Bike\_renting\_project

Sourav Kumar Dokania

23 December 2020

**Contents**

[Chapter 1 3](#_Toc59583009)

[Introduction: 3](#_Toc59583010)

[**1.1 Problem Statement** 3](#_Toc59583011)

[**1.2 Data** 3](#_Toc59583012)

[Chapter 2 4](#_Toc59583013)

[Methodology 4](#_Toc59583014)

[**2.1. Data Pre Processing** 4](#_Toc59583015)

[**2.1.1 Missing value analysis** 5](#_Toc59583016)

[**2.1.2 Outlier analysis** 6](#_Toc59583017)

[**2.1.3 Feature Selection** 8](#_Toc59583018)

[2.2 Modelling 9](#_Toc59583019)

[**2.2.1 Model Selection** 9](#_Toc59583020)

[**2.2.2 Decision Tree Regression** 9](#_Toc59583021)

[**2.2.3 Random Forest Model** 9](#_Toc59583022)

[Chapter 3 11](#_Toc59583023)

[Conclusion 11](#_Toc59583024)

[**3.1 Model Evaluation** 11](#_Toc59583025)

[**3.1.1 Mean absolute error** 11](#_Toc59583026)

[Appendix A Extra figures 12](#_Toc59583027)

[**Box Plots for Outlier Analysis** 12](#_Toc59583028)

[**Correlation Matrix for Feature Selection** 14](#_Toc59583029)

[**Normality Plot** 15](#_Toc59583030)

[Appendix B R code 16](#_Toc59583031)

# Chapter 1

# Introduction:

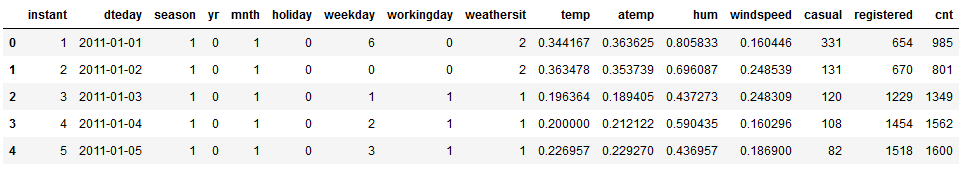
**1.1 Problem Statement**

The project requires predicting bike rent numbers based on the environmental and seasonal setting. Each year, season and varying work days, weekday’s people rent bikes to get to work and to their regular tasks. It makes it easier for the people to have transport that they can own temporarily as well as it is offered at low cost. This helps to keep the business running. The data set contains all environmental setting over the years and requires us to predict bike rental count based on that.

**1.2 Data**

Our task is to scrutinize the data based on the setting of environment, and seasons and apply different regression models to predict the rental count.

Table 1.1 Day sample first few data



There are total of 16 variables, out of which 15 are independent variable and one dependent variable. Variable: “cnt” is out target variable here.

# check all column names

bike\_day.columns

Index(['instant', 'dteday', 'season', 'yr', 'mnth', 'holiday', 'weekday',

'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed',

'casual', 'registered', 'cnt'],

dtype='object')

# Chapter 2

# Methodology

**2.1. Data Pre Processing**

Any data that has to be modelled or analysed has to go through pre-processing techniques. This is the basic requirement as data may come in many forms, and just by looking at data we can miss a lot of information that can be useful. Pre-processing techniques involves checking all the parameters, their data types, uniqueness, dimensions, variability, normalization. Such techniques are useful to get an adequate idea of all the variables and how they may or may not be related to each other. Values ranges can be converted according to the desired problem statement.

# check the target or dependent variable

bike\_day["cnt"].describe()

#type(bike\_day["cnt"]), variable : cnt is float

count 731.000000

mean 4504.348837

std 1937.211452

min 22.000000

25% 3152.000000

50% 4548.000000

75% 5956.000000

max 8714.000000

Name: cnt, dtype: float64

Describe is a method that shows all the statistic description of a particular variable in a data and it is very important to know the formation.

