Bike Renting

Pesala Ravi Kumar

14/08/2019

Contents

Chapter 1

Introduction

1.1 Problem Statement

The objective of this Case is to Predication of bike rental count on daily based on the environmental and seasonal settings. Now a days there are many organizations who are running these bike rental work. In order to reduce the effort of renting the bike in different time periods and conditions by applying some machine learning concepts. We would like to predict the count of bike rental based on the conditions when the bike is renting by customers which are already known and easy to calculate further predictions.

1.2 Data

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
instant	dteday	season	yr	mnth	holiday	weekday	workingda	weathersi	temp	atemp	hum	windspee	casual	registerec	cnt	
1	########	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985	
2	########	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801	
3	########	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349	
4	########	1	0	1	0	2	1	1	0.2	0.212122	0.590435	0.160296	108	1454	1562	
5	########	1	0	1	0	3	1	1	0.226957	0.22927	0.436957	0.1869	82	1518	1600	
6	########	1	0	1	0	4	1	1	0.204348	0.233209	0.518261	0.089565	88	1518	1606	
7	########	1	0	1	0	5	1	2	0.196522	0.208839	0.498696	0.168726	148	1362	1510	
8	########	1	0	1	0	6	0	2	0.165	0.162254	0.535833	0.266804	68	891	959	
9	########	1	0	1	0	0	0	1	0.138333	0.116175	0.434167	0.36195	54	768	822	
10	########	1	0	1	0	1	1	1	0.150833	0.150888	0.482917	0.223267	41	1280	1321	
11	########	1	0	1	0	2	1	2	0.169091	0.191464	0.686364	0.122132	43	1220	1263	
12	########	1	0	1	0	3	1	1	0.172727	0.160473	0.599545	0.304627	25	1137	1162	
13	########	1	0	1	0	4	1	1	0.165	0.150883	0.470417	0.301	38	1368	1406	
14	########	1	0	1	0	5	1	1	0.16087	0.188413	0.537826	0.126548	54	1367	1421	
15	########	1	0	1	0	6	0	2	0.233333	0.248112	0.49875	0.157963	222	1026	1248	
16	########	1	0	1	0	0	0	1	0.231667	0.234217	0.48375	0.188433	251	953	1204	
17	########	1	0	1	1	1	0	2	0.175833	0.176771	0.5375	0.194017	117	883	1000	
18	########	1	0	1	0	2	1	2	0.216667	0.232333	0.861667	0.146775	9	674	683	
19	########	1	0	1	0	3	1	2	0.292174	0.298422	0.741739	0.208317	78	1572	1650	
20	########	1	0	1	0	4	1	2	0.261667	0.25505	0.538333	0.195904	83	1844	1927	

day

Fig 1.2.1 Data

As you can see in the fig.1.2.1 above we have the following 15 variables using which we have to predict the one of the variables.

1. Date
2. Season
3. Year
4. Month
5. Holiday

6. Weekday
7. Working Day
8. Weather
9. temp
10. atemp
11. Humidity
12. Casual
13. Registered
14. Count

Chapter 2

Methodology

2.1 Pre Processing

Any predictive modeling requires that we look at the data before we start modeling. However, in data mining terms looking at data refers to so much more than just looking. Looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is often called as Exploratory Data Analysis.

2.1.1 Missing Value Analysis

In statistics, missing data, or missing values, occur when no data value is stored for the variable in an observation. Missing data are a common occurrence and can have a significant effect on the conclusions that can be drawn from the data.

Why missing the value?

Human Error, refuse to answer, optional

In order to handle Missing Values, there are 2 cases Ignore or Impute Missing value.

Before Imputing Understand why value is missing and by plotting graphs.

Delete the observations where you not to impute.

Techniques to impute missing values:

1. Fill with central statistics like
    i. Mean
    ii. Mode
    iii. Median
2. Distance base/ Data Mining Method – KNN Imputation
3. Prediction method – Machine Learning models

Selecting the technique for missing values

1. Create a small subset of total data.
2. Delete some values manually.
3. Use multiple methods to fill.
4. See which technique fills correctly.
5. Select that technique for finding missing value analysis.

2.1.2 Outlier Analysis

Outlier, it is an observation which inconsistence to rest of data.

Causes of the outlier are,

1. Poor data quality or contamination.
2. Low quality measurements.
3. Manual error.
4. Malfunctioning equipment.
5. Correct but exceptional data

Effect of outliers is that it will gives data which is not present in data because of the outlier.

Assume data as 1, 3, 5, 7, and 14 now try to impute using mean value. Mean will be 6 which is not present under data so it might reflect the modelling.

Steps to detect an outlier are,

1. Detect variable with outlier using graphical tools
2. Replace all with NA
3. Apply the missing value analysis on NA records to impute new Values.

2.1.3 Feature Selection

It is also called as variable selection or attribute selection.

Selecting a subset of relevant features like variables, predictors for model construction.

Advantages of Feature selection is Dimensionality Reduction (Variable reduction).

Techniques to dimensionality reduction,

1. Correlation analysis.
2. Chi-square test of independence.

2.2 Modeling

2.2.1 Model Selection

In our early stages of analysis during preprocessing we have come to understand that in order to predict the count of bike rentals we need to predict the casual and registered bikes so that we can add those two and calculate the total count of bike rentals.

In order to predict total count we'll divide the data into 2 data sets with casual and registered variables separately.

