Bike Renting

Sanjib Kumar Mishra

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.

1.2 Data

Dataset Details:

#R code to getting the no. of variables and observations in the dataset (rent).

```
dim(rent)
```

Output:

```
[1] 731  16
```

Here it shows the dataset is having 731 observations and 16 variables

The details of the 16 data attributes in the dataset are as follows -
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)
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

*Here “cnt” is my target variable.

1.3 Exploratory Data Analysis

Before doing the exploratory analysis let’s check the structure of the dataset.

#Structure of the dataset

```
str(rent)
'data.frame':	731 obs. of  16 variables:
 $ instant   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : Factor w/ 731 levels "2011-01-01","2011-01-02",..: 1 2 3 4 5 6 7 8 9 10
...
 $ season    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ yr        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ mnth      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ weekday   : int  6 0 1 2 3 4 5 6 0 1 ...
 $ workingday: int  0 0 1 1 1 1 1 0 0 1 ...
 $ weathersit: int  2 2 1 1 1 1 2 2 1 1 ...
 $ temp      : num  0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num  0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num  0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num  0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : int  331 131 120 108 82 88 148 68 54 41 ...
 $ registered: int  654 670 1229 1454 1518 1518 1362 891 768 1280 ...
 $ cnt       : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
```

Here I removed the instant and dteday variables from the dataset as it
will not Put any value on my project.

```
# Remove variable from the dataset
rent=rent[,-c(1,2)]
```

Now I have only 14 variables.

Out of these 14 variables some are in factor format. So, I convert
these variables to factor.

```
#Convert to factor
colnames(rent)
for (i in c(1,2,3,4,5,6,7))
{rent[,i]=as.factor(rent[,i])
}
```

Now the new structure of the dataset are as follows;

```
str(rent)
'data.frame':	731 obs. of  14 variables:
 $ season    : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
 $ yr        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ mnth      : Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ weekday   : Factor w/ 7 levels "0","1","2","3",..: 7 1 2 3 4 5 6 7 1 2 ...
 $ workingday: Factor w/ 2 levels "0","1": 1 1 2 2 2 2 2 1 1 2 ...
 $ weathersit: Factor w/ 3 levels "1","2","3": 2 2 1 1 1 1 2 2 1 1 ...
 $ temp      : num  0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num  0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num  0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num  0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : int  331 131 120 108 82 88 148 68 54 41 ...
 $ registered: int  654 670 1229 1454 1518 1518 1362 891 768 1280 ...
 $ cnt       : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
```

Chapter 2

Methodology

2.1 Pre Processing

Data preprocessing is a technique that involves transforming raw data into an understandable format. Real-world data is often **incomplete**, **inconsistent**, and/or **lacking** in certain **behaviours or trends**, and is likely to contain many **errors**. Data preprocessing is a proven method of resolving such issues. Data preprocessing **prepares raw data** for **further processing**.

2.1.1 Missing Value Analysis

R code for Missing value analysis:

```
sum(is.na(rent))
[1] 0
```

I Checked for the missing value in bike renting dataset and found there is no missing value present in the dataset as shown above.

