EDWISOR.COM

REPORT

Project Name: Bike Renting

Submitted By:

Rishabh Singh

ABSTRACT

The objective of this Case is to Predication of bike rental count on daily basis, based on the environmental and seasonal settings. The dataset contains 16 variables each representing different attributes and properties. The total number of observations are 700+ entries. This project is divided into four major stages.

The first part contains the Exploratory Data Analysis where we check the head of the dataset, how each variable is type defined and use describe method to check the max, min and other values.

The second part contains the Data preprocessing and data cleaning techniques to make it ready for various Machine Learning Algorithms.

The third part contains the training our model on various Machine Learning Algorithms and testing our model on them.

The last and the final part of the project contains the prediction of our sample test data by most appropriate algorithm.

List of Contents

INTRODUCTION

The Bike Rental case is to Predication of bike rental count on daily basis based on the environmental and seasonal settings.

The details of data attributes in the dataset are as follows:

- instant: Record index
- dteday: Date
- season: Season (1:spring, 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 from Holiday Schedule)
- weekday: Day of the week
- workingday: If day is neither weekend nor holiday is 1, otherwise is 0.
- weathersit: (extracted from Freemeteo)
  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

Out of the total of 16 variables of the dataset:

Categorical variables are: 7

Continuous variables are: 7

Index/String variable: 2

Exploratory Data Analysis

1) Set working directory: To import and export all data to a location
2) Load Data: training data as bikerental_train
3) Checking the shape of Data: 731 rows, 16 columns
4) Checking the type of each variable in the dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 731 entries, 0 to 730
