**Project Report on**

**Bike Renting**

**Submitted by:**

**OmPrakash Sah Kalwar**

# edWisor E-learning

**Gurgaon, Haryana**

**DECLARATION**

I, OmPrakash Sah Kalwar, student of B.E Batch 2020 passout of CMR Institute Of Technology, Visvesvaraya Technological University, Aecs Layout, Kundhanahalli gate, Banglore-560037 declares that Project Report on Bike Rentingsubmitted and is the original work conducted by me.

The information and data given in the report is authentic to the best of my knowledge.

OmPrakash Sah Kalwar

Place: Banglore-560037

**Content**

**Topic** **Page**

1.**Introduction**

1.1 Problem Statement 4

1.2 Data 4

1.3 Description 5

**Research Methodology** 6-21

**Conclusion**  22

**Summary** 22-23

**INTODUCTION**

* 1. **Problem Statememt**

The objective of this Case is to Predication of bike rental count on daily based on the

environmental and seasonal settings.

**1.2 Data**

instant: Record index

dteday: Date

season: Season (1:springer, 2:summer, 3:fall, 4:winter)

yr: Year (0: 2011, 1:2012)

mnth: Month (1 to 12)

hr: Hour (0 to 23)

holiday: weather day is holiday or not (extracted fromHoliday Schedule)

weekday: Day of the week

workingday: If day is neither weekend nor holiday is 1, otherwise is 0.

weathersit: (extracted fromFreemeteo)

1: Clear, Few clouds, Partly cloudy, Partly cloudy

2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist

3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered

clouds

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

temp: 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)

atemp: Normalized feeling temperature in Celsius. The values are derived via

(t-t\_min)/(t\_maxt\_

min), t\_min=-16, t\_max=+50 (only in hourly scale)

hum: Normalized humidity. The values are divided to 100 (max)

windspeed: Normalized wind speed. The values are divided to 67 (max)

casual: count of casual users

registered: count of registered users

cnt: count of total rental bikes including both casual and registered

* 1. **Description**

The dataset “Bike-Sharing-Dataset” was obtained by the UCI Machine Learning Repository. This is a collection of databeses, domain theories and data generators which are used by the machine learning community for empirical analyses. The archive was created in 1987 by David Aha and fellow graduate students at UC Irvine. Since then it has been widely used by student, educators and researchers. The current website was designed in 2007. The UCI Machine Learning Repository is based on donations of researchers, mostly outside of UCI. We found the dataset “Bike Sharing Dataset” under the index “regression” and chose the sub-dataset “day”.

This dataset contains the hourly and daily count of rental bikes between years 2011 and 2012 in Capital bikeshare system with the corresponding weather and seasonal information. Capital bikeshare has over 350 stations in Washington, D. C. , Arlington, Alexandria, VA und Montgomery County and MD. Bike sharing systems are a new way of traditional bike rentals. The wohle process from memberhsip to rental and retrun back has become automatic. The data was generated by 500 bike-sharing programs and was collected by the Laboratory of Artificial Intelligence and Decision Support (LIAAD), University of Porto. The Laboratory of Artificial Intelligence and Decision Support (LIAAD), University of Porto, aggregated the data on two hourly and daily basis and then extracted and added the corresponding weather and seasonal information.

**Research Methodology**

To achieve the proposed objectives of the study, Primary data regarding Predication of bike rental count on daily based on the environmental and seasonal settings.

We have used different libraries to approach the problem statement.

Importing necessary libraries

import pandas as pd # data processing

import os #To Interact with local system directories

import numpy as np # linear algebra

import matplotlib.pyplot as plt # some plotting

import seaborn as sns # Seaborn is a library for making statistical graphics in Python

from scipy import stats #for stattistical model

from scipy.stats import chi2\_contingency

from sklearn.ensemble import RandomForestClassifier # checking if this is available

# from sklearn import cross\_validation

%matplotlib inline

Understanding of data and Describing the data.