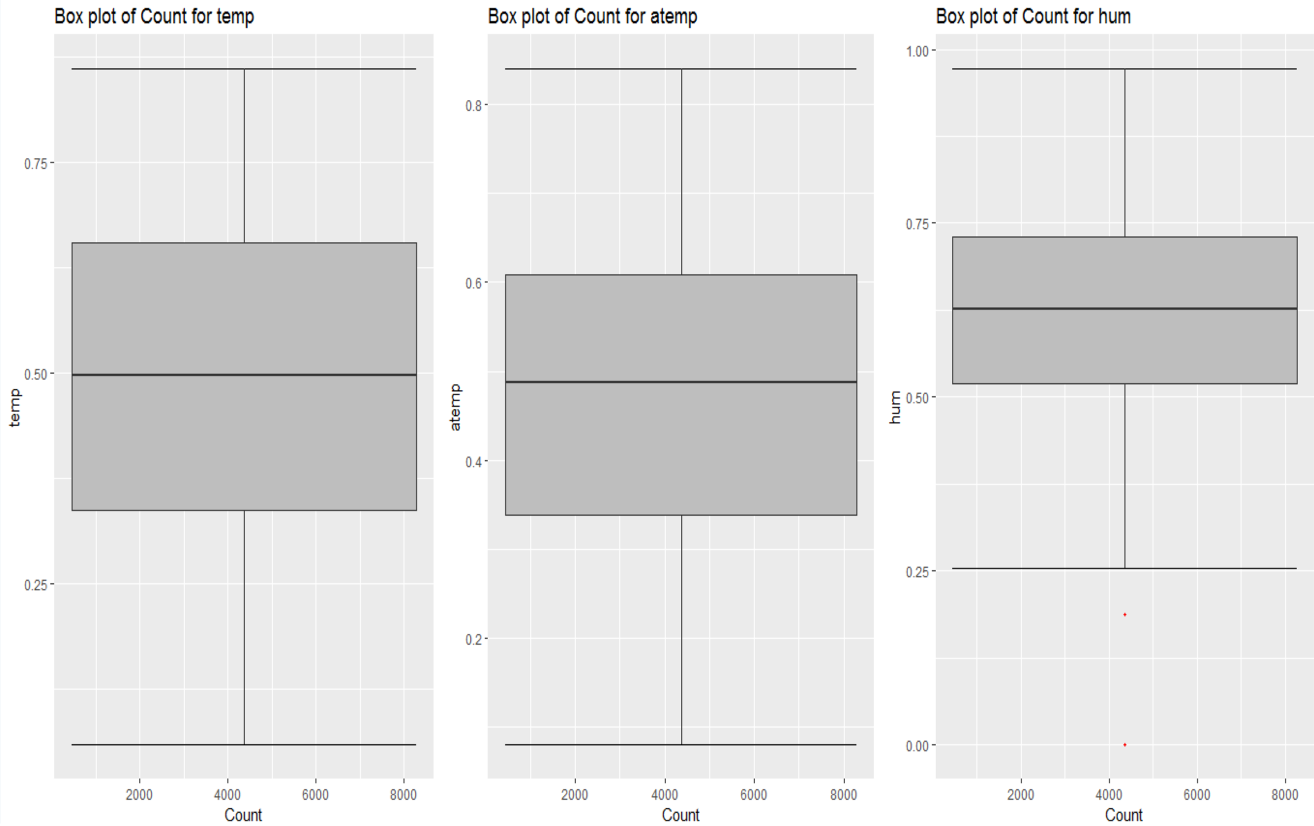
2.1.2 Outlier Analysis

Applied the box plot in all the numeric variables of the dataset and found variables like; humidity, wind speed, casual is having outliers which will affect the dataset. So,I removed all the outliers from the variables.