Data columns (total 16 columns):
instant       731 non-null int64
dteday        731 non-null object
season        731 non-null int64
yr            731 non-null int64
mnth          731 non-null int64
holiday       731 non-null int64
weekday       731 non-null int64
workingday    731 non-null int64
weathersit    731 non-null int64
temp          731 non-null float64
atemp         731 non-null float64
hum           731 non-null float64
windspeed     731 non-null float64
casual        731 non-null int64
registered    731 non-null int64
cnt           731 non-null int64
dtypes: float64(4), int64(11), object(1)
memory usage: 91.5+ KB
```

5) Check the head of the dataset to know the first five entries:

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
1	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
2	3	2011-01-03	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
3	4	2011-01-04	1	0	1	0	2	1	1	0.200000	0.212122	0.590435	0.160296	108	1454	1562
4	5	2011-01-05	1	0	1	0	3	1	1	0.226957	0.229270	0.436957	0.186900	82	1518	1600

6) Apply describe() method to calculate the mean, variance, and standard deviation etc of the dataset:

	instant	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	
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	73
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	84
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	68
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
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	31
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	71
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	109
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	341

DATA Pre-processing

1) **Missing Value Analysis**: Missing value pertains to condition when one or more values are absent from one or more independent variables of the dataset. The reason for missing values could be many. Some commonly included reasons are Human error, Refuse to answer during survey, optional box questionnaire etc. Missing values in a dataset may be imputed or that particular variable may be dropped depending upon the percentage of missing values present in the column.