| **instant** | **season** | **yr** | **mnth** | **holiday** | **weekday** | **workingday** | **weathersit** | **temp** | **atemp** | **hum** | **windspeed** | **casual** | **registered** | **cnt** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| count | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 | 731.000000 |
| mean | 366.000000 | 2.496580 | 0.500684 | 6.519836 | 0.028728 | 2.997264 | 0.683995 | 1.395349 | 0.495385 | 0.474354 | 0.627894 | 0.190486 | 848.176471 | 3656.172367 | 4504.348837 |
| std | 211.165812 | 1.110807 | 0.500342 | 3.451913 | 0.167155 | 2.004787 | 0.465233 | 0.544894 | 0.183051 | 0.162961 | 0.142429 | 0.077498 | 686.622488 | 1560.256377 | 1937.211452 |
| min | 1.000000 | 1.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000 | 0.059130 | 0.079070 | 0.000000 | 0.022392 | 2.000000 | 20.000000 | 22.000000 |
| 25% | 183.500000 | 2.000000 | 0.000000 | 4.000000 | 0.000000 | 1.000000 | 0.000000 | 1.000000 | 0.337083 | 0.337842 | 0.520000 | 0.134950 | 315.500000 | 2497.000000 | 3152.000000 |
| 50% | 366.000000 | 3.000000 | 1.000000 | 7.000000 | 0.000000 | 3.000000 | 1.000000 | 1.000000 | 0.498333 | 0.486733 | 0.626667 | 0.180975 | 713.000000 | 3662.000000 | 4548.000000 |
| 75% | 548.500000 | 3.000000 | 1.000000 | 10.000000 | 0.000000 | 5.000000 | 1.000000 | 2.000000 | 0.655417 | 0.608602 | 0.730209 | 0.233214 | 1096.000000 | 4776.500000 | 5956.000000 |
| max | 731.000000 | 4.000000 | 1.000000 | 12.000000 | 1.000000 | 6.000000 | 1.000000 | 3.000000 | 0.861667 | 0.840896 | 0.972500 | 0.507463 | 3410.000000 | 6946.000000 | 8714.000000 |

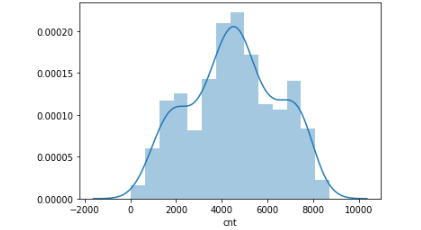
Univariate Analysis

Target variable analysis

Descriptive of statistics summary

Check whether target variable is normal or not

sns.distplot(df\_day['cnt']);



Distribution independent numeric variables

#Check whether variable 'temp'is normal or not

sns.distplot(df\_day['temp']);

#Check whether variable 'atemp'is normal or not

sns.distplot(df\_day['atemp']);

#Check whether variable 'hum'is normal or not

sns.distplot(df\_day['hum']);

#Check whether variable 'windspeed'is normal or not

sns.distplot(df\_day['windspeed']);

#Check whether variable 'casual'is normal or not

sns.distplot(df\_day['casual']);

#Check whether variable 'registered'is normal or not

sns.distplot(df\_day['registered']);

It is clearly showing that chances of outliers present in 'casual' variable

Bivariate Relationship

Relation between Numerical Variable 'temp' and target variable 'cnt'

df\_day['temp'].value\_counts()

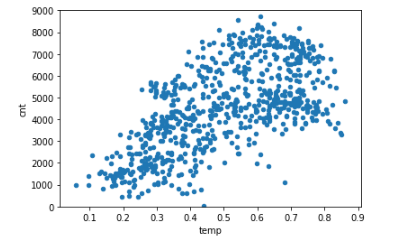
#Now draw scatter plot between 'temp' and 'cnt' variables

var = 'temp'

data = pd.concat([df\_day['cnt'], df\_day[var]], axis=1)

data.plot.scatter(x=var, y='cnt', ylim=(0,9000));

It is showing there is good relation between 'temp' and 'cnt'



Relation between Numerical Variable 'atemp' and target variable 'cnt'

df\_day['atemp'].value\_counts()

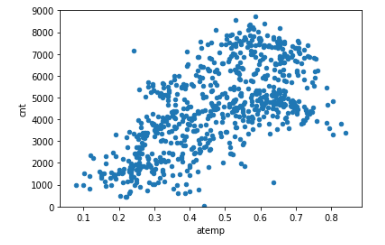
#Now draw scatter plot between 'temp' and 'cnt' variables

var = 'atemp'

data = pd.concat([df\_day['cnt'], df\_day[var]], axis=1)

data.plot.scatter(x=var, y='cnt', ylim=(0,9000));

It is showing there is good relation between 'atemp' and 'cnt'



Relation between Numerical Variable 'hum' and target variable 'cnt'

df\_day['hum'].value\_counts()