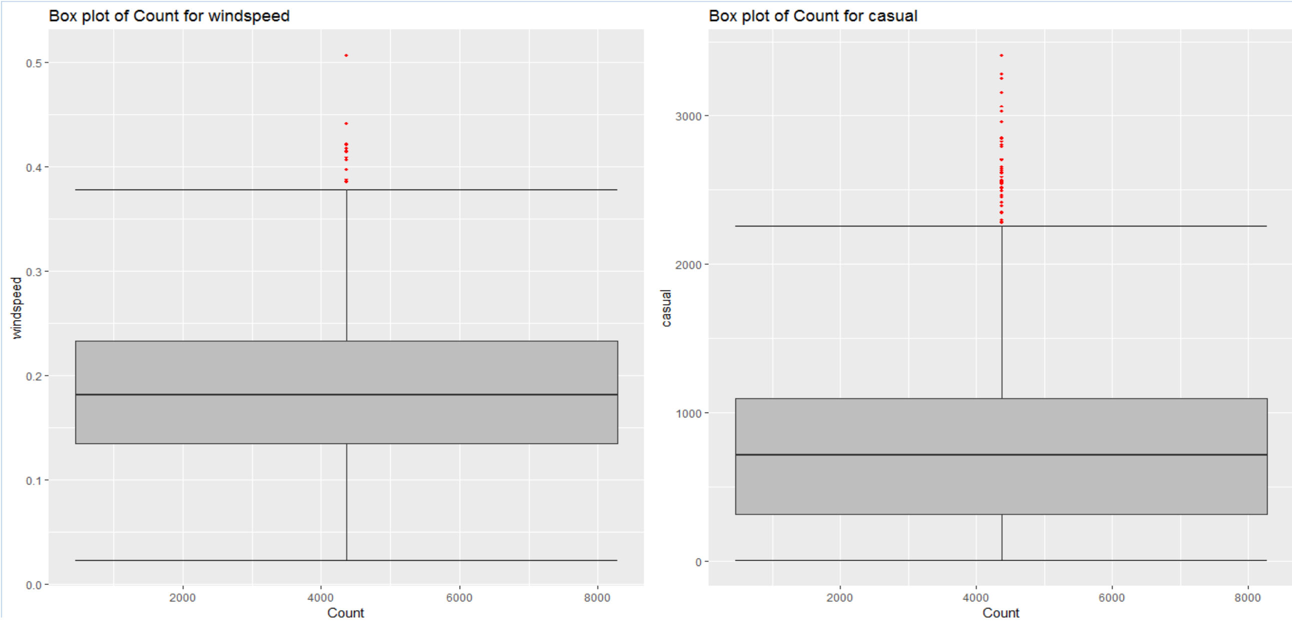
R code for removal of outlier :

```
for(i in cnames)
{
 print(i)
 val=rent[,i][rent[,i]%in% boxplot.stats(rent[,i])$out]
 print(length(val))
 rent=rent[which(!rent[,i]%in%val),]
}
```

