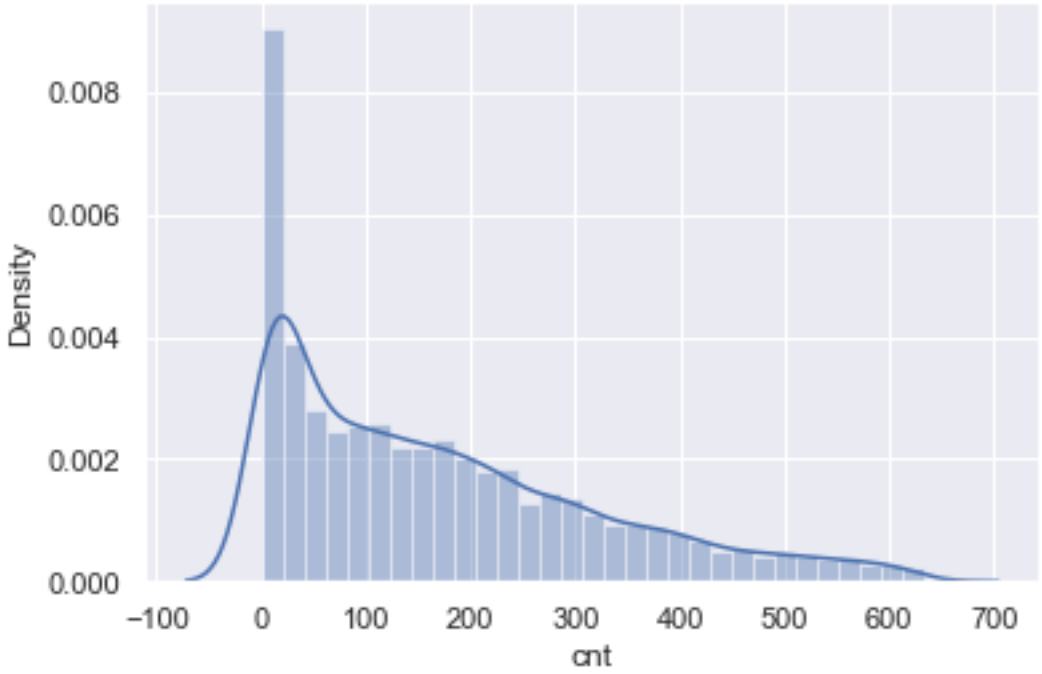
Out[127]:

```
<AxesSubplot:xlabel='cnt', ylabel='Density'>
```




```
print("Samples in train set with outliars:{}".format(len(train)))
q1 = train.cnt.quantile(0.25)
q3 = train.cnt.quantile(0.75)
iqr = q3 - q1
lower_bound = q1 -(1.5 * iqr)
upper_bound = q3 +(1.5 * iqr)
train_preprocessed = train.loc[(train.cnt >= lower_bound) & (train.cnt <= upper_bound)]
print("Samples in train set without outliers:{}".format(len(train_preprocessed)))
sns.distplot(train_preprocessed.cnt);
```

```
Samples in train set with outliars:10427
Samples in train set without outliers:10125
```