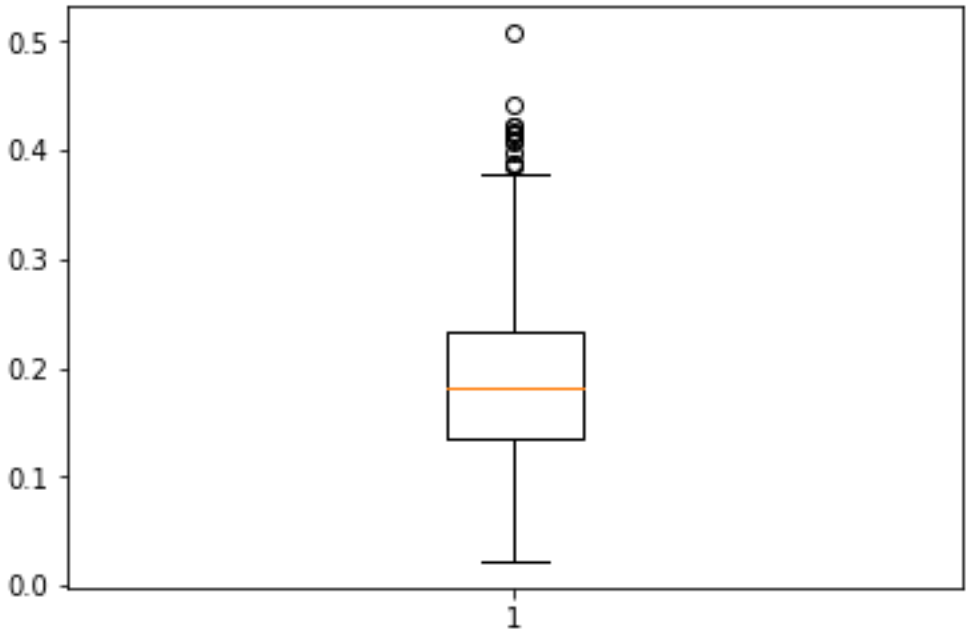
   There are various methods to impute missing values, some of which are:

   a) Mean
   b) Mode
   c) Median
   d) KNN imputation
   e) Prediction method

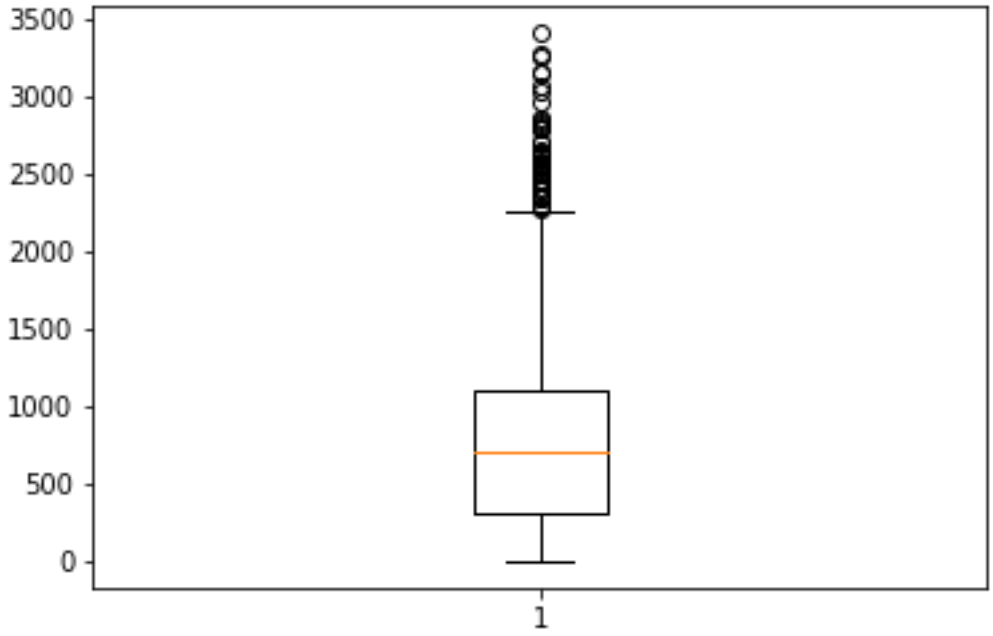
   Checking the missing value in the Dataset: Our Dataset does not contain any missing value.

2) **Outlier Analysis**: Observations inconsistent with the rest of the dataset is called an outlier. Causes of outliers are poor or contaminated data, low quality measurements, malfunctioning equipment, correct but exceptional data.

   - Plot boxplot to visualize Outliers.
   - Separate Numeric and Categorical variables.



Outliers in "windspeed"



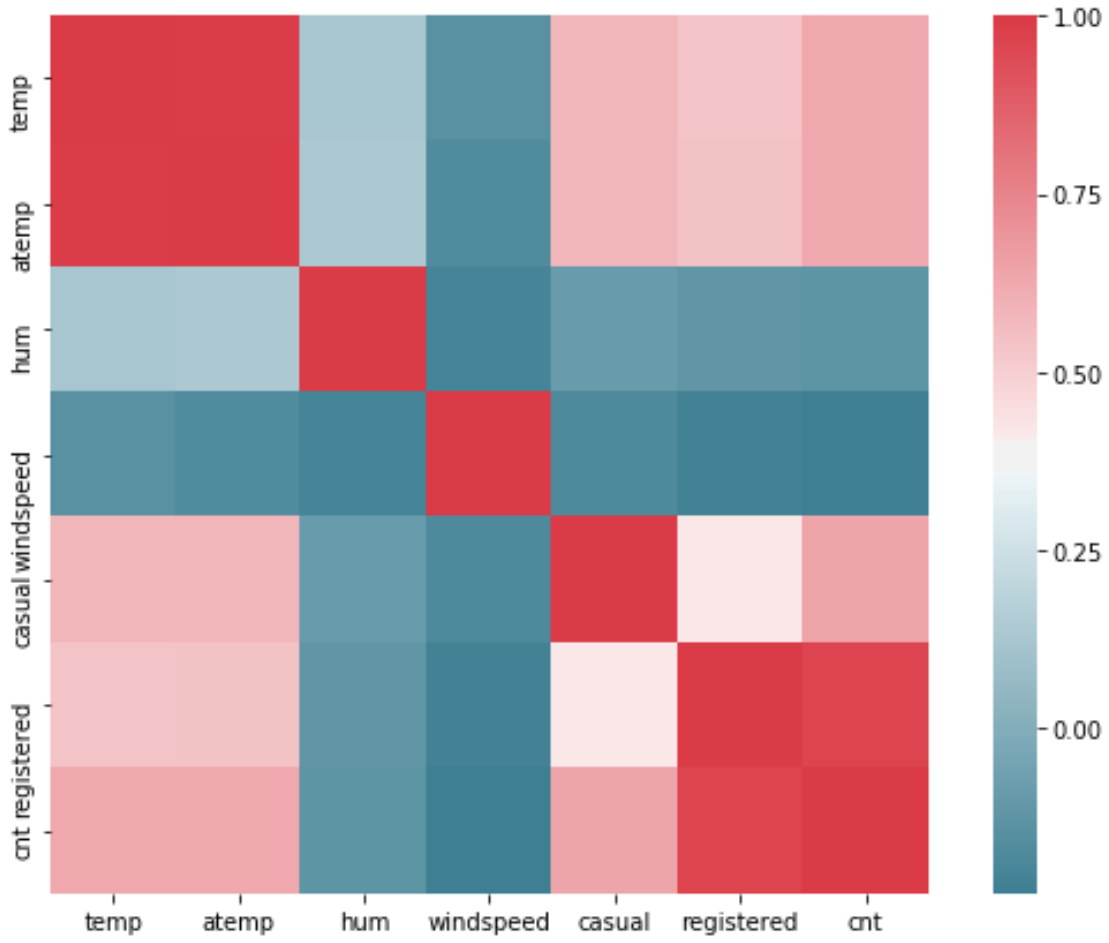
Outliers in "casual"

- Calculate outliers and inliers in each numeric variable.
- Calculate inter quartile range.
- Drop those observations which are beyond outliers and inliers.
- Checking again the shape of Dataset after removing outliers.

3) **Feature Selection**: Feature selection refers to selecting a subset of relevant features for use in model development. It refers to a subset of learning algorithms input variable upon which it should focus attention while ignoring the rest. It also reduces the dimensionality of the dataset.

The following set of steps are followed for feature selection:

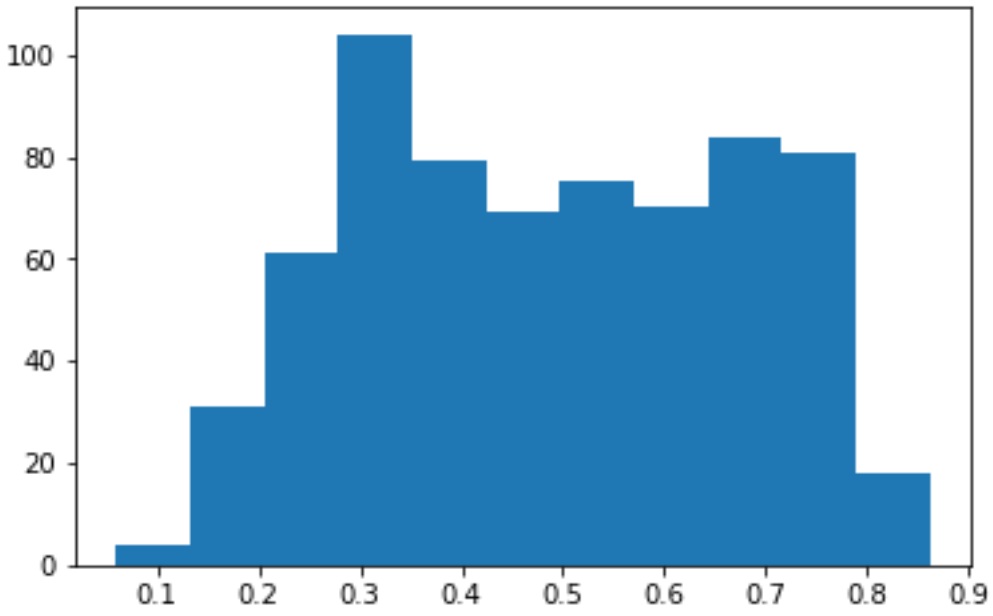
- Generate Correlation Matrix.
- Plot heatmap using seaborn library to identify correlation between the various continuous variables:



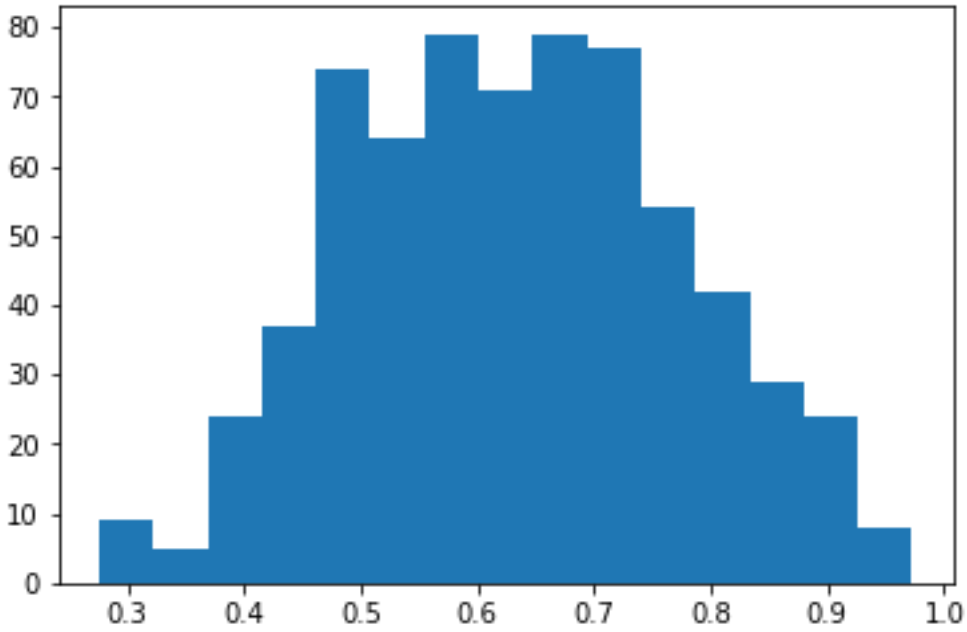
Heatmap for Feature selection

- As can be seen from the above correlation plot that there is a very high positive correlation between “temp” and “atemp”.
- Since our Dataset contain categorical variable so there is need to do Chi square test for feature selection.

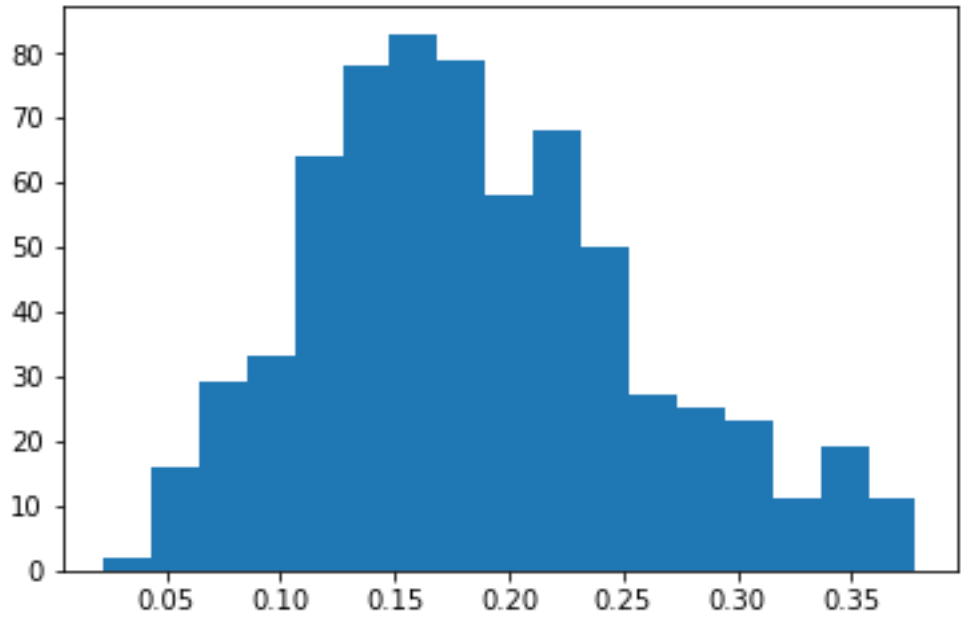