Box plot for outliers

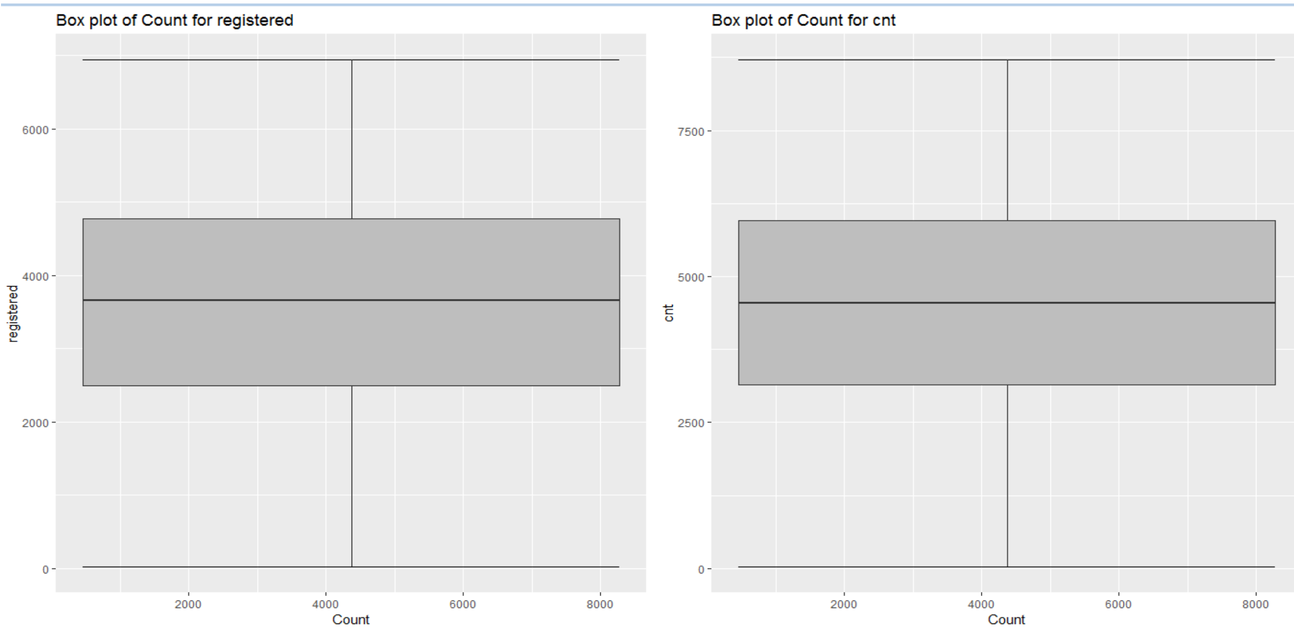


(Fig 1)



Box plot of Count for windspeed
windspeed
0.0
0.1
0.2
0.3
0.4
0.5
2000
4000
6000
8000
Count
Box plot of Count for casual
casual
0
1000
2000
3000
2000
4000
6000
8000
Count

(Fig 2)



Box plot of Count for registered
registered
0
2000
4000
6000
2000
4000
6000
8000
Count
Box plot of Count for cnt
cnt
0
2500
5000
7500
2000
4000
6000
8000
Count

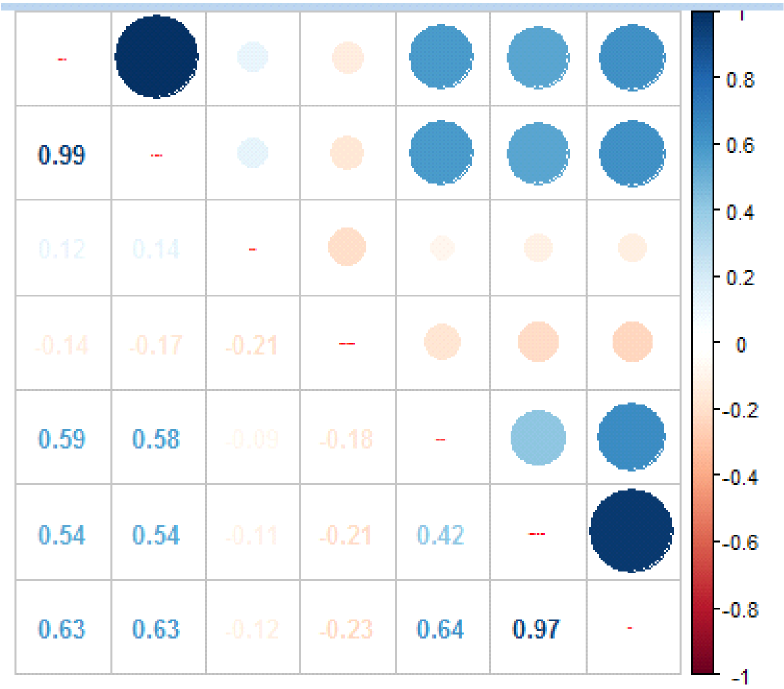
(Fig 3)

2.1.3 Correlation Check

- In feature selection method I checked the association between two variables.
- For continuous variables I used correlation analysis which tells the direction and strength of the linear relationship between two quantitative variables.
- I found temp and atemp are highly correlated with each other and registered and cnt is highly correlated.
- So I remove atemp and registered from the dataset.

*As cnt= registered + casual
So, removed casual from the dataset because it has no use as I am considering “cnt” as my target variable..