Casual Dataset

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	index	season	yr	mnth	holiday	weekday	workingda	weathersi	temp	atemp	hum	windspee	casual	
2	1	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	
3	2	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	
4	3	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	
5	4	1	0	1	0	2	1	1	0.2	0.212122	0.590435	0.160296	108	
6	5	1	0	1	0	3	1	1	0.226957	0.22927	0.436957	0.1869	82	
7	6	1	0	1	0	4	1	1	0.204348	0.233209	0.518261	0.089565	88	
8	7	1	0	1	0	5	1	2	0.196522	0.208839	0.498696	0.168726	148	
9	8	1	0	1	0	6	0	2	0.165	0.162254	0.535833	0.266804	68	
10	9	1	0	1	0	0	0	1	0.138333	0.116175	0.434167	0.36195	54	
11	10	1	0	1	0	1	1	1	0.150833	0.150888	0.482917	0.223267	41	
12	11	1	0	1	0	2	1	2	0.169091	0.191464	0.686364	0.122132	43	
13	12	1	0	1	0	3	1	1	0.172727	0.160473	0.599545	0.304627	25	
14	13	1	0	1	0	4	1	1	0.165	0.150883	0.470417	0.301	38	
15	14	1	0	1	0	5	1	1	0.16087	0.188413	0.537826	0.126548	54	
16	15	1	0	1	0	6	0	2	0.233333	0.248112	0.49875	0.157963	222	
17	16	1	0	1	0	0	0	1	0.231667	0.234217	0.48375	0.188433	251	
18	17	1	0	1	1	1	0	2	0.175833	0.176771	0.5375	0.194017	117	
19	18	1	0	1	0	2	1	2	0.216667	0.232333	0.861667	0.146775	9	
20	19	1	0	1	0	3	1	2	0.292174	0.298422	0.741739	0.208317	78	
21	20	1	0	1	0	4	1	2	0.261667	0.25505	0.538333	0.195904	83	
22	21	1	0	1	0	5	1	1	0.1775	0.157833	0.457083	0.353242	75	
23	22	1	0	1	0	6	0	1	0.05913	0.07907	0.4	0.17197	93	
24	23	1	0	1	0	0	0	1	0.096522	0.098839	0.436522	0.2466	150	
25	24	1	0	1	0	1	1	1	0.097391	0.11793	0.491739	0.15833	86	
26	25	1	0	1	0	2	1	2	0.223478	0.234526	0.616957	0.129796	186	
27	26	1	0	1	0	3	1	3	0.2175	0.2036	0.8625	0.29385	34	
28	27	1	0	1	0	4	1	1	0.195	0.2197	0.6875	0.113837	15	
29	28	1	0	1	0	5	1	2	0.203478	0.223317	0.793043	0.1233	38	
30	29	1	0	1	0	6	0	1	0.196522	0.212126	0.651739	0.145365	123	
31	30	1	0	1	0	0	0	1	0.216522	0.250322	0.722174	0.073983	140	
32	31	1	0	1	0	1	1	2	0.180833	0.18625	0.60375	0.187192	42	
33	32	1	0	2	0	2	1	2	0.192174	0.23453	0.829565	0.053213	47	
34	33	1	0	2	0	3	1	2	0.26	0.254417	0.775417	0.264308	72	
35	34	1	0	2	0	4	1	1	0.186957	0.177878	0.437826	0.277752	61	
36	35	1	0	2	0	5	1	2	0.211304	0.228587	0.585217	0.127839	88	
37	36	1	0	2	0	6	0	2	0.233333	0.243058	0.929167	0.161079	100	

dataCasual

Registered Dataset

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1		season	yr	mnth	holiday	weekday	workingda	weathersi	temp	atemp	hum	windspee	registered	
2	1	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	654	
3	2	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	670	
4	3	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	1229	
5	4	1	0	1	0	2	1	1	0.2	0.212122	0.590435	0.160296	1454	
6	5	1	0	1	0	3	1	1	0.226957	0.22927	0.436957	0.1869	1518	
7	6	1	0	1	0	4	1	1	0.204348	0.233209	0.518261	0.089565	1518	
8	7	1	0	1	0	5	1	2	0.196522	0.208839	0.498696	0.168726	1362	
9	8	1	0	1	0	6	0	2	0.165	0.162254	0.535833	0.266804	891	
10	9	1	0	1	0	0	0	1	0.138333	0.116175	0.434167	0.36195	768	
11	10	1	0	1	0	1	1	1	0.150833	0.150888	0.482917	0.223267	1280	
12	11	1	0	1	0	2	1	2	0.169091	0.191464	0.686364	0.122132	1220	
13	12	1	0	1	0	3	1	1	0.172727	0.160473	0.599545	0.304627	1137	
14	13	1	0	1	0	4	1	1	0.165	0.150883	0.470417	0.301	1368	
15	14	1	0	1	0	5	1	1	0.16087	0.188413	0.537826	0.126548	1367	
16	15	1	0	1	0	6	0	2	0.233333	0.248112	0.49875	0.157963	1026	
17	16	1	0	1	0	0	0	1	0.231667	0.234217	0.48375	0.188433	953	
18	17	1	0	1	1	1	0	2	0.175833	0.176771	0.5375	0.194017	883	
19	18	1	0	1	0	2	1	2	0.216667	0.232333	0.861667	0.146775	674	
20	19	1	0	1	0	3	1	2	0.292174	0.298422	0.741739	0.208317	1572	
21	20	1	0	1	0	4	1	2	0.261667	0.25505	0.538333	0.195904	1844	
22	21	1	0	1	0	5	1	1	0.1775	0.157833	0.457083	0.353242	1468	
23	22	1	0	1	0	6	0	1	0.05913	0.07907	0.4	0.17197	888	
24	23	1	0	1	0	0	0	1	0.096522	0.098839	0.436522	0.2466	836	
25	24	1	0	1	0	1	1	1	0.097391	0.11793	0.491739	0.15833	1330	
26	25	1	0	1	0	2	1	2	0.223478	0.234526	0.616957	0.129796	1799	
27	26	1	0	1	0	3	1	3	0.2175	0.2036	0.8625	0.29385	472	
28	27	1	0	1	0	4	1	1	0.195	0.2197	0.6875	0.113837	416	
29	28	1	0	1	0	5	1	2	0.203478	0.223317	0.793043	0.1233	1129	
30	29	1	0	1	0	6	0	1	0.196522	0.212126	0.651739	0.145365	975	
31	30	1	0	1	0	0	0	1	0.216522	0.250322	0.722174	0.073983	956	
32	31	1	0	1	0	1	1	2	0.180833	0.18625	0.60375	0.187192	1459	
33	32	1	0	2	0	2	1	2	0.192174	0.23453	0.829565	0.053213	1313	
34	33	1	0	2	0	3	1	2	0.26	0.254417	0.775417	0.264308	1454	
35	34	1	0	2	0	4	1	1	0.186957	0.177878	0.437826	0.277752	1489	
36	35	1	0	2	0	5	1	2	0.211304	0.228587	0.585217	0.127839	1620	
37	36	1	0	2	0	6	0	2	0.233333	0.243058	0.929167	0.161079	905	

dataRegistered

Ready

You always start your model building from the simplest to more complex so after applying all the models select the best model according to accuracy and MAPE and etc.