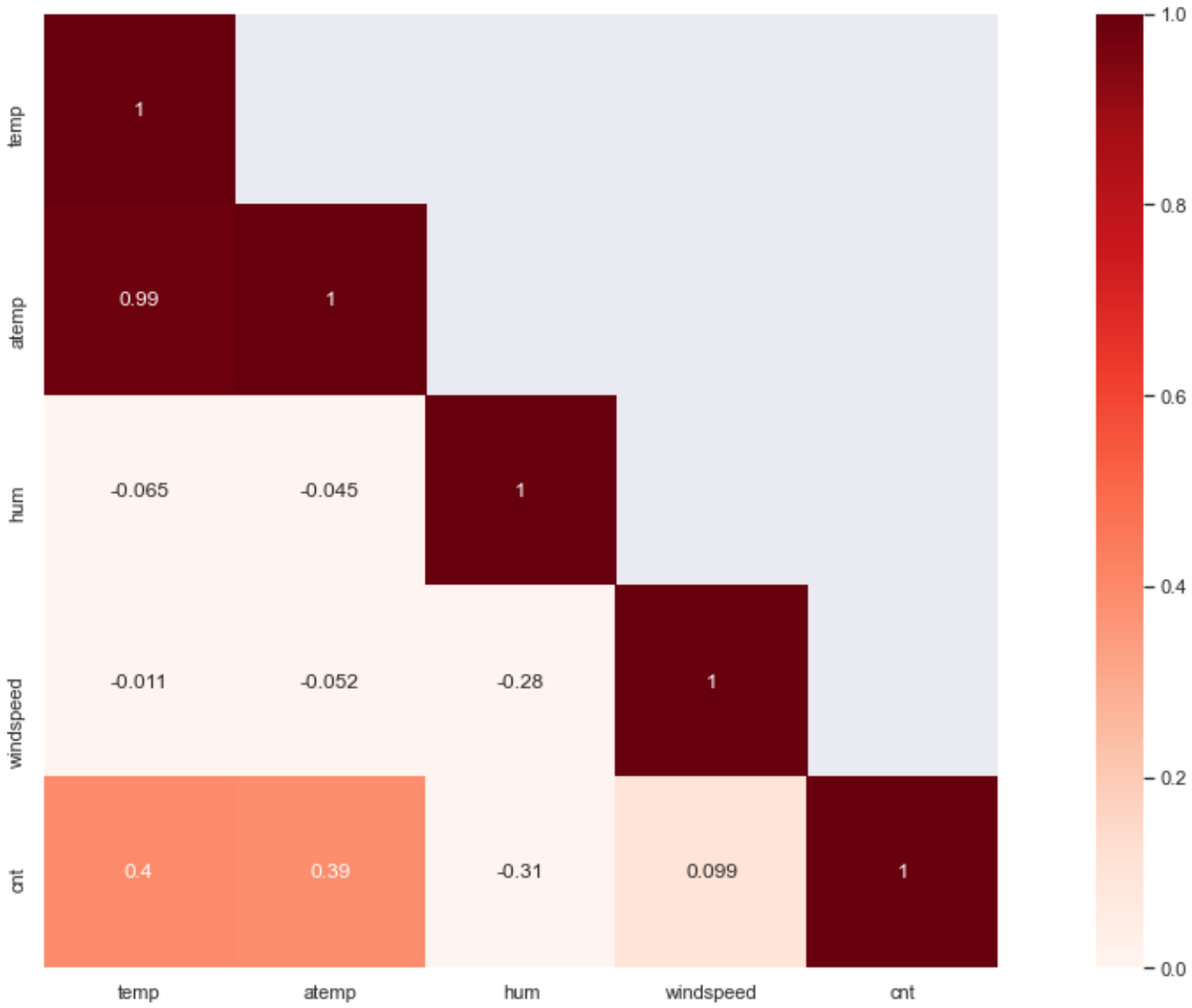


#Correlation Analysis

```
matrix=train[number_features+target].corr()
heat = np.array(matrix)
heat[np.tril_indices_from(heat)] = False
fig,ax= plt.subplots()
fig.set_size_inches(20,10)
sns.set(font_scale=1.0)
sns.heatmap(matrix, mask=heat,vmax=1.0, vmin=0.0, square=True,annot=True, cmap="Reds")
```