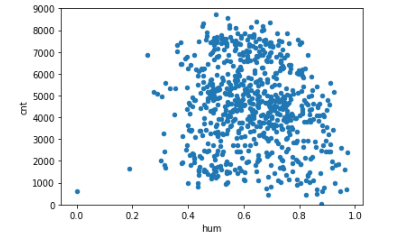
#Now draw scatter plot between 'hum' and 'cnt' variables

var = 'hum'

data = pd.concat([df\_day['cnt'], df\_day[var]], axis=1)

data.plot.scatter(x=var, y='cnt', ylim=(0,9000));

It is showing there is average relation between 'atemp' and 'cnt'



Relation between Numerical Variable 'windspeed' and target variable 'cnt'

df\_day['windspeed'].value\_counts()

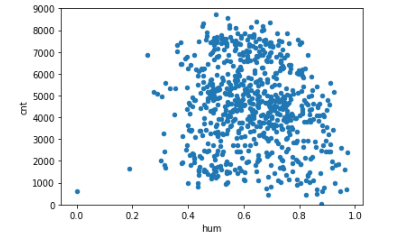
#Now draw scatter plot between 'windspeed' and 'cnt' variables

var = 'windspeed'

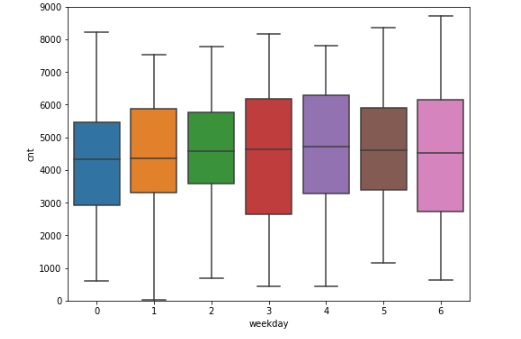
data = pd.concat([df\_day['cnt'], df\_day[var]], axis=1)

data.plot.scatter(x=var, y='cnt', ylim=(0,9000));

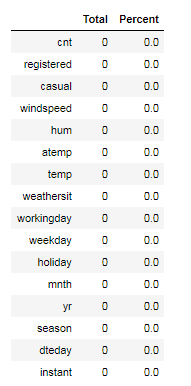
It is showing there is nagative relation between 'windspeed' and 'cnt'



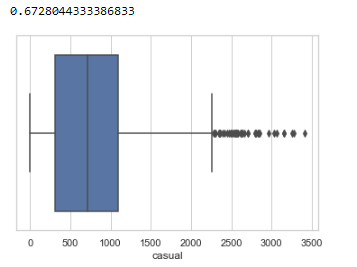
Below Boxplot is saying that for all the weekdays median in between 4000- 5000



Finding total missing values. Finding total missing values and percentage of missing data to compare the data.



Already all numeric variable are in normalize form so , we are not analysing Outliers here the six numerics variables are present out of six four variables are in normalize form temp,atem,hum,windspread are in normalize form no need to check outliers casual and registered have to check outliers. Seaborn is a library for making statistical graphics in Python changing background white Orientation of the plot horizontal It seems Outliers are present in 'Casual' variable but we are keeping as it is , will detect and conver outliers during tuning process. Correlation between 'casual' and 'cnt' before removal of outliers.



Determinig the 75 and 25 percntage values

cnames = ['casual']

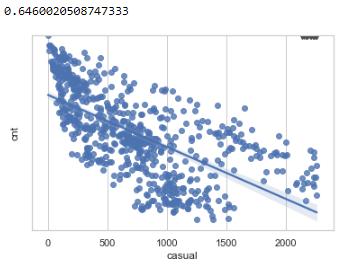
for i in cnames:

q75, q25 = np.percentile(df\_day.loc[:,i], [75 ,25])

iqr = q75 - q25

q75, q25

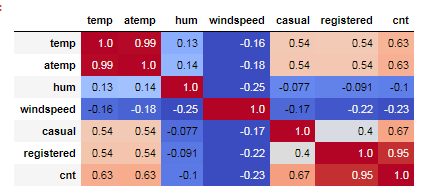
Calculating minimum and maximum values. Correlation between 'casual' and 'cnt' after removal of outliers.



Feature selection and Selection of numerical feature based on pearson correlation.