2.2.2 Linear Regression

vif(saved_dataCasual[,-12])

```
>
> vif(saved_dataCasual[,-12])
    Variables       VIF
1      season  3.548413
2          yr  1.020253
3        mnth  3.333672
4     holiday  1.083126
5     weekday  1.024076
6  workingday  1.076392
7  weathersit  1.748741
8        temp 63.321299
9       atemp 64.343361
10        hum  1.918309
11  windspeed  1.199259
```

Fig 2.2.2.1 vif(Casualdata)

vifcor(saved_dataCasual[,-12], th = 1.0)

```
11  windspeed  1.199259
> vifcor(saved_dataCasual[,-12], th = 1.0)
No variable from the 11 input variables has collinearity problem.

The linear correlation coefficients ranges between:
min correlation ( temp ~ weekday ):  -0.0001699624
max correlation ( atemp ~ temp ):  0.9917016

---------- VIFs of the remained variables --------
    Variables       VIF
1      season  3.548413
2          yr  1.020253
3        mnth  3.333672
4     holiday  1.083126
5     weekday  1.024076
6  workingday  1.076392
7  weathersit  1.748741
8        temp 63.321299
9       atemp 64.343361
10        hum  1.918309
11  windspeed  1.199259
> lr_model_casual = lm(casual~., data = saved_dataCasual)
```

Fig 2.2.2.2 vifcor(CasualData)

lr_model_casual = lm(casual~., data = saved_dataCasual)

summary(lr_model_casual)

```
11  windspeed  1.199259
> lr_model_casual = lm(casual~., data = saved_dataCasual)
> summary(lr_model_casual)

Call:
lm(formula = casual ~ ., data = saved_dataCasual)

Residuals:
     Min       1Q   Median       3Q      Max 
-1258.79  -221.62   -13.06   179.80  1620.28 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  700.793    106.358   6.589 8.56e-11 ***
season        61.824     24.244   2.550 0.010977 *  
yr           286.685     28.861   9.933  < 2e-16 ***
mnth         -15.744      7.562  -2.082 0.037690 *  
holiday     -274.214     89.012  -3.081 0.002144 ** 
weekday       26.364      7.216   3.653 0.000278 ***
workingday  -828.251     31.882 -25.979  < 2e-16 ***
weathersit  -113.049     34.696  -3.258 0.001174 ** 
temp        1194.842    621.486   1.923 0.054931 .  
atemp        894.855    703.714   1.272 0.203920    
hum         -393.230    139.024  -2.829 0.004807 ** 
windspeed   -862.054    202.020  -4.267 2.24e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 386.3 on 719 degrees of freedom
Multiple R-squared:  0.6883,	Adjusted R-squared:  0.6835 
F-statistic: 144.3 on 11 and 719 DF,  p-value: < 2.2e-16
```

Fig 2.2.2.3 summary(linearRegressionModel)

vif(saved_dataRegistered[,-12])

```
> vif(saved_dataRegistered[,-12])
    Variables       VIF
1      season  3.548413
2          yr  1.020253
3        mnth  3.333672
4     holiday  1.083126
5     weekday  1.024076
6  workingday  1.076392
7  weathersit  1.748741
8        temp 63.321299
9       atemp 64.343361
10        hum  1.918309
11  windspeed  1.199259
```

Fig 2.2.2.4 vif(RegisteredData)

vifcor(saved_dataRegistered[,-12], th = 1.0)

```
> vifcor(saved_dataRegistered[,-12], th = 1.0)
No variable from the 11 input variables has collinearity problem.

The linear correlation coefficients ranges between:
min correlation ( temp ~ weekday ):  -0.0001699624
max correlation ( atemp ~ temp ):  0.9917016

---------- VIFs of the remained variables --------
    Variables       VIF
1      season  3.548413
2          yr  1.020253
3        mnth  3.333672
4     holiday  1.083126
5     weekday  1.024076
6  workingday  1.076392
7  weathersit  1.748741
8        temp 63.321299
9       atemp 64.343361
10        hum  1.918309
11  windspeed  1.199259
> lr_model_registered = lm(registered~., data = saved_dataRegistered)
> summary(lr_model_registered)
```

Fig 2.2.2.5 vifcor(RegisteredData)

lr_model_registered = lm(registered~., data = saved_dataRegistered)

summary(lr_model_registered)

```
> lr_model_registered = lm(registered~., data = saved_dataRegistered)
> summary(lr_model_registered)

Call:
lm(formula = registered ~ ., data = saved_dataRegistered)

Residuals:
    Min      1Q  Median      3Q     Max 
-3945.2  -350.7    77.2   430.5  1595.6 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   768.21     185.55   4.140 3.88e-05 ***
season        447.95      42.30  10.591  < 2e-16 ***
yr           1754.02      50.35  34.836  < 2e-16 ***
mnth          -23.24      13.19  -1.761 0.078618 .  
holiday      -244.78     155.29  -1.576 0.115403    
weekday        42.70      12.59   3.392 0.000733 ***
workingday    948.61      55.62  17.055  < 2e-16 ***
weathersit   -497.94      60.53  -8.226 9.04e-16 ***
temp          834.07    1084.23   0.769 0.441982    
atemp        2678.42    1227.69   2.182 0.029456 *  
hum          -625.63     242.54  -2.580 0.010091 *  
windspeed   -1695.52     352.44  -4.811 1.83e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 673.9 on 719 degrees of freedom
Multiple R-squared:  0.8163,	Adjusted R-squared:  0.8135 
F-statistic: 290.4 on 11 and 719 DF,  p-value: < 2.2e-16
```

Fig 2.2.2.6 summary(linearRegressionModel)

Chapter 3

Conclusion

3.1 Model Evaluation

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. We can compare the models using any of the following criteria:

1. Predictive performance
2. MAPE
3. MSE
4. RMSE

Predictive performance can be measured by comparing Predictions of the models with real values of the target variables, and calculating some average error measure.

3.1.1 MAPE (Mean Absolute Error)

Measures accuracy as the percentage of error.

It is calculated as the average of the unsigned percentage error, as shown in the example below: Many organizations focus primarily on the MAPE when assessing forecast accuracy.

3.1.2 MSE & RMSE (Mean Square Error & Root Mean Square Error)

Steps to calculate MSE,

1. Find the regression line.
2. Insert your X values into the linear regression equation to find the new Y values
3. Subtract the new Y value from the original to get the error.
4. Square the errors.
5. Add up the errors.
6. Find the mean.

Calculate RMSE as root of MSE.