Correlation analysis plot:



#Removal of atemp, casual & registered variables
rent=rent[,-c(9,12,13)]

2.2 Modeling

2.2.1 Model Selection

I am selecting the Regression model for Bike Renting dataset to predict the count on daily based on the environmental and seasonal settings.
For Regression applied linear Regression and Random Forest . But linear regression model gives the higher accuracy than others. So, I choose linear regression model for the dataset.

2.2.2 Model Evaluation

Linear Regression

Split the dataset into train and test with 70% and 30% respectively.

Then applied the linear regression in train data and the results of the model are mentioned below.

R Code and the output:

```
index=sample(nrow(rent),0.7*nrow(rent))

train=rent[index,]

test=rent[-index,]

lm=lm(cnt~.,data=train)

summary(lm)
```

Output:

```
> lm=lm(cnt~.,data=train)
> summary(lm)

Call:
lm(formula = cnt ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max 
-3932.4  -316.9    72.4   447.6  1932.7 

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1482.73     266.08   5.572 4.36e-08 ***
season2       832.95     221.31   3.764 0.000190 ***
season3       758.62     246.97   3.072 0.002259 ** 
season4      1265.24     202.43   6.250 9.60e-10 ***
yr1          1911.52      65.91  29.000  < 2e-16 ***
mnth2         169.94     155.61   1.092 0.275392    
mnth3         485.96     188.26   2.581 0.010160 *  
mnth4         447.02     290.60   1.538 0.124696    
mnth5         781.73     313.71   2.492 0.013068 *  
mnth6         497.86     333.46   1.493 0.136144    
mnth7         185.56     361.20   0.514 0.607702    
mnth8         576.15     347.86   1.656 0.098370 .  
mnth9        1029.64     300.75   3.424 0.000675 ***
mnth10        817.85     268.01   3.052 0.002412 ** 
mnth11         94.23     256.89   0.367 0.713921    
mnth12        189.83     203.60   0.932 0.351656    
holiday1     -631.79     235.46  -2.683 0.007564 ** 
weekday1      341.21     125.83   2.712 0.006955 ** 
weekday2      396.42     120.35   3.294 0.001067 ** 
weekday3      440.24     118.53   3.714 0.000230 ***
weekday4      514.10     123.88   4.150 3.99e-05 ***
weekday5      559.38     118.60   4.717 3.22e-06 ***
weekday6      341.48     125.14   2.729 0.006610 ** 
workingday1       NA         NA      NA       NA    

weathersit2  -498.11      89.18  -5.586 4.06e-08 ***
weathersit3 -1694.57     218.09  -7.770 5.48e-14 ***
temp         4336.67     464.92   9.328  < 2e-16 ***
hum         -1392.31     348.47  -3.995 7.55e-05 ***
windspeed   -2712.45     500.40  -5.421 9.76e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 693.8 on 445 degrees of freedom
Multiple R-squared:  0.8689,	Adjusted R-squared:  0.8609 
F-statistic: 109.2 on 27 and 445 DF,  p-value: < 2.2e-16
```

Linear Regression in python

Python code and the Output

```
from sklearn.cross_validation import train_test_split
train,test=train_test_split(rent,test_size=0.2)
import statsmodels.api as sm
lm=sm.OLS(train.iloc[:,10],train.iloc[:,0:10]).fit()
lm.summary()
```

Output

OLS Regression Results

Dep. Variable:	cnt	R-squared:	0.970
Model:	OLS	Adj. R-squared:	0.969
Method:	Least Squares	F-statistic:	1709.
Date:	Thu, 01 Nov 2018	Prob (F-statistic):	0.00
Time:	14:59:49	Log-Likelihood:	-4390.5
No. Observations:	540	AIC:	8801.
Df Residuals:	530	BIC:	8844.
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
season	561.1586	60.697	9.245	0.000	441.922	680.395
yr	2014.5882	70.993	28.377	0.000	1875.126	2154.050
mnth	-39.7000	19.049	-2.084	0.038	-77.122	-2.278
holiday	-369.7268	234.593	-1.576	0.116	-830.573	91.119
weekday	74.7075	18.685	3.998	0.000	38.002	111.413
workingday	518.5003	83.692	6.195	0.000	354.091	682.909
weathersit	-660.0044	92.291	-7.151	0.000	-841.305	-478.704
temp	5103.7799	207.936	24.545	0.000	4695.300	5512.260
hum	273.9647	291.187	0.941	0.347	-298.058	845.988
windspeed	-577.7733	427.447	-1.352	0.177	-1417.473	261.926