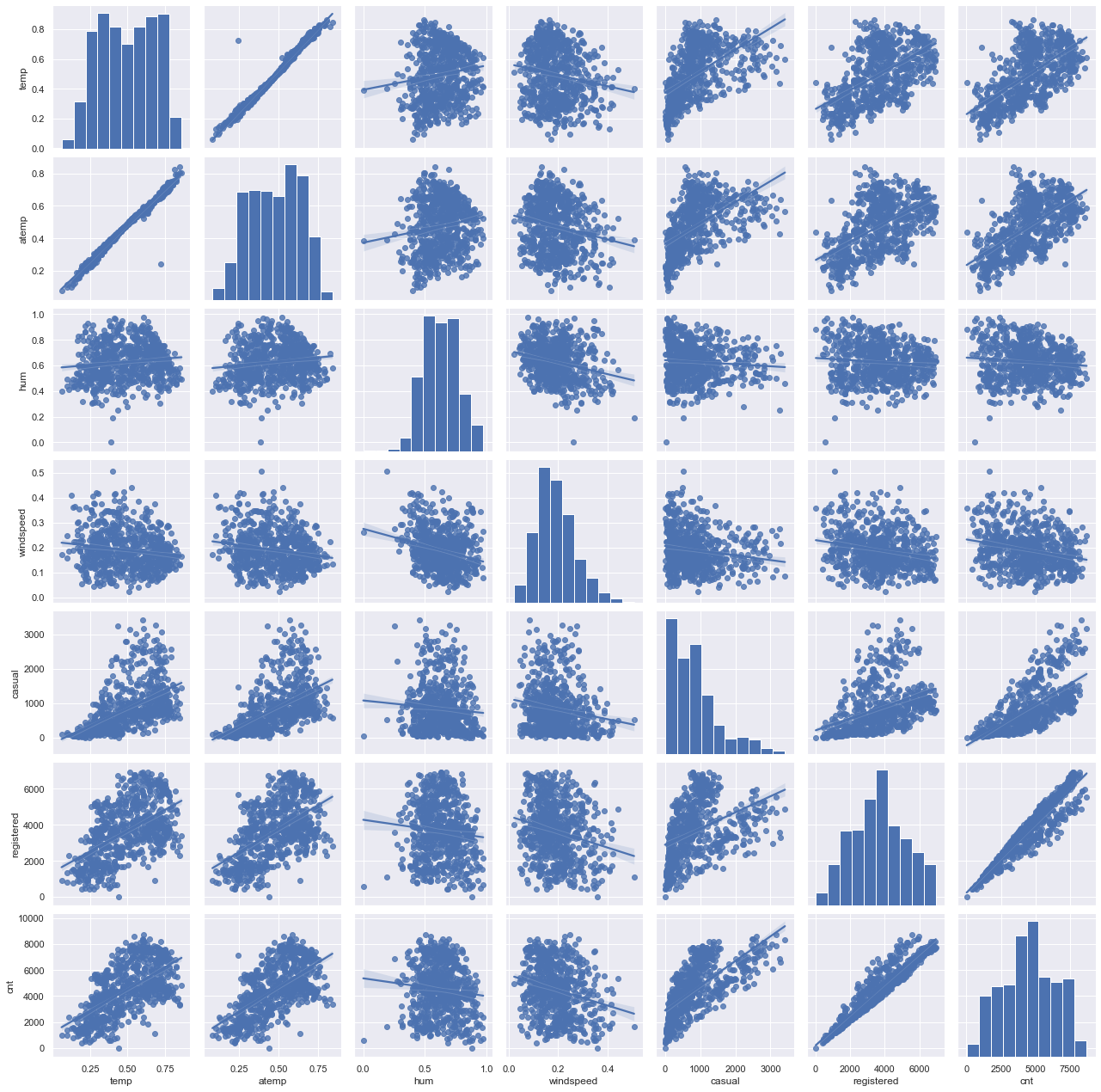
Draw correlation matrix between all numeric variables and analyse what are the variables are important.

day\_numeric.corr(method='pearson').style.format("{:.2}").background\_gradient(cmap=plt.get\_cmap('coolwarm'), axis=1)



Check relationship with scatter plots. As per scatter plots and above Correlation graph there is strong relation Independent variable 'temp' and 'atemp'.There is a poor relation between Independent variable 'hum' and dependent variable 'cnt' so dropping two variables for feature selection.

numeric\_features.shape



Feature Scaling and normality Check. Dividing Test and train data using skilearn train\_test\_split.