Appendix

R code:

```
rm(list=ls())
install.packages(c("dmm","dplyr","plyr","reshape","ggplot2","data.table","
psych","usdm","caret","DMwR"))
data=read.csv("day.csv",header=T)
newData=subset(data, select =
c("season","yr","mnth","holiday","weekday","workingday","weathersit","temp
","atemp","hum","windspeed","casual","registered","cnt"))
savedData=newData
dataCasual = subset(newData, select =
c("season","yr","mnth","holiday","weekday","workingday","weathersit","temp
","atemp","hum","windspeed","casual"))
saved_dataCasual = dataCasual;
dataRegistered = subset(newData, select =
c("season","yr","mnth","holiday","weekday","workingday","weathersit","temp
","atemp","hum","windspeed","registered"))
saved_dataRegistered = dataRegistered
write.csv(dataCasual, "dataCasual.csv",row.names = T)
write.csv(dataRegistered, "dataRegistered.csv",row.names = T)


# Retrive numeric data
numeric_index_casual = sapply(dataCasual,is.numeric)
numeric_data_casual = dataCasual[,numeric_index_casual]
numeric_data_cols_casual = colnames(numeric_data_casual)

numeric_index_registered = sapply(dataRegistered,is.numeric)
numeric_data_registered = dataRegistered[,numeric_index_registered]
numeric_data_cols_registered = colnames(numeric_data_registered)

#Calculate outliers in dataCasual
dataCasual1 = dataCasual
for(i in numeric_data_cols_casual)
{
  print(i)
  val_casual =
dataCasual1[,i][dataCasual1[,i]%in%boxplot.stats(dataCasual1[,i])$out]
  print(length(val_casual))
  dataCasual1 = dataCasual1[which(!dataCasual1[,i]%in%val_casual),]
}
# Replace all outliers in dataCasual with NA and impute missing using
missing value analysis
dataCasual1 = dataCasual
for(i in numeric_data_cols_casual)
{
  val_casual =
dataCasual1[,i][dataCasual1[,i]%in%boxplot.stats(dataCasual1[,i])$out]
  dataCasual1[,i][dataCasual1[,i]%in%val_casual] = NA
```

```
}

#Calculate outliers in dataRegistered
dataRegistered1 = dataRegistered
for(i in numeric_data_cols_registered)
{
  print(i)
  val_registered =
dataRegistered1[,i][dataRegistered1[,i]%in%boxplot.stats(dataRegistered1[,
i])$out]
  print(length(val_registered))
  dataRegistered1 =
dataRegistered1[which(!dataRegistered1[,i]%in%val_registered),]
}
# Replace all outliers in dataRegistered with NA and impute missing using
missing value analysis
dataRegistered1 = dataRegistered
for(i in numeric_data_cols_registered)
{

val_registered=dataRegistered1[,i][dataRegistered1[,i]%in%boxplot.stats(da
taRegistered1[,i])$out]
  dataRegistered1[,i][dataRegistered1[,i]%in%val_registered] = NA
}

# Apply KNN imputation for dataCasual
require(DMwR)
dataCasual1 = knnImputation(dataCasual1,k=5)

# Apply Mean imputation for columns with NA (because of error: there are
not sufficient cases)
dataCasual1$holiday[is.na(dataCasual1$holiday)] =
mean(dataCasual1$holiday,na.rm = T)
dataCasual1$hum[is.na(dataCasual1$hum)] = mean(dataCasual1$hum,na.rm = T)
dataCasual1$windspeed[is.na(dataCasual1$windspeed)] =
mean(dataCasual1$windspeed,na.rm = T)
dataCasual1$casual[is.na(dataCasual1$casual)] =
mean(dataCasual1$casual,na.rm = T)

# Apply KNN imputation for dataRegistered
dataRegistered1 = knnImputation(dataRegistered1,k=5)

# Apply Mean imputation for columns with NA (because of error: there are
not sufficient cases)
dataRegistered1$holiday[is.na(dataRegistered1$holiday)] =
mean(dataRegistered1$holiday,na.rm = T)
dataRegistered1$hum[is.na(dataRegistered1$hum)] =
mean(dataRegistered1$hum,na.rm = T)
dataRegistered1$windspeed[is.na(dataRegistered1$windspeed)] =
mean(dataRegistered1$windspeed,na.rm = T)
dataRegistered1$registered[is.na(dataRegistered1$registered)] =
mean(dataRegistered1$registered,na.rm = T)
```

```
# chi-square test of Independency
# factor_index=sapply(dataCasual1, is.factor)
# factor_data=dataCasual1[,factor_index]
#
# for (i in 1:ncol(dataCasual1)-1)
# {
#
print(chisq.test(table(dataCasual1$dataCasual1[,length(dataCasual1)],dataC
asual1[,i])))
# }

# select Dependent columns from dataCasual
nonDependentCols_casual = names(dataCasual1)%in% c("casual")
dependentData_casual = dataCasual1[!nonDependentCols_casual]
dependentCols_casual = colnames(dependentData_casual)
# select Dependent columns from dataRegistered
nonDependentCols_registered = names(dataRegistered1)%in% c("registered")
dependentData_registered = dataRegistered1[!nonDependentCols_registered]
depedentCols_registered = colnames(dependentData_registered)

# Feature Scaling
# require(graphics)
# for(i in dependentCols_casual)
# {
#   print(i)
#   qqnorm(dependentData_casual$i)
#   hist(dependentData_casual$i)
# }

# Sampling Techniques
d_dataCasual = dependentData_casual
simpleRandomSampling_dataCasual =
d_dataCasual[sample(nrow(d_dataCasual),100,replace = F),]
d_dataRegistered = dependentData_registered
simpleRandomSampling_dataRegistered =
d_dataRegistered[sample(nrow(d_dataRegistered),100,replace = F),]

# # divide train & test data
train_index_casual = sample(1:nrow(saved_dataCasual), 0.8 *
nrow(saved_dataCasual), prob = NULL)
train_data_casual = saved_dataCasual[train_index_casual,]
test_data_casual = saved_dataCasual[-train_index_casual,]
train_index_registered = sample(1:nrow(saved_dataRegistered), 0.8 *
nrow(saved_dataRegistered), prob = NULL)
train_data_registered = saved_dataRegistered[train_index_registered,]
test_data_registered = saved_dataRegistered[-train_index_registered,]
# # builds decision tree
# library(rpart)
# fit = rpart(registered~., data=train_data_registered, method="anova")
# library(MASS)
# predictions = predict(fit, test_data_registered[,-12])
# # Calculate MAPE, MSE, RMSE, MAE
# library(DMwR)
```

```
# regr.eval(test_data_registered[,12], predictions, stats =
c('mae','mape','mse','rmse'))
# mae         mape         mse         rmse
# 5.827114e+02 2.279066e+00 7.503266e+05 8.662140e+02


# KNN
# install.packages("caret")
# train_index_casual = sample(1:nrow(saved_dataCasual), 0.8 *
nrow(saved_dataCasual), prob = NULL)
# train_data_casual = saved_dataCasual[train_index_casual,]
# test_data_casual = saved_dataCasual[-train_index_casual,]
# train_index_registered = sample(1:nrow(saved_dataRegistered), 0.8 *
nrow(saved_dataRegistered), prob = NULL)