We can also check if some variables have categorical values by checking the uniqueness of the values present. For ex:

bike\_day["weathersit"].unique()

array([2, 1, 3], dtype=int64)

**2.1.1 Missing value analysis**

Data can be huge and it is highly possible that some of the data points in variables may be not present or may be lost. Sometime the data we deal with has a lot of variables and thousands or lakhs of observations. It becomes nearly impossible for us to glance or count the number of missing values.

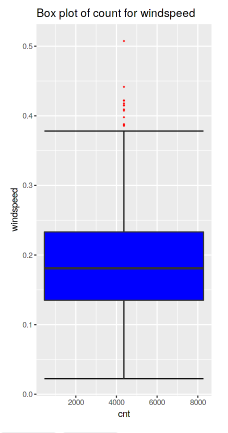
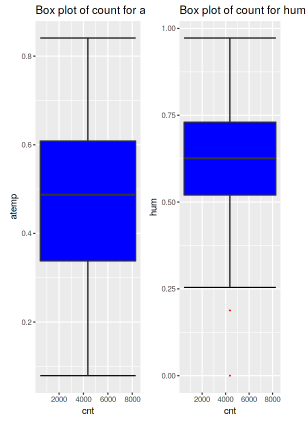
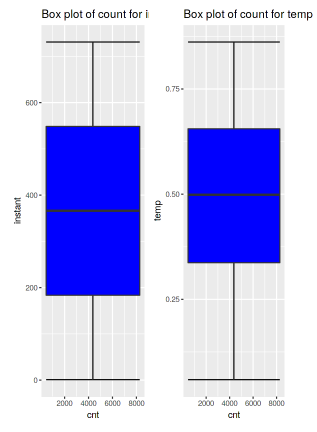
missing\_value = pd.DataFrame(bike\_day.isnull().sum())

This line of code helps us in getting all the missing values and also tells us how many counts are there in the variable.

| **0** |
| --- |
| **instant** | 0 |
| **dteday** | 0 |
| **season** | 0 |
| **yr** | 0 |
| **mnth** | 0 |
| **holiday** | 0 |
| **weekday** | 0 |
| **workingday** | 0 |
| **weathersit** | 0 |
| **temp** | 0 |
| **atemp** | 0 |
| **hum** | 0 |
| **windspeed** | 0 |
| **casual** | 0 |
| **registered** | 0 |
| **cnt** | 0 |

**2.1.2 Outlier analysis**

There are some data whose mean, median, std all are deviated from the actual plots. This happens mostly in case of continuous variables as outliers can only be applied to those. Some values are extreme values and fall outside the range of most of the values. Outliers can be analysed using the box plot. Box plots are a grat way to look at the values and their distributions. It helps to examine the variable its 25th, 50th(median), 75th percentile, inter quartile range.



**2.1.3 Feature Selection**

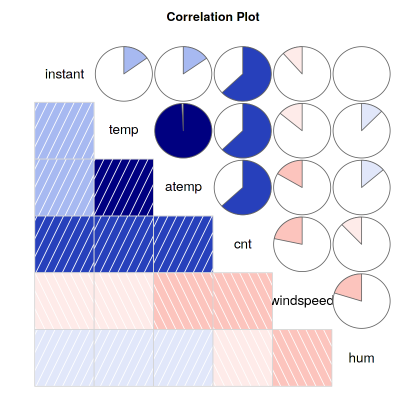
There may be multiple variables that may not be important for the modelling of the data set. Some features or variables may carry the same information or are highly correlated to each other. Numeric and categorical values have two different ways of selecting features that are unique, independent and are helpful for predicting and choosing best model. Here we have used correlation matrix to determine which variables are correlated. Correlation matrix can only be applied to numeric data or continuous variables. For categorical variable we choose chi-square test. Here the target variable is numeric so correlation matrix has been used.

# correlation matrix to determine relationship between continuous variables.

library(corrgram)

corrgram(bike\_day[,numeric\_index], order = F,

upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")



It can be clearly observed that Temp and atemp variable are highly correlated so we can ignore atemp variable for further analysis of our model.