Out[129]:

```
<AxesSubplot:>
```




```
features.remove('atemp')
x_train = train_preprocessed[features].values
y_train = train_preprocessed[target].values.ravel()
# Sort validation set for plots

val = val.sort_values(by=target)
x_val = val[features].values
y_val = val[target].values.ravel()
x_test = test[features].values

table = PrettyTable()
```

```
table.field_names = ["Model", "Mean Squared Error", "R? score"]

models = [
   SGDRegressor(max_iter=1000, tol=1e-3),
   Lasso(alpha=0.1),
   ElasticNet(random_state=None),
   Ridge(alpha=.5),
   SVR(gamma='auto', kernel='linear'),
   SVR(gamma='auto', kernel='rbf'),
   NuSVR(gamma='auto'),
   RandomForestRegressor( random_state=None, n_estimators=300)
      ]

for model in models:
   model.fit(x_train, y_train)
   y_res = model. predict(x_val)

   mse = mean_squared_error(y_val, y_res)
   score = model.score(x_val, y_val)

   table.add_row([type(model).__name__, format(mse, '.2f'), format(score, '.2f')])

print(table)
```

```
+----------------------+--------------------+----------+
|         Model        | Mean Squared Error | R? score |
+----------------------+--------------------+----------+
|      SGDRegressor    |      28375.79      |   0.18   |
|         Lasso        |      23139.53      |   0.33   |
|       ElasticNet     |      27747.25      |   0.20   |
|         Ridge        |      23114.58      |   0.33   |
|          SVR         |      26233.71      |   0.24   |
|          SVR         |      19323.25      |   0.44   |
|         NuSVR        |      19334.33      |   0.44   |
| RandomForestRegressor |      5841.47       |   0.83   |
+----------------------+--------------------+----------+
```

```
#Random Forest Model
# Table setup
table= PrettyTable()
table.field_names = ["Model", "Dataset", "MSE", "MAE", 'RMSLE', "R² score"]

# Model training
```

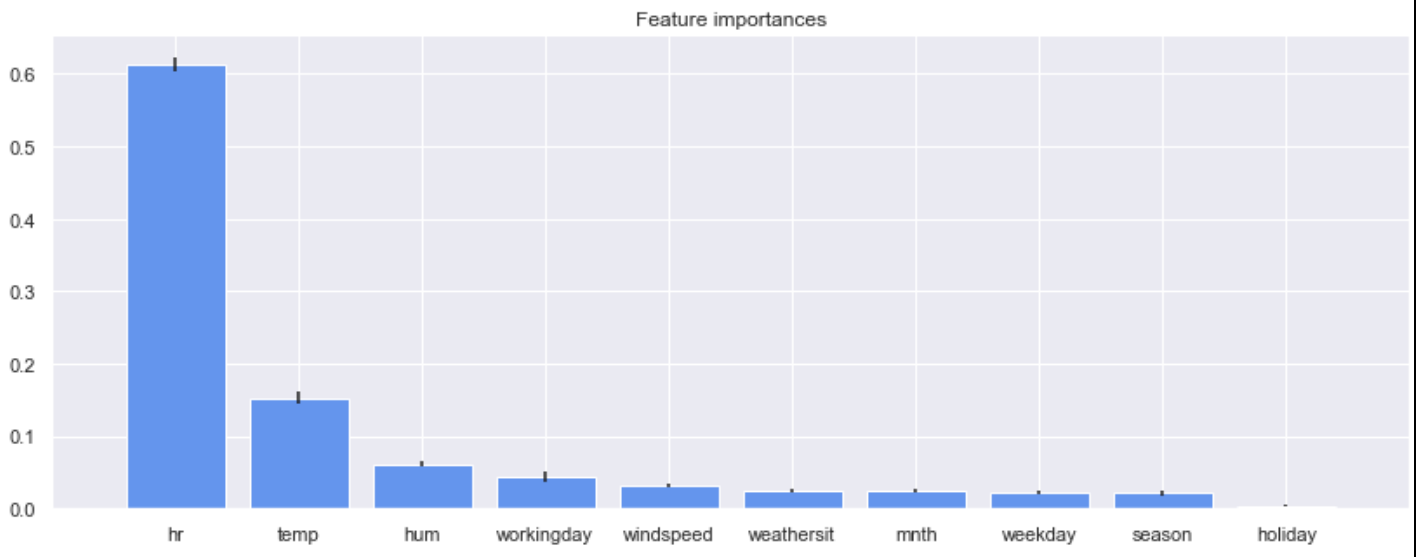
```
model = RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None, max_features='auto', max_leaf_nodes
=None,
min_impurity_decrease=0.0,  min_impurity_split=None, min_samples_leaf=1, min_samples_split=4,
min_weight_fraction_leaf=0.0,   n_estimators=200,   n_jobs=None, oob_score=False, random_state=None, verbose=0, warm_s
tart=False)
model.fit(x_train, y_train)

def evaluate(x, y, dataset):
   pred = model.predict(x)
   mse=mean_squared_error(y, pred)
   mae=mean_absolute_error(y,pred)
   score = model.score(x, y)
   rmsle = np.sqrt(mean_squared_log_error(y, pred))
   table.add_row([type(model) , dataset, format(mse, '.2f'), format(mae, '.2f'), format(rmsle, '.2f'), format(score, '.2f')])

evaluate(x_train, y_train, 'training')
evaluate(x_val, y_val, 'validation')
print(table)
```

```
+-------------------------------------------------------------+------------+---------+-----
--+-------+----------+
|                            Model                            |  Dataset   |   MSE   |  MAE
  | RMSLE | R² score |
+-------------------------------------------------------------+------------+---------+-----
--+-------+----------+
| <class 'sklearn.ensemble._forest.RandomForestRegressor'> |  training  |  688.08 | 17.2
7 |  0.22 |   0.97   |
| <class 'sklearn.ensemble._forest.RandomForestRegressor'> | validation | 5880.36 | 47.3
5 |  0.42 |   0.83   |
+-------------------------------------------------------------+------------+---------+-----
--+-------+----------+
```

```
#Feature importance
importances = model.feature_importances_
std = np.std([tree.feature_importances_ for tree in model.estimators_], axis=0)
indices = np.argsort(importances)[::-1]

# Print the feature ranking
print("Feature ranking:")

for f in range(x_val.shape[1]):
   print("%d. feature %s (%f)" % (f + 1, features[indices[f]], importances[indices[f]]))
```

```
Feature ranking:
1. feature hr (0.612987)
2. feature temp (0.153037)
3. feature hum (0.061663)
4. feature workingday (0.044289)
5. feature windspeed (0.031500)
6. feature weathersit (0.024915)
7. feature mnth (0.023815)
8. feature weekday (0.022891)
9. feature season (0.021279)
10. feature holiday (0.003622)
```

```
# Plot the feature importances of the forest
plt.figure(figsize=(14,5))
plt.title("Feature importances")
plt.bar(range(x_val.shape[1]),
     importances[indices],
     color="cornflowerblue",
     yerr=std[indices],
     align="center")
plt.xticks(range(x_val.shape[1]), [features[i] for i in indices])
plt.xlim([-1, x_val.shape[1]])
plt.show()
```