4) **Feature Scaling**: Feature scaling refers to the task of converting the various values of the column within the specified limits. Feature scaling is done to bring all the variable values within the limits so that no single variable has more influence over other variables while developing the model. It is the process of reducing the variations either within or between the variables. It brings all the variables into proper proportions with one another.
Feature scaling commonly uses two methods to compute.
   - Normalization
   - Standardisation



Distribution of "temp"



Distribution of "hum"



Distribution of "windspeed"

As can be seen from above histograms our variables are not normally distributed so we have to adopt Normalization method of feature scaling.

MODEL DEVELOPMENT

Machine learning is, programming computers to optimize a performance criterion using example data or past experience. During the development of this model we applied various machine learning algorithms.

- Divide data into train and test for model development and to check accuracy and deduce which model is best.

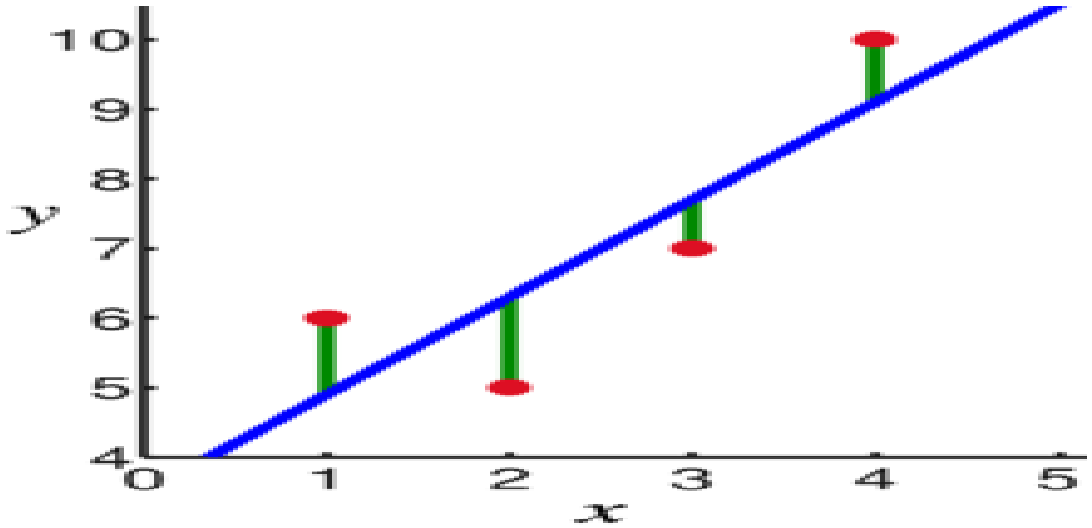
1) Linear Regression:

It is a Prediction Model. There are two types of Linear Regression

- Simple linear regression
- Multiple linear regression

It describes relationship among variables. The one simple case is where a dependent variable may be related to independent or explanatory variable.

- Equation expressing this relation is the line:
  $Y = b_0 + b_1x_1 + b_2x_2 + \ldots$
  Where b_0 is the intercept and x_1, x_2 are independent variables.
- For a given set of values we need to calculate values for b_0 and b_1.
- Look for a line which minimizes the sum of the residuals.

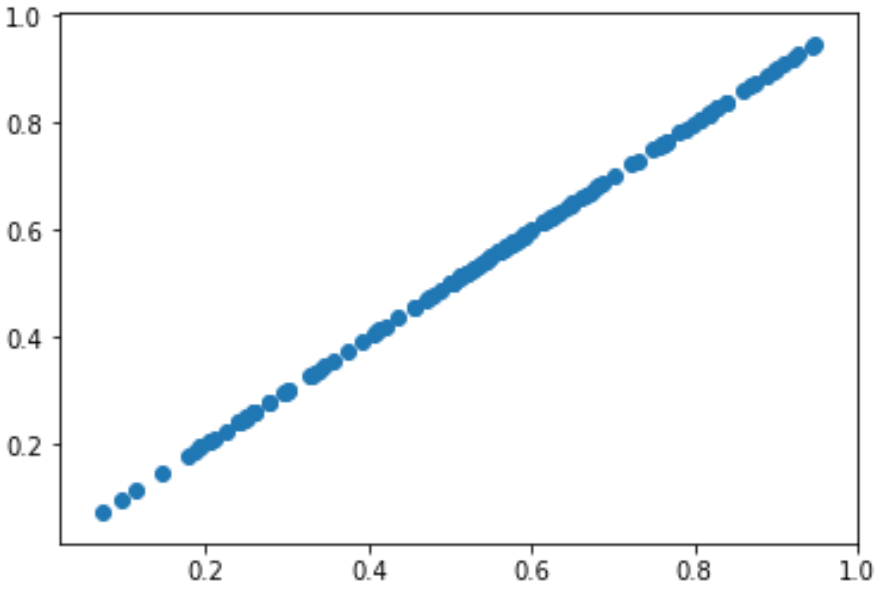


Linear Regression Line

- Import Libraries for Linear Regression.
- Build and train Linear Regression model using X_train and y_train.
- Predict model outcome using test data X_test.
- Regression Evaluation Metrics:

```
MAE: 1.329512480500107e-15
MAPE: 1.329512480500107e-13
MSE: 2.5982763363476973e-30
RMSE: 1.6119169756372992e-15
```

- Scatter Plot:



2) **Decision Tree**:

Decision Tree is a predictive model based on a branching series of Boolean tests. It can be used for classification and regression both type of target variables. There are number of different types of decision trees that can be used in Machine learning algorithms. Decision tree is a rule. Each branch connects nodes with “and” and multiple branches are connected by “or”. They are extremely easy to understand by the business users. Decision trees build some intuitions about your customer base. E.g. “are customers with different family sizes truly different?”

- Import Libraries.
- Model Development: Build and train model using X_train and y_train
- Predict model outcome using test data X_test
- Evaluation Metrics:

```
MAE: 0.0642289049213027
MAPE: 6.4228904921302705
MSE: 0.00604590404751492 55
RMSE: 0.07775541169278782
```

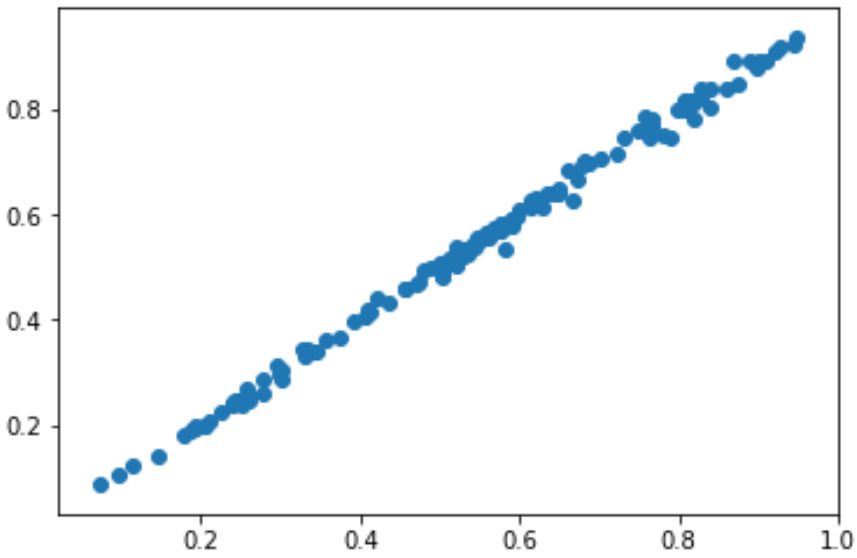
3) Random Forest:

Random forest is an ensemble that consists of many decision trees. The term came from random decision forests that was first proposed by Tin Kam Ho of Bell Labs in 1995. This method combines Breiman's "bagging" idea and the random selection of features. It outputs the class that is the mode of the class's output by individual trees. It calculates results by evaluating mean for regression and mode for classification. This algorithm can be used for both classification and regression

- Import Libraries
- Develop and train random forest model on train data using X_train and y_train
- Predict new test cases using X_test
- Evaluation Metrics:

```
MAE: 0.009039584140203335
MAPE: 0.9039584140203335
MSE: 0.00015948257204420502
RMSE: 0.01262864094208894
```

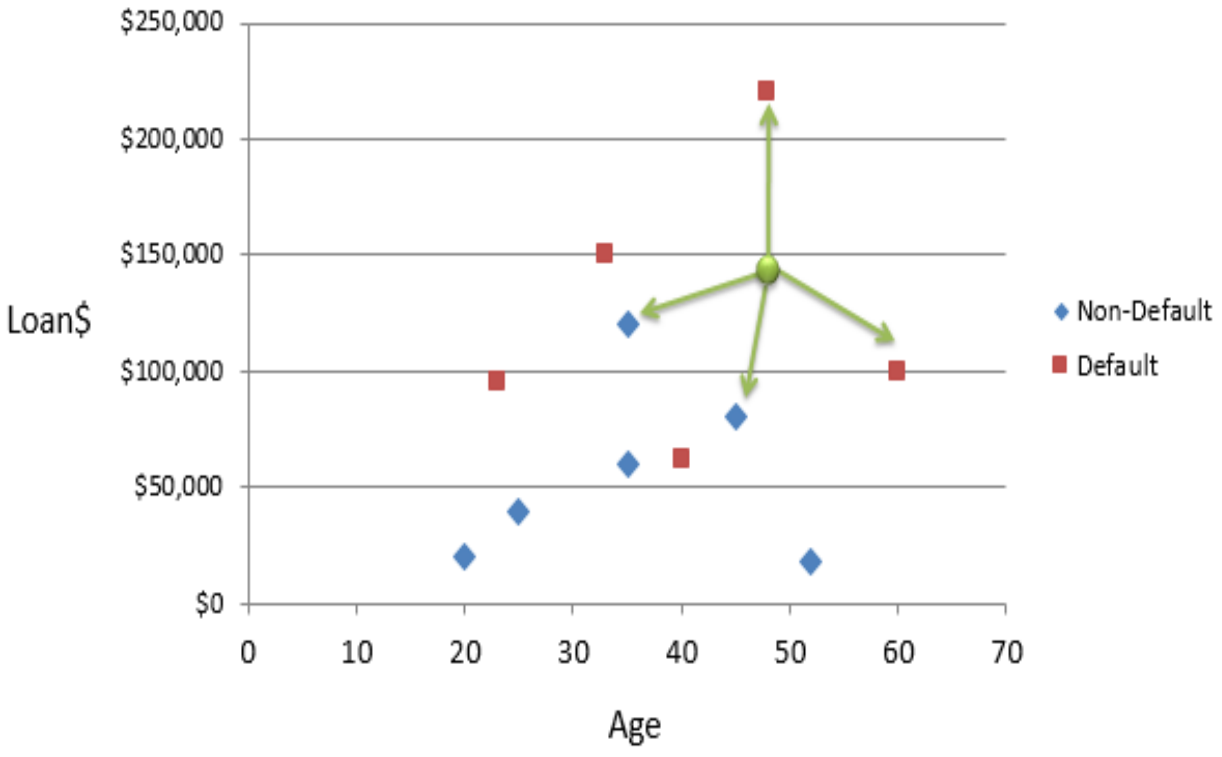
- Scatter Plot:



4) K Nearest Neighbour:

KNN stands for K-Nearest Neighbour. KNN is simple algorithm that stores all available cases and classifies new cases based on a similarity measure. It is a Supervised Machine Learning Algorithm. It can be used for both classification and regression. It is considered to be a Lazy Learning Algorithm.

Pick a number of neighbours you want to use for classification or regression. Choose a method to measure distances from test point. Keep a data set with records. For every new point, identify the number of nearest neighbours you picked using the method you choose. Let them vote if it is a classification or take a mean/median for regression

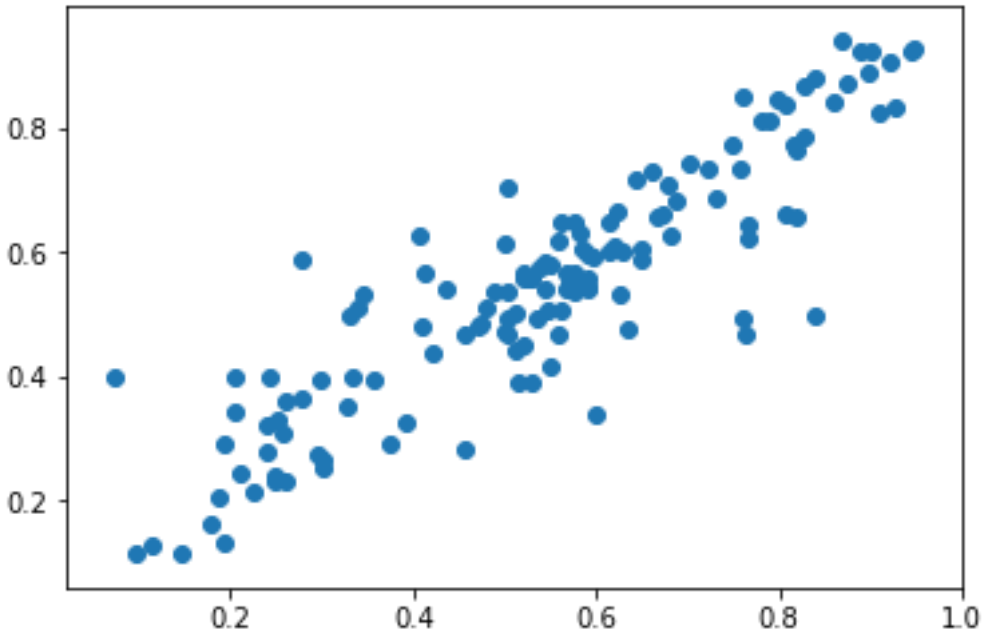


KNN Algorithm

- Import Libraries
- Develop and train KNN model on train data using X_train and y_train
- Predict new test cases using X_test
- Evaluation Metrics:

```
MAE: 0.06793765230297738
MAPE: 6.793765230297738
MSE: 0.009485599184749094
RMSE: 0.09739404080717204
```

- Scatter Plot:



SUMMARY OF MODELS

Now that we have a few models for predicting the target variable, we need to decide which one to choose. There are several criteria that exist for evaluating and comparing models. Based on MAPE we can deduce that Linear Regression is the best Algorithm to predict our target variable.

- DECISION TREE

  MAE for Decision tree = 0.07285
  MAPE for Decision tree = 7.285%
  MSE for Decision tree = 0.00713
  RMSE for Decision tree = 0.0844

- RANDOM FOREST

  MAE for Random Forest = 0.01188
  MAPE for Random Forest = 1.188%
  MSE for Random Forest = 0.0002834
  RMSE for Random Forest = 0.01683

- LINEAR REGRESSION

  MAE for Linear Regression = 1.32951e-15
  MAPE for Linear Regression = 1.32951e-13
  MSE for Linear Regression = 2.59827e-30
  RMSE for Linear Regression = 1.61191e-15

- KNN K NEAREST NEIGHBOUR

  MAE for KNN Regression = 0.07225
  MAPE for KNN Regression = 7.2257%
  MSE for KNN Regression = 0.009388
  RMSE for KNN Regression = 0.09689

PREDICTING TEST DATA

After forming the machine learning algorithm and tested it for test data we have fetch out from train dataset the most accurate method. Now we are ready to predict any external data given to us. Now since we are not given with any test data we create a sample dataset out of our given train data just to confirm the working of our model using the most appropriate algorithm we have formed in previous section. As we have deduced from MAPE values in our previous section that Linear Regression gives most accurate model so we will use Linear Regression to predict the target value.

Steps followed for predicting target variable in test data:

- Let us create example output with sample input
- Checking the shape of the test dataset
- Drop the "instant", "dteday", "atemp" column from the Dataset as our model is not trained for it.
- Now we apply our model to Linear Regression to predict our target variable "cnt".
- Our 'cnt_pred' column gets added to as the last column of the dataset.
- Writing this new dataset with predicted variable as a CSV file.

The CSV file of the target variable is also included.

APPENDIX: Python Code

```
import os
import pandas as pd
import numpy as np
from fancyimpute import KNN
import matplotlib.pyplot as plt
from scipy.stats import chi2_contingency
import seaborn as sns
from random import randrange, uniform
```

```
os.chdir("C:/Users/risha/Desktop/BIKE rental project/Bike_rental")
```

```
os.getcwd()
bikerental_train = pd.read_csv('day.csv')
bikerental_train.shape
bikerental_train.info()
bikerental_train.head(5)
bikerental_train.describe()
```

```
missing_val = pd.DataFrame(bikerental_train.isnull().sum())
cat_names =
["season","yr","mnth","holiday","weekday","workingday","weathersit"]
cont_names =
["temp","atemp","hum","windspeed","casual","registered","cnt"]
```

```
%matplotlib inline
plt.boxplot(bikerental_train['windspeed'])
plt.boxplot(bikerental_train['casual'])
for i in cont_names:
q75, q25 = np.percentile(bikerental_train.loc[:,i], [75 ,25])
iqr = q75 - q25
min = q25 - (iqr*1.5)
max = q75 + (iqr*1.5)
bikerental_train =
bikerental_train.drop(bikerental_train[bikerental_train.loc[:,i] < min].index)
bikerental_train =
bikerental_train.drop(bikerental_train[bikerental_train.loc[:,i] > max].index)
```