# 2.2 Modelling

**2.2.1 Model Selection**

In the rudimentary stage it is not possible to know which model to will be best suited to our requirement during pre-processing. Therefore we cannot rely on one single model for predicting our data. Model selection are based on factors such as dependent variable type, error metrics etc.

The target variable in this data set is of numeric type and thus we will use regression analysis. Has the target variable been categorical or nominal we had to use classification analysis for the model.

Here for this data set we have used three models to predict the outcome of our target variables namely:

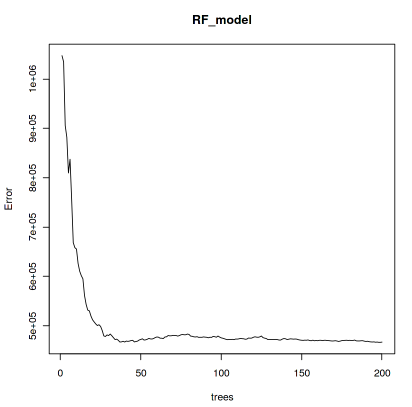
1. Decision Tree regression
2. Random Forest regression
3. Linear regression.

**2.2.2 Decision Tree Regression**

This regression a model is described by its name itself, it is like a tree where each time the dataset is divided into smaller subsets, and continually building up trees connecting to the previous one. The final tree consists of leaf and nodes.

**2.2.3 Random Forest Model**

Random forest simply put takes multiple decision tree to finalize the output rather than one decision tree at a time. These are much accurate than one decision tree and can work upon values in thousands.



# Chapter 3

# Conclusion

**3.1 Model Evaluation**

After all the techniques we have used we have got three models that fits the purpose. But one of these is the best suited for our analysis. We have to evaluate and compare the models using some of the criteria such as:

1. Predictive performance
2. Interpretability
3. Computational Efficiency.

In this case we are going to use predictive performance to evaluate and compare all three models.

We will be using Mean absolute error to determine best possible model for our dataset.

**3.1.1 Mean absolute error**

#Calculate MAPE

def MAPE(y\_true, y\_pred):

mape = np.mean(np.abs((y\_true - y\_pred) / y\_true))\*100

return mape

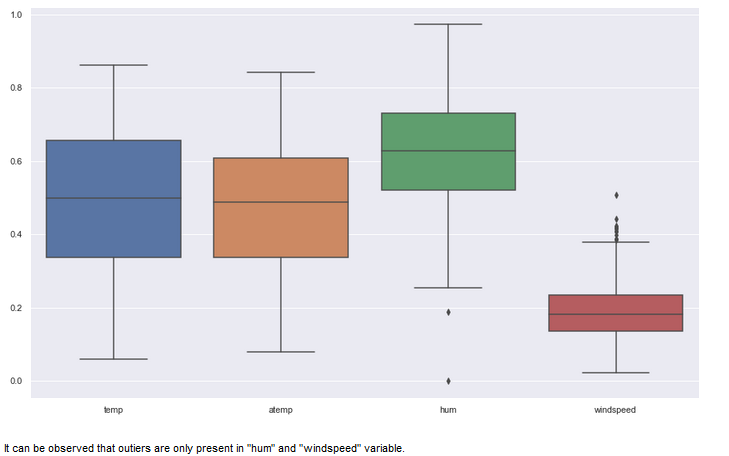
MAPE(test\_target\_feature, prediction\_dt)