Omnibus:	66.816	Durbin-Watson:	1.946
Prob(Omnibus):	0.000	Jarque-Bera (JB):	106.586
Skew:	-0.802	Prob(JB):	7.16e-24
Kurtosis:	4.471	Cond. No.	103.

Random Forest

Applied random forest with ntree 500 and the results are as follows;

R Code and the output:

library(randomForest)

rf=randomForest(cnt~.,data = train,ntree=500)

predictions=predict(rf,test[,-11])

r2=1-(sum(test$cnt-predictions)^2)/sum((test$cnt-mean(test$cnt))^2)

r2

Output:

```
> rf=randomForest(cnt~.,data = train,ntree=500)
> predictions=predict(rf,test[,-11])
> r2=1-(sum(test$cnt-predictions)^2)/sum((test$cnt-mean(test$cnt))^2)
> r2
[1] 0.8637488
```

Output of Random forest with 100 ntree:

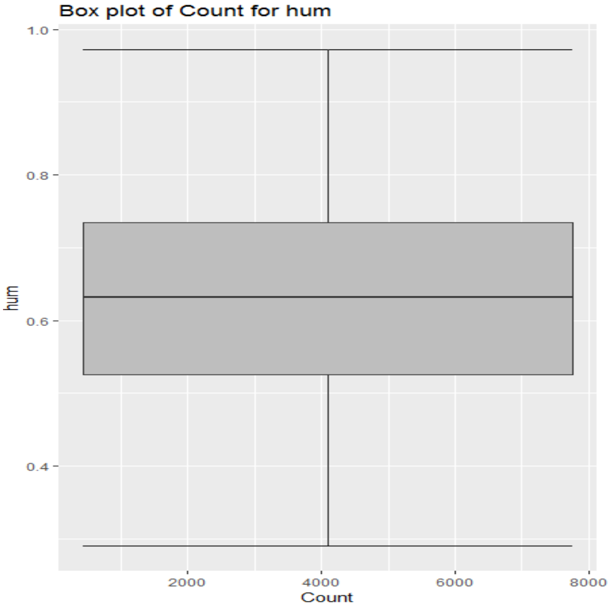
```
> rf=randomForest(cnt~.,data = train,ntree=100)
> predictions=predict(rf,test[,-11])
> r2=1-(sum(test$cnt-predictions)^2)/sum((test$cnt-mean(test$cnt))^2)
> r2
[1] 0.8099373
```