10.References

[1] Pham Thi, Thanh Thoa Timoney, Joe Ravichandran, Shyram Mooney, Peter Winstanley, A.. (2017). Bike Renting Data Analysis: The Case of Dublin
City.

[2] Normark, Daniel Cochoy, Franck Hagberg, Johan Ducourant, Hélène. (2018). Mundane intermodality: a comparative analysis of bike-renting practices. Mobilities. 1-17. 10.1080/17450101.20178.1504651.

[3] Prediction model of demand for public bicycle rental based on land use Shuichao Zhang, Zhuping Zhou, Haiming Hao, Jibiao Zhou

[4] Understanding the Usage Patterns of Bicycle-Sharing Systems to Predict Users' Demand: A Case Study in Wenzhou, China

[5] Implementation of Web – Based Bike Renting Application "Bike – Sharing",International Journal of Computer Science and Mobile Computing Ratieh Indah Permitasari1 , Riad Sahara2 1,2Faculty of Computer Science, Mercu Buana University, Indonesia.

[6] Peter Mooney, Padraig Corcoran, Adam C. Winstanley. Preliminary Results of a Spatial Analysis of Dublin City's Bike Rental Scheme. GIS Research UK, 2010.

[7] Ji Won Yoon, Fabio Pinelli, Francesco CalabreseCityride. A Predictive Bike Sharing Journey Advisor

[8] Yan Pan, Ray Chen Zheng, Jiaxi Zhang, Xin Yao. Predicting bike sharing demand using recurrent neural networks. 2018 International Conference on Identification and Knowledge in the Internet of Things, IIKI 2018.

[9] Hong Yang, Kun Xie, Zhenyu Wang, Kaan Ozbay. Use of Deep Learning to Predict Daily Usage of Bike Sharing System.

[10] Zhifeng Wang. Regression Model for Bike Sharing Service by Using Machine Learning. Asian Journal of Social Studies, 2019.