Decision Tree Regressor

#Importing Decision Tree Regressor from sklear.tree

from sklearn.tree import DecisionTreeRegressor

#Train/Test is a method to measure the accuracy of your model.

train\_features\_one = train[['season','yr','mnth','holiday','weekday','weathersit','temp','windspeed','casual','registered']].values

train\_target\_feature = train['cnt'].values

test\_feature = test[['season','yr','mnth','holiday','weekday','weathersit','temp','windspeed','casual','registered']].values

test\_target\_feature= test['cnt'].values

train\_features\_one

#target\_feature

# Implement decision tree algorithm

# Fit your first decision tree: my\_tree\_one

my\_tree\_one = DecisionTreeRegressor()

my\_tree\_one = my\_tree\_one.fit(train\_features\_one, train\_target\_feature)

print(my\_tree\_one)

#Decision tree for regression

#fit\_DT = DecisionTreeRegressor(max\_depth=2).fit(train.iloc[:,2:13], train.iloc[:,13])

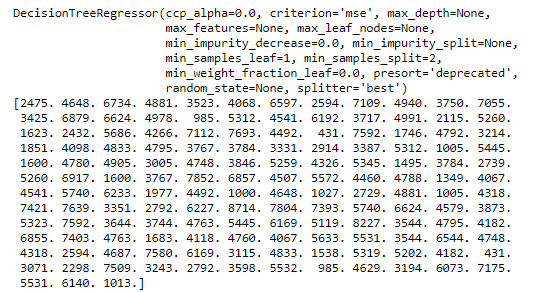
#Apply model on test data

predictions\_DT = my\_tree\_one.predict(test\_feature)

print(predictions\_DT)

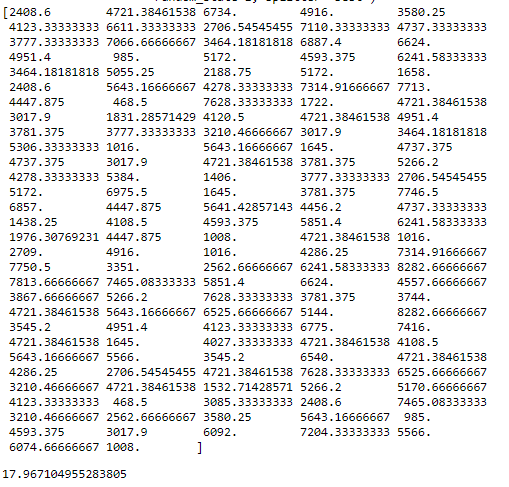
# prediction by using predict method

#predictions\_DT = my\_tree\_one.predict(test\_feature)



Calculate MAPE.

MAPE(test\_target\_feature,predictions\_DT\_two)



RSS - Residual Sum

#test\_target\_feature,predictions\_DT\_two

rss= ((test\_target\_feature-predictions\_DT\_two)\*\*2).sum()

print(rss)

MSE = np.mean((test\_target\_feature-predictions\_DT\_two)\*\*2)

print(MSE)

#RMSE

rmse=np.sqrt(MSE)

print(rmse)

def RMSE(y\_test,y\_predict):

mse = np.mean((y\_test-y\_predict)\*\*2)

print("Mean Square : ",mse)

rmse=np.sqrt(mse)

print("Root Mean Square : ",rmse)

return rmse

#MAPE

MAPE(test\_target\_feature,predictions\_DT\_two)

#RMSE

RMSE(test\_target\_feature,predictions\_DT\_two)

We get:

Mean Square : 66033.36276633634

Root Mean Square : 256.96957556554503

Random Forest

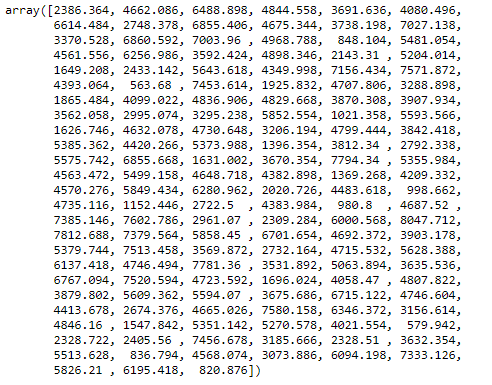
# Instantiate random forest and train on new features

from sklearn.ensemble import RandomForestRegressor

RF\_model\_one = RandomForestRegressor(n\_estimators= 500, random\_state=100).fit(train\_features\_one,train\_target\_feature)

# Predict the model using predict funtion

RF\_predict\_one= RF\_model\_one.predict(test\_feature)