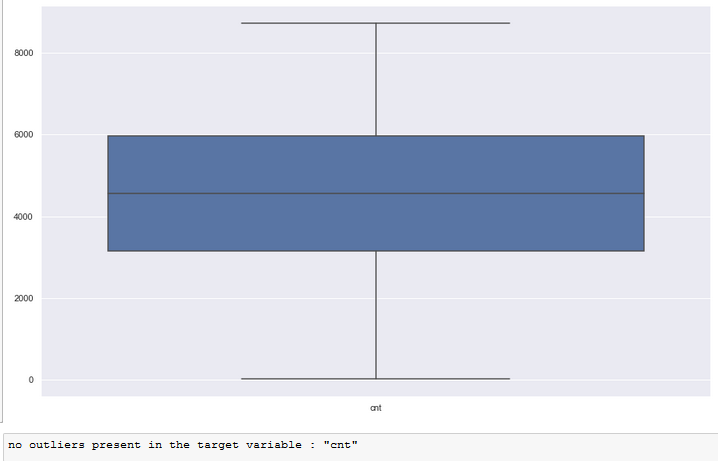
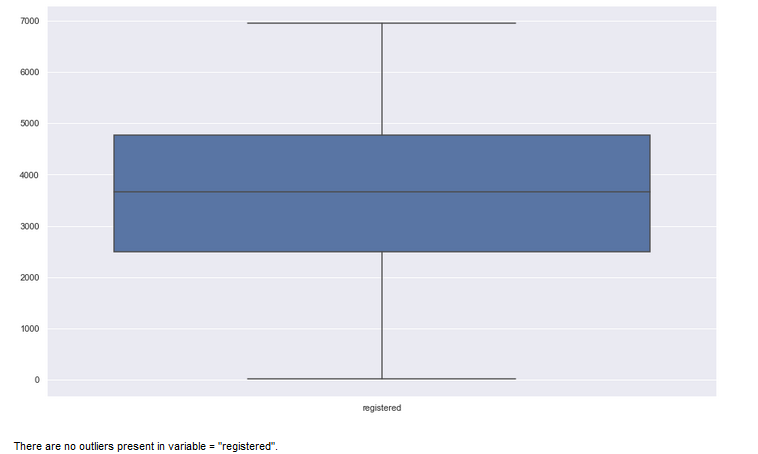
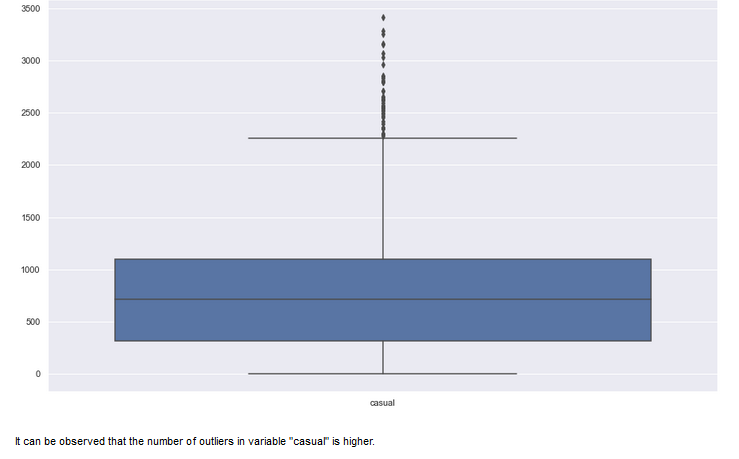
Above mention is the code that is used to determine the Mean absolute error. The model which has the lowest mean absolute error has been chosen for the data. W

We have chosen Random Forest as our model.