# train_data_registered = saved_dataRegistered[train_index_registered,]
# test_data_registered = saved_dataRegistered[-train_index_registered,]
# library(class)
# knn_predictions_casual = knn(train_data_casual[,1:11],
test_data_casual[,1:11], train_data_casual$casual, k=5)
# knn_predictions_registered = knn(train_data_registered[,1:11],
test_data_registered[,1:11], train_data_registered$registered, k=5)
# #Accuracy
# knn_CM_casual = table(knn_predictions_casual, test_data_casual$casual)
# sum(diag(knn_CM_casual))/nrow(test_data_casual)
# TN_casual = knn_CM_casual[0,0]
# FN_casual = knn_CM_casual[1,0]
# TP_casual = knn_CM_casual[1,1]
# FP_casual = knn_CM_casual[0,1]
#
# knn_CM_registered = table(knn_predictions_registered,
test_data_registered$registered)
# sum(diag(knn_CM_registered))/nrow(test_data_registered)
# TN_registered = knn_CM_registered[0,0]
# FN_registered = knn_CM_registered[1,0]
# TP_registered = knn_CM_registered[1,1]
# FP_registered = knn_CM_registered[0,1]
#
# library(DMwR)
# regr.eval(test_data_casual[,12], knn_predictions_casual, stats =
c('mae','mape','mse','rmse'))
# regr.eval(test_data_registered[,12], knn_predictions_registered, stats =
c('mae','mape','mse','rmse'))

# Linear Regression (Registered)
install.packages("usdm")
library(usdm)

vif(saved_dataCasual[,-12])
vifcor(saved_dataCasual[,-12], th = 1.0)
lr_model_casual = lm(casual~., data = saved_dataCasual)
summary(lr_model_casual)
lr_prediction_casual = predict(lr_model_casual, test_data_casual[,1:11])
library(DMwR)
library(MASS)
```

```
regr.eval(test_data_registered[,12], lr_prediction_casual, stats =
c('mae','mape','mse','rmse'))


vif(saved_dataRegistered[,-12])
vifcor(saved_dataRegistered[,-12], th = 1.0)
lr_model_registered = lm(registered~., data = saved_dataRegistered)
summary(lr_model_registered)
lr_prediction_registered = predict(lr_model_registered,
test_data_casual[,1:11])
library(DMwR)
library(MASS)
regr.eval(test_data_registered[,12], lr_prediction_registered, stats =
c('mae','mape','mse','rmse'))


# KMeans Clustering
install.packages("NbClust")
library(NbClust)
d_casual = saved_dataCasual
clusters_casual = NbClust(d_casual, min.nc = 2, max.nc = 10, method =
"kmeans")
barplot(table(clusters_casual$Best.nc[1,]), xlab="X", ylab="Y", main="")
kmeans_model_casual = kmeans(d_casual,4,nstart = 25)
cluster_accuracy_casual =
table(d_casual$casual,kmeans_model_casual$cluster)

d_registered = saved_dataRegistered
clusters_registered = NbClust(d_registered, min.nc = 2, max.nc = 10,
method = "kmeans")
barplot(table(clusters_registered$Best.nc[1,]), xlab="X", ylab="Y",
main="")
kmeans_model_registered = kmeans(d_registered,4,nstart = 25)
cluster_accuracy_registered =
table(d_registered$registered,kmeans_model_registered$cluster)

library(ggplot2)
library(scales)
library(psych)
library(gplots)

newData_casual = dataCasual
newData_registered = dataRegistered

# Bar plot ( Categorical variables VS Target variable)
# Casual Data
ggplot(newData_casual, aes_string(x=newData_casual$season,
y=newData_casual$casual)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("season") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Season vs Casual ") + theme(text =
element_text(size=10))
```

```
ggplot(newData_casual, aes_string(x=newData_casual$mnth,
y=newData_casual$casual)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("month") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Month vs Casual ") + theme(text =
element_text(size=10))

ggplot(newData_casual, aes_string(x=newData_casual$holiday,
y=newData_casual$casual)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("holiday") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Holiday vs Casual ") + theme(text =
element_text(size=10))

ggplot(newData_casual, aes_string(x=newData_casual$weekday,
y=newData_casual$casual)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("weekday") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Weekday vs Casul ") + theme(text =
element_text(size=10))

ggplot(newData_casual, aes_string(x=newData_casual$workingday,
y=newData_casual$casual)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("workingday") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Working Day vs Casual") + theme(text =
element_text(size=10))

ggplot(newData_casual, aes_string(x=newData_casual$weathersit,
y=newData_casual$casual)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("Weather") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot weather vs casual ") + theme(text =
element_text(size=10))

# Registered Data
ggplot(newData_registered, aes_string(x=newData_registered$season,
y=newData_registered$registered)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("season") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Season vs Registered ") + theme(text =
element_text(size=10))

ggplot(newData_registered, aes_string(x=newData_registered$mnth,
y=newData_registered$registered)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("month") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
```

```
  ggtitle("Bar plot Month vs Registered ") + theme(text =
element_text(size=10))

ggplot(newData_registered, aes_string(x=newData_registered$holiday,
y=newData_registered$registered)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("holiday") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Holiday vs Registered ") + theme(text =
element_text(size=10))

ggplot(newData_casual, aes_string(x=newData_casual$weekday,
y=newData_casual$casual)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("weekday") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Weekday vs Registered ") + theme(text =
element_text(size=10))

ggplot(newData_registered, aes_string(x=newData_registered$workingday,
y=newData_registered$registered)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("workingday") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Working Day vs Registered") + theme(text =
element_text(size=10))

ggplot(newData_registered, aes_string(x=newData_registered$weathersit,
y=newData_registered$registered)) +
  geom_bar(stat = "identity",fill="Blue") + theme_bw() +
  xlab("Weather") + ylab("casual") +
  scale_y_continuous(breaks = pretty_breaks(n=10)) +
  ggtitle("Bar plot Weather vs Registered ") + theme(text =
element_text(size=10))
```