3. Conclusion:

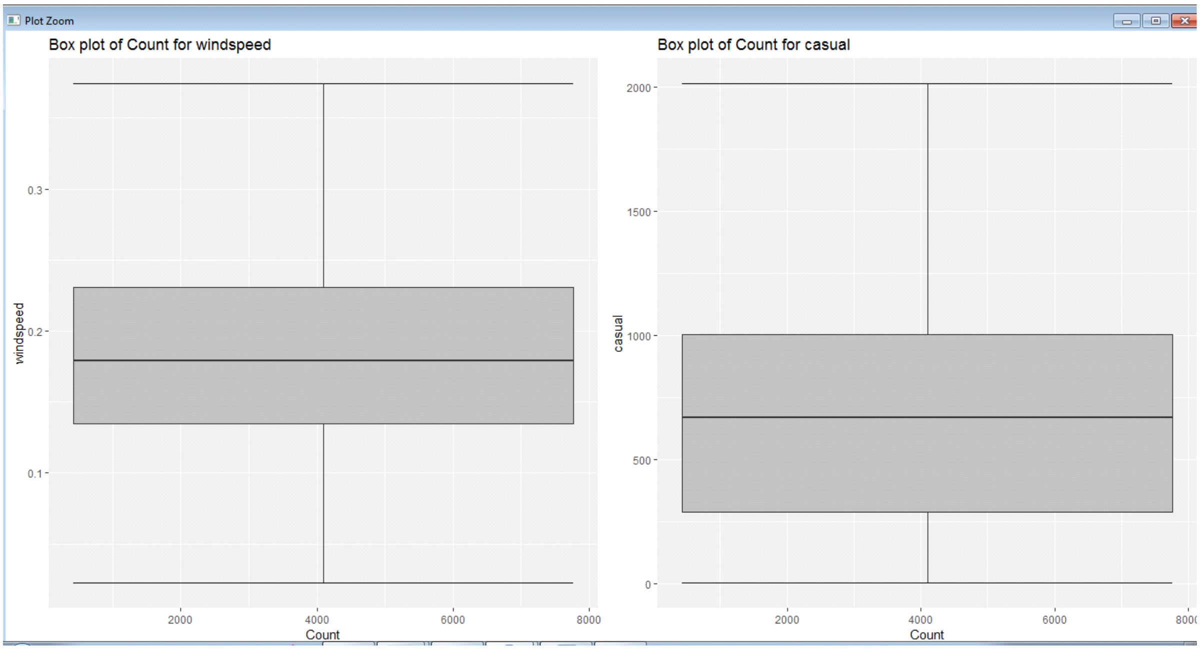
We have developed liner regression that can be applied on the bike sharing data for predicting the bike business. From the model we can see that weather playing an important role in the bike business. In sunny weather we can see that there is high usage of bike compared to rainy weather. We have tried different models like random forest, decision tree. Random forest and decision tree not gave a good result. So we choose linear regression over random forest and decision tree. This model will help the company based on the condition.