Evaluate Random forest using MAPE

MAPE(test\_target\_feature,RF\_predict\_one)

#Here it is stating accuracy of the model increases

Evaluate Model using RMSE

RMSE(test\_target\_feature,RF\_predict\_one)

Tuning Random Forest Model

importances = list(RF\_model\_one.feature\_importances\_)

print(importances)

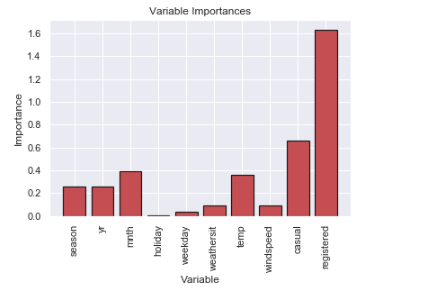
# List of tuples with variable and importance

#zip() can accept any type of iterable, such as files, lists, tuples, dictionaries, sets, and so on

feature\_importances = [(feature, round(importance, 2)) for feature, importance in zip(train\_features\_one, importances)]

# Sort the feature importances by most important first

feature\_importances = sorted(feature\_importances, key = lambda x: x[1], reverse = True)



Statsmodels. statsmodels is a Python module that provides classes and functions for the estimation of many different statistical models, as well as for conducting statistical tests, and statistical data exploration. An extensive list of result statistics are available for each estimator.

import statsmodels.api as sm

#develop Linear Regression model using sm.ols

#The Statsmodels package provides different classes for linear regression, including OLS

linear\_regression\_model = sm.OLS(train\_target\_feature, train\_features\_one).fit()

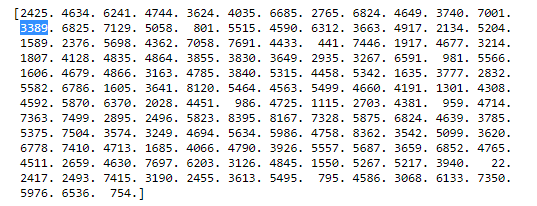
#Summary of model

linear\_regression\_model.summary()

#predict the model

predict\_LR = linear\_regression\_model.predict(test\_feature)

print(predict\_LR)



Evaluate model using MAPE

MAPE(test\_target\_feature,predict\_LR)

#Predict the model using RMSE

RMSE(test\_target\_feature,predict\_LR)

It is showing that Linear Regression model is best suitable for the dataset

**Conclusion**

We can say, that the amount of bike rentals depends mainly on the weather and on the real and feeled temperature. The analysis shows that there is a positive relationship between the amount of bike rentals and temperature. And as we can see in the plot 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.

As we plotted the mean humidity, the mean temperature, the mean windspeed an the mean total rentals per months. As we can see the total amount of bike rentals increases with the temperature per month. Whereas it seems that the rentals are independent of the windspeed and the humidity, because they are almost constant over the months. This also confirms on the one hand the high correlation between rentals and temperature and on the other hand that nice weather could be a good predictor. As per the scatter plot we can see their is a liner relation between number of bike rented and temperature the warmmer the temp the more bike get rented the graph of 'weathersit' shows that people tends to rent more bike in clear weather

Another important factor to understand the bike rental trends is temperature. There was not a significant difference between real temperature and feeling of the temperature. Total Temperature was significantly correlated with total bike rentals. Irrespective of the type of bike rental the temperature (real) was equally correlated with both casual and registered rentals. However, one important thing to mention again is that the registered users were higher overall compared to casual users, hence the results could be biased or misleading, but overall it seems like temperature plays an important role for total count. Furthermore, weather situation was found to be significant predictor of bike rentals. However,the impact of holidays is not significant. One reason for holidays not being significant could be that there were probably a very few holidays during the years for it to have a significant impact.

**Summary**

Amount of bike rentals depends mainly on the weather and on the real and feeled temperature. The analysis shows that there is a positive relationship between the amount of bike rentals and temperature. And as we can see in the plot 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.

Because of the mild temperatures in spring and winter and the warm weather in summer and fall, temperature should be highly correlated with the total amount of bike rentals.

In this analysis we correlated the raw temperature (converted nomalized temperature), the converted feeled temperature (raw.atemp) and the mean of both. All three kinds of temperatures are positive correlated with the total amount of bike rentals. We also see that the correlation value is not different across the three types of temperatures . Anyway the analysis clearly shows, that there is a relationship between those two variables.