Python code:

```
import os

os.getcwd()
os.chdir("C:/Users/gopin/Documents/R/BikeRental-Project")

import pandas as pd
import numpy as np
import matplotlib as mlt
import matplotlib.pyplot as plt
from sklearn.neighbors import KNeighborsClassifier

data = pd.read_csv("day.csv")
```

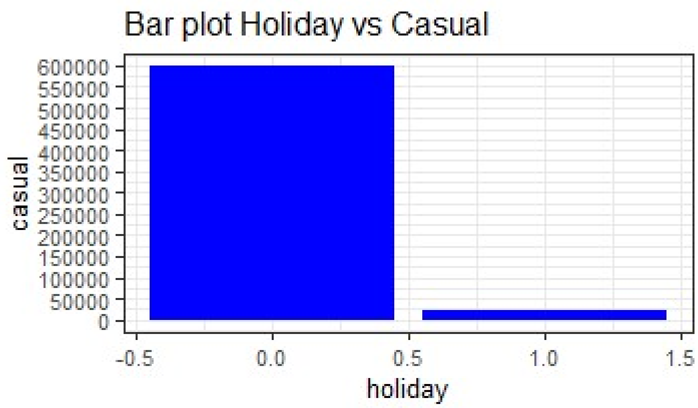
```
savedData = data

dataCasual =
savedData[["season","yr","mnth","holiday","weekday","workingday","weathers
it","temp","atemp","hum","windspeed","casual"]]
dataRegsitered =
savedData[["season","yr","mnth","holiday","weekday","workingday","weathers
it","temp","atemp","hum","windspeed","registered"]]

dCasual = dataCasual.copy()
dRegistered = dataRegsitered.copy()

# Store continuous variable names
cnames_C =
["season","yr","mnth","holiday","weekday","workingday","weathersit","temp"
,"atemp","hum","windspeed","casual"]
cnames_R =
["season","yr","mnth","holiday","weekday","workingday","weathersit","temp"
,"atemp","hum","windspeed","registered"]

# Detect outliers & delete(Casual)
for i in cnames_C:
    q75, q25 = np.percentile(dCasual.loc[:,i],[75,25])
    iqr = q75 - q25
    innerfence = q25 - (iqr * 1.5)
    outerfence = q75 + (iqr * 1.5)
    dCasual = dCasual.drop(dCasual[dCasual.loc[:,i] < innerfence].index)
    dCasual = dCasual.drop(dCasual[dCasual.loc[:,i] > outerfence].index)
# Replace with NA
dCasual = dataCasual.copy()
for i in cnames_C:
    q75, q25 = np.percentile(dCasual.loc[:,i],[75,25])
    iqr = q75 - q25
    innerfence = q25 - (iqr * 1.5)
    outerfence = q75 + (iqr * 1.5)
    dCasual.loc[dCasual[i] < innerfence,:i] = np.nan
    dCasual.loc[dCasual[i] > outerfence,:i] = np.nan
# Calculate Missing values
missing_C = pd.DataFrame(dCasual.isnull().sum())
# Impute using Mode method
for i in cnames_C:
    dCasual[i] = dCasual[i].fillna(dCasual[i].mode()[0])
#missing_C = pd.DataFrame(dCasual.isnull().sum())
savedDataCasual = dCasual

# Detect outliers & delete (Registered)
for i in cnames_R:
    q75, q25 = np.percentile(dRegistered.loc[:,i],[75,25])
    iqr = q75 - q25
    innerfence = q25 - (iqr * 1.5)
    outerfence = q75 + (iqr * 1.5)
    dRegistered = dRegistered.drop(dRegistered[dRegistered.loc[:,i] <
innerfence].index)
```

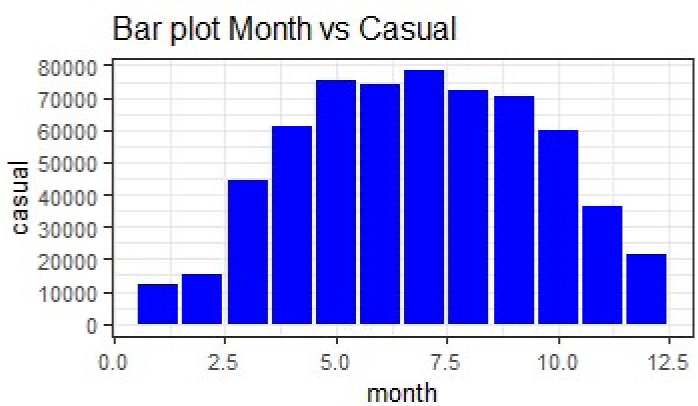
```
    dRegistered = dRegistered.drop(dRegistered[dRegistered.loc[:,i] >
outerfence].index)
# Replace with NA
dRegistered = dRegistered.copy()
for i in cnames_R:
    q75, q25 = np.percentile(dRegistered.loc[:,i],[75,25])
    iqr = q75 - q25
    innerfence = q25 - (iqr * 1.5)
    outerfence = q75 + (iqr * 1.5)
    dRegistered.loc[dRegistered[i] < innerfence,:i] = np.nan
    dRegistered.loc[dRegistered[i] > outerfence,:i] = np.nan
# Calculate Missing values
missing_R = pd.DataFrame(dRegistered.isnull().sum())
# Impute using Mode method
for i in cnames_R:
    dRegistered[i] = dRegistered[i].fillna(dRegistered[i].mode()[0])

savedDataRegistered = dRegistered

# Feature selection
'''import seaborn as sns
from scipy.stats import chi2_contingency
from random import randrange,uniform
for i in dCasual.columns:
    print(i)
    p = chi2_contingency(pd.crosstab(dCasual['casual'],dCasual[i]))

'''
# Gives Barplot
# %matplotlib inline
# plt.hist(dCasual['weekday'], bins='auto')

# Sampling using Systematic sampling
simpleRandomSampling_C = dCasual.sample(100)
simpleRandomSampling_R = dRegistered.sample(100)

from sklearn import tree
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

# Train and Test data
dCasual = savedDataCasual.copy()
xc = dCasual.values[:, 0:11]
yc = dCasual.values[:, 11]
xc_train, xc_test, yc_train, yc_test =
train_test_split(xc,yc,test_size=0.2)

# Linear Regression model for Data Casual
import statsmodels.api as sm
train_c, test_c = train_test_split(dCasual,test_size=0.2)
model_C = sm.OLS(train_c.iloc[:,11], train_c.iloc[:,0:11]).fit()
model_C.summary()

dRegistered = savedDataRegistered.copy()
```