Appendix 1– Extra Figures

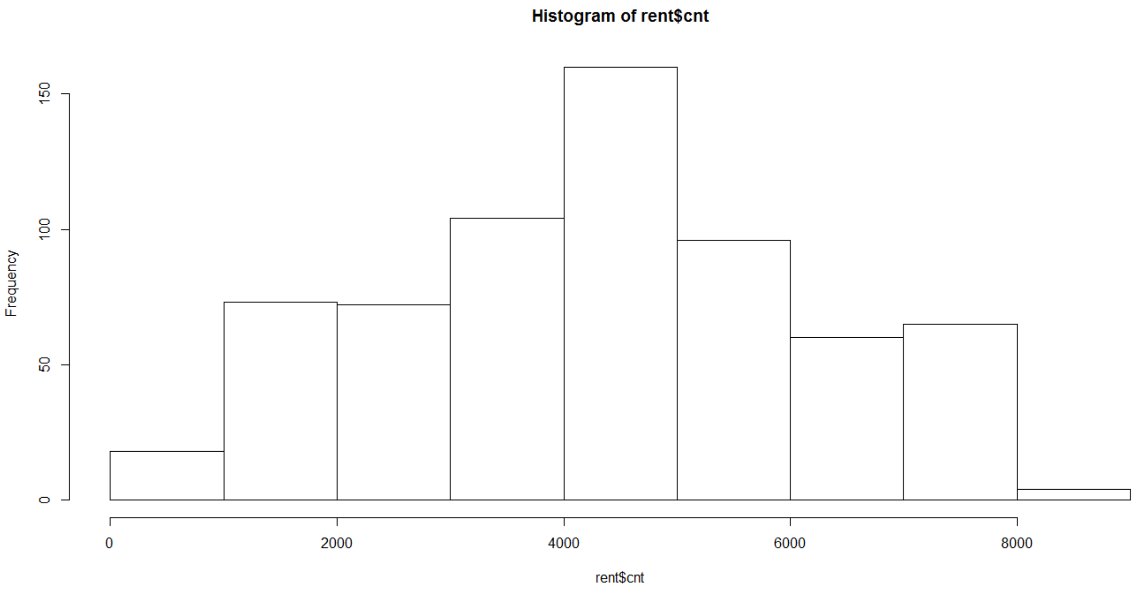
Box plot for humidity without outliers



Box plot for windspeed & casual without outliers



Distribution of Target variable



Appendix 2 – R Code

```
setwd("G:/bike rental project/R")

rent=read.csv("day.csv",header=T)

rent=rent[,-c(1,2)]

for (i in c(1,2,3,4,5,6,7))

{rent[,i]=as.factor(rent[,i])

}

sum(is.na(rent))

numeric_index=sapply(rent,is.numeric)

numeric_data=rent[,numeric_index]

cnames=colnames(numeric_data)

cnames

library(ggplot2)

for(i in 1:length(cnames))

{assign(paste0("gn", i),ggplot(aes_string(y = (cnames[i]),x = "cnt")

                              ,data = subset(rent))+ stat_boxplot(geom = "errorbar", width =
0.5)+

        geom_boxplot(outlier.colour="RED",

                     fill ="grey", outlier.shape = 18, outlier.size = 1, notch =
FALSE)+theme(legend.position = 'bottom')

       +        labs(y = cnames[i],x = 'Count')+

        ggtitle(paste("Box plot of Count for", cnames[i])))}

library(gridExtra)

gridExtra::grid.arrange(gn1,gn2,gn3,ncol = 3)

gridExtra::grid.arrange(gn4,gn5,ncol = 2)

gridExtra::grid.arrange(gn6,gn7,ncol = 2)

for(i in cnames)

{
```

```
  print(i)
  val=rent[,i][rent[,i]%in% boxplot.stats(rent[,i])$out]
  print(length(val))
  rent=rent[which(!rent[,i]%in%val),]
}
library(corrplot)
correlation=cor(rent[,numeric_index])
correlation
corrplot.mixed(correlation,tl.offset=0.01,tl.cex=0.01)
findCorrelation(correlation,cutoff = 0.6)
rent=rent[,-c(9,12,13)]
index=sample(nrow(rent),0.7*nrow(rent))
train=rent[index,]
test=rent[-index,]
lm=lm(cnt~.,data=train)
summary(lm)
```

Appendix 3 – Python Code

```
import os
import pandas as pd
import numpy as np
import matplotlib as mlt
import matplotlib.pyplot as plt
import seaborn as sn
os.chdir("G:/bike rental project")
rent=pd.read_csv("day.csv")
rent=rent.drop(['instant','dteday'],axis=1)
rent.isnull().sum()
cnames=['temp','atemp','hum','windspeed','casual','registered','cnt']
for i in cnames:
    q75,q25=np.percentile(rent.loc[:,i],[75,25])
    iqr=q75-q25
    min=q25-(iqr*1.5)
    max=q75+(iqr*1.5)
    rent=rent.drop(rent[rent.loc[:,i]<min].index)
    rent=rent.drop(rent[rent.loc[:,i]>max].index)
rent_corr=rent.loc[:,cnames]
f,ax=plt.subplots(figsize=(7,5))
corr=rent_corr.corr()
sn.heatmap(corr,mask=np.zeros_like(corr,dtype=np.bool),cmap=sn.diverging_palett
e(220,10,as_cmap=True),square=True,ax=ax)
rent=rent.drop(['atemp','casual','registered'],axis=1)
from sklearn.cross_validation import train_test_split
train,test=train_test_split(rent,test_size=0.2)
import statsmodels.api as sm
lm=sm.OLS(train.iloc[:,10],train.iloc[:,0:10]).fit()
lm.summary()
```