```
df_corr = bikerental_train.loc[:,cont_names]
f, ax = plt.subplots(figsize=(10, 7))
```

```
corr = df_corr.corr()
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool),
cmap=sns.diverging_palette(220, 10, as_cmap=True),square=True, ax=ax)
```

- ```
  for i in cat_names:
  print(i)
  chi2, p, dof, ex = chi2_contingency(pd.crosstab(bikerental_train['cnt'],
  bikerental_train[i]))
   print(p)
   print(chi2)
  ```

- ```
  bikerental_train = bikerental_train.drop(["atemp"], axis=1)
  ```
- ```
  %matplotlib inline
  ```
- ```
  plt.hist(bikerental_train["temp"], bins='auto')
  ```
- ```
  cont_names = ["temp","hum","windspeed","casual","registered","cnt"]
  ```
- ```
  for i in cont_names:
  bikerental_train[i] = (bikerental_train[i] -
  np.min(bikerental_train[i]))/(np.max(bikerental_train[i]) -
  np.min(bikerental_train[i]))
  ```

- ```
  from sklearn import tree
  from sklearn.metrics import accuracy_score
  from sklearn.model_selection import train_test_split
  from sklearn import metrics
  from sklearn.model_selection import cross_val_score
  ```

- ```
  X = bikerental_train.values[:, 2:14]
  Y = bikerental_train.values[:,14]
  X_train, X_test, y_train, y_test = train_test_split( X, Y, test_size = 0.2)
  ```

- ```
  DT_model = tree.DecisionTreeRegressor(max_depth=2).fit(X_train, y_train)
  ```
- ```
  predictions_DT = DT_model.predict(X_test)
  ```
- ```
  print('MAE:', metrics.mean_absolute_error(y_test, predictions_DT))
  MAPE = metrics.mean_absolute_error(y_test, predictions_DT)
  print("MAPE:", MAPE*100)
  print('MSE:', metrics.mean_squared_error(y_test, predictions_DT))
  print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions_DT)))
  ```

- ```
  from sklearn.ensemble import RandomForestRegressor
  ```
- ```
  RF_model = RandomForestRegressor(n_estimators = 10).fit(X_train, y_train)
  ```
- ```
  RF_Predictions = RF_model.predict(X_test)
  ```

- print('MAE:', metrics.mean_absolute_error(y_test, RF_Predictions))
  MAPE = metrics.mean_absolute_error(y_test, RF_Predictions)
  print("MAPE:", MAPE*100)
  print('MSE:', metrics.mean_squared_error(y_test, RF_Predictions))
  print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, RF_Predictions)))
- plt.scatter(y_test,RF_Predictions)

- from sklearn.linear_model import LinearRegression

- lm = LinearRegression()
  lm.fit(X_train,y_train)
- LR_predictions = lm.predict(X_test)
- print('MAE:', metrics.mean_absolute_error(y_test, LR_predictions))
  MAPE = metrics.mean_absolute_error(y_test, LR_predictions)
  print("MAPE:", MAPE*100)
  print('MSE:', metrics.mean_squared_error(y_test, LR_predictions))
  print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, LR_predictions)))
- plt.scatter(y_test,LR_predictions)

- from sklearn.neighbors import KNeighborsRegressor
- KNN_model = KNeighborsRegressor(n_neighbors = 3).fit(X_train, y_train)
- KNN_Predictions = KNN_model.predict(X_test)
- print('MAE:', metrics.mean_absolute_error(y_test, KNN_Predictions))
  MAPE = metrics.mean_absolute_error(y_test, KNN_Predictions)
  print("MAPE:", MAPE*100)
  print('MSE:', metrics.mean_squared_error(y_test, KNN_Predictions))
  print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, KNN_Predictions)))
- plt.scatter(y_test,KNN_Predictions)

- sample_bikerenting = bikerental_train.sample(500)
- sample_bikerenting = sample_bikerenting.drop(["instant", "dteday",], axis=1)
- sample_bikerenting = sample_bikerenting.drop(["cnt"], axis=1)
- sample_bikerenting["cnt_pred"] = lm.predict(sample_bikerenting)
- sample_bikerenting.to_csv('sample_bike_predict.csv',index=False)

Important Note

- INSTRUCTION TO RUN AND DEPLOY THE CODE IS WRITTEN WITH THE CODE ITSELF.
- ONLY THE FILE LOCATION SHOULD BE CHANGED IN OS.CHDIR FUNCTION ACCORDING TO THE USER FILE LOCATION.
- USER CAN SIMPLY USE SHIFT + ENTER TO GET ALL COMMAND RUN.
- FILES INCLUDED ARE:
    1. PYTHON CODE FILE
    2. PROJECT REPORT IN BOTH WORD AND PDF FORMATS
    3. R CODE FILE
    4. DATASET WITH TARGET VARIABLE IN .CSV FORMAT


Thank you