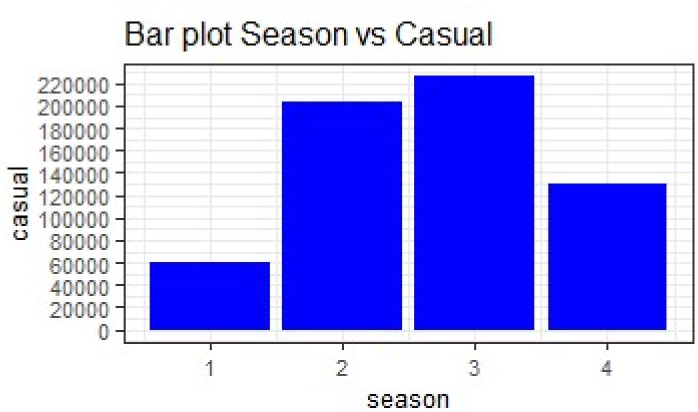
```
xr = dRegistered.values[:, 0:11]
yr = dRegistered.values[:, 11]
xr_train, xr_test, yr_train, yr_test =
train_test_split(xr,yr,test_size=0.2)

# Linear Regression model for Data Registered
import statsmodels.api as sm
train_c, test_c = train_test_split(dRegistered,test_size=0.2)
model_R = sm.OLS(train_c.iloc[:,11], train_c.iloc[:,0:11]).fit()
model_R.summary()

import ggplot
from ggplot import *
dC = savedDataCasual.copy()

ggplot(dC, aes(x='season', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Season") + ylab("Casual") + ggtitle("Barplot_seasonVScasual") +
theme.bw()

ggplot(dC, aes(x='holiday', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Holiday") + ylab("Casual") + ggtitle("Barplot_holidayVScasual")
+ theme.bw()

ggplot(dC, aes(x='mnth', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Month") + ylab("Casual") + ggtitle("Barplot_monthVScasual") +
theme.bw()

ggplot(dC, aes(x='weather', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Weather") + ylab("Casual") + ggtitle("Barplot_weatherVScasual")
+ theme.bw()

ggplot(dC, aes(x='weekday', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Weekday") + ylab("Casual") + ggtitle("Barplot_weekDayVScasual")
+ theme.bw()

ggplot(dC, aes(x='workkingday', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Working Day") + ylab("Casual") +
ggtitle("Barplot_workingDayVScasual") + theme.bw()

ggplot(dC, aes(x='season', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
```

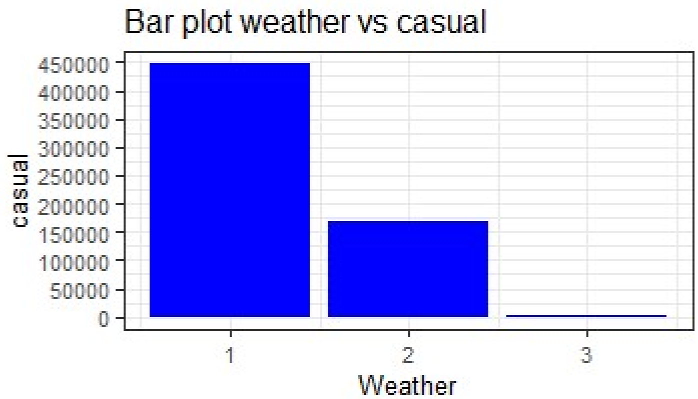
```
    xlab("Season") + ylab("Casual") + ggtitle("Barplot_seasonVScasual") +
theme.bw()


dR = savedDataRegistered.copy()
ggplot(dR, aes(x='season', y='registered')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Season") + ylab("Registered") +
ggtitle("Barplot_seasonVSregistered") + theme.bw()

ggplot(dR, aes(x='holiday', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Holiday") + ylab("Registered") +
ggtitle("Barplot_holidayVSregistered") + theme.bw()

ggplot(dR, aes(x='mnth', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Month") + ylab("Registered") +
ggtitle("Barplot_monthVSregistered") + theme.bw()

ggplot(dR, aes(x='weather', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Weather") + ylab("Registered") +
ggtitle("Barplot_weatherVSregistered") + theme.bw()

ggplot(dR, aes(x='weekday', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Weekday") + ylab("Registered") +
ggtitle("Barplot_weekDayVSregistered") + theme.bw()

ggplot(dR, aes(x='workkingday', y='casual')) +\
    geom_bar(fill="blue") +\
    scale_color_brewer(type="diverging", palette=4) +\
    xlab("Working Day") + ylab("Registered") +
ggtitle("Barplot_workingDayVSregistered") + theme.bw()
```

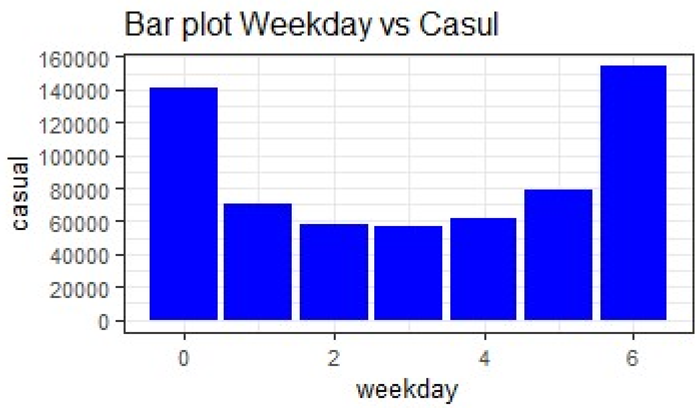
Visualizations:

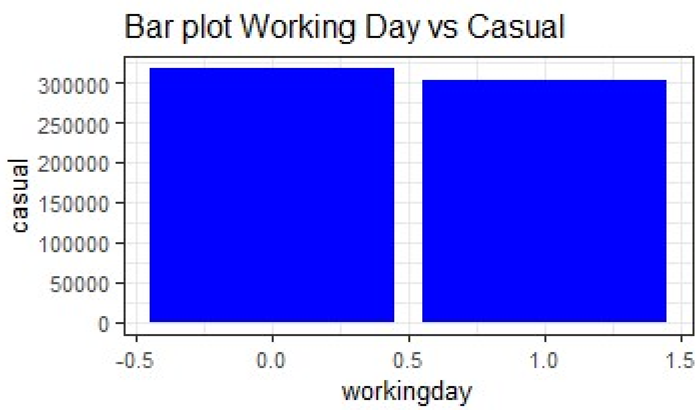


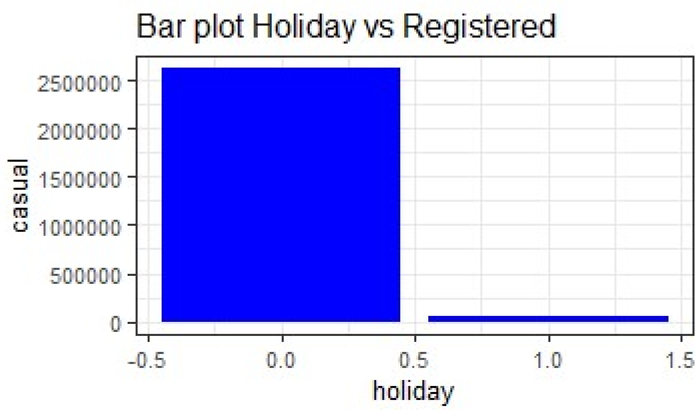


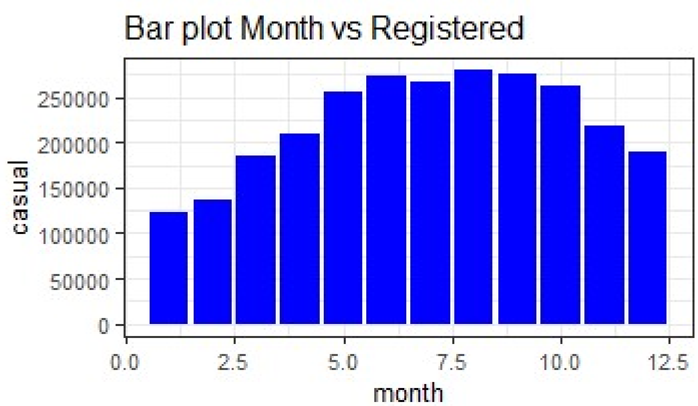


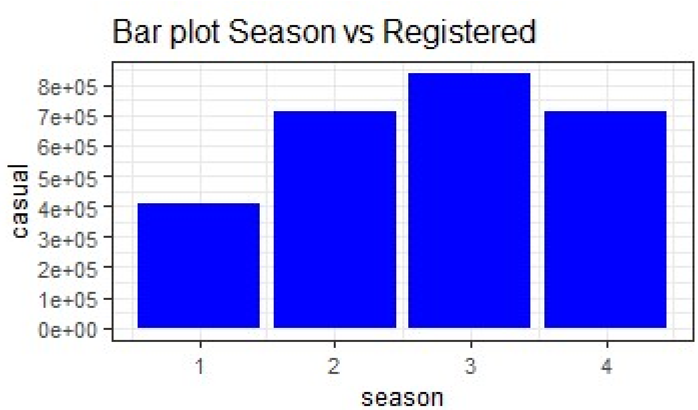

Bar plot weather vs casual
casual
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
1
2
3
Weather

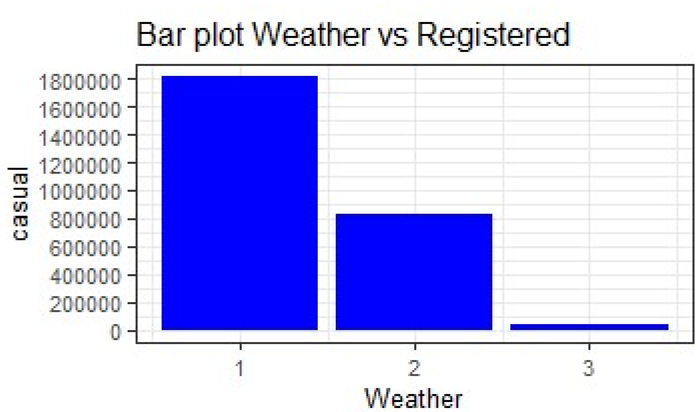

Bar plot Weekday vs Casul
casual
160000
140000
120000
100000
80000
60000
40000
20000
0
0
2
4
6
weekday

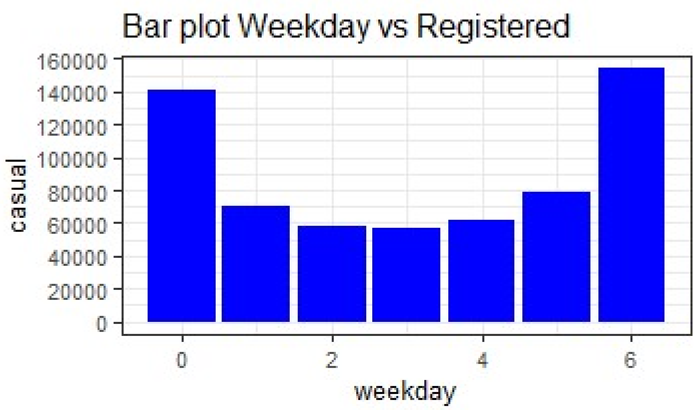

Bar plot Working Day vs Casual
casual
300000
250000
200000
150000
100000
50000
0
-0.5
0.0
0.5
1.0
1.5
workingday

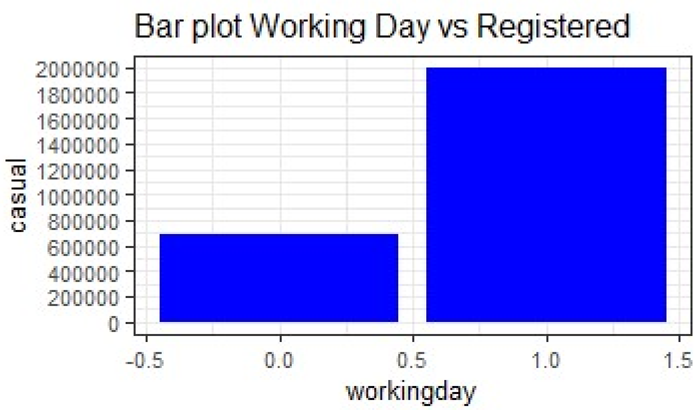

Bar plot Holiday vs Registered
casual
2500000
2000000
1500000
1000000
500000
0
-0.5
0.0
0.5
1.0
1.5
holiday


Bar plot Month vs Registered
casual
250000
200000
150000
100000
50000
0
0.0
2.5
5.0
7.5
10.0
12.5
month


Bar plot Season vs Registered
casual
8e+05
7e+05
6e+05
5e+05
4e+05
3e+05
2e+05
1e+05
0e+00
1
2
3
4
season


Bar plot Weather vs Registered
casual
1800000
1600000
1400000
1200000
1000000
800000
600000
400000
200000
0
1
2
3
Weather


Bar plot Weekday vs Registered
casual
160000
140000
120000
100000
80000
60000
40000
20000
0
0
2
4
6
weekday


Bar plot Working Day vs Registered
casual
2000000
1800000
1600000
1400000
1200000
1000000
800000
600000
400000
200000
0
-0.5
0.0
0.5
1.0
1.5
workingday