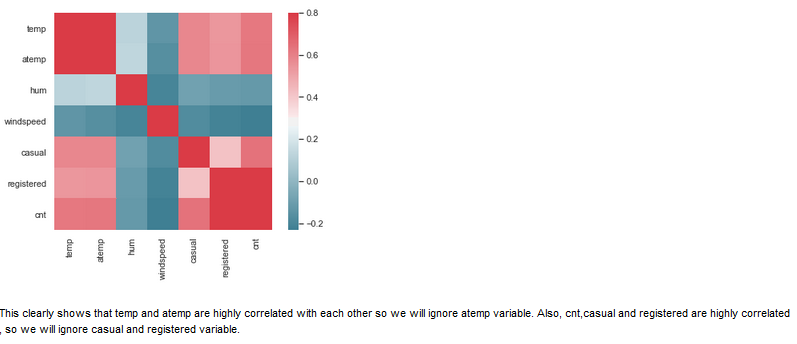
# Appendix A Extra figures

**Box Plots for Outlier Analysis**

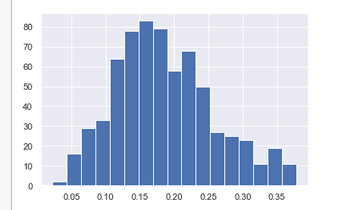
****



**Correlation Matrix for Feature Selection**



**Normality Plot**



# Appendix B R code

# %% [markdown]

# \*\*Project title\*\* :- \*\*\*\*Bike Renting using R\*\*\*\*

# %% [markdown]

# \*\*Problem statement :-\*\*

#

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

# environmental and seasonal settings.

# %% [markdown]

# \*\*Exploratery Data Analysis\*\*

# %% [markdown]

# \*\*Importing the metapackage library\*\*

# %% [code] {"\_execution\_state":"idle"}

#import tidyverse metapackage library

install.packages(c("dplyr", "plyr", "reshape", "ggplot2", "data.table"))

install.packages("dmm")

library(tidyverse)

# %% [markdown]

# \*\*Importing the Data\*\*

# %% [code]

#Importing the csv file

bike\_day <-read.csv("f://day.csv")

head(bike\_day,5)

# %% [code]

tail(bike\_day,5)

# %% [markdown]

# Remove the casual and registered variable due to total\_count is combined of both.

#

# %% [code]

#Create new dataset excluding casual and registered variables

bike\_day <- subset(bike\_day,select=-c(casual,registered))

head(bike\_day,5)

# %% [markdown]

# \*\*Dimension of Dataset\*\*

# %% [code]

#Dimension of dataset

dim(bike\_day)

# %% [markdown]

# \*\*Summary of the Dataset\*\*

# %% [code]

#Summary of the dataset

summary(bike\_day)

# %% [markdown]

# \*\*Structure of Dataset\*\*

# %% [code]

#Structure of dataset

str(bike\_day)

# %% [markdown]

# Get column names

# %% [code]

colnames(bike\_day)

# %% [markdown]

# \*\*Typecasting the datetime and numerical attributes\*\*

# %% [code]

#Typecasting the datetime and numerical attributes to category

bike\_day$dteday <- as.Date(bike\_day$dteday)

bike\_day$yr <- as.factor(bike\_day$yr)

bike\_day$mnth <- as.factor(bike\_day$mnth)

bike\_day$season <- as.factor(bike\_day$season)

bike\_day$holiday <- as.factor(bike\_day$holiday)

bike\_day$weekday <- as.factor(bike\_day$weekday)

bike\_day$workingday <- as.factor(bike\_day$workingday)

bike\_day$weathersit <- as.factor(bike\_day$weathersit)

# %% [markdown]

# Check all the data types, if correctly changed

# %% [code]

str(bike\_day)

# %% [markdown]

# \*\*Missing value analysis\*\*

# %% [code]

#Missing values in dataset

miss\_val <- data.frame(apply(bike\_day,2,function(x){sum(is.na(x))}))

names(miss\_val)[1]='missing\_val'

miss\_val

# %% [markdown]

# Outlier Analysis

# Box plot and analysis

# %% [code]

numeric\_index = sapply(bike\_day,is.numeric) #selecting only numeric

numeric\_data = bike\_day[,numeric\_index]

cnames = colnames(numeric\_data)

for (i in 1:length(cnames))

{

assign(paste0("gn",i), ggplot(aes\_string(y = (cnames[i]), x = "cnt"), data = subset(bike\_day))+

stat\_boxplot(geom = "errorbar", width = 0.5) +

geom\_boxplot(outlier.colour="red", fill = "blue" ,outlier.shape=18,

outlier.size=1, notch=FALSE) +

theme(legend.position="bottom")+

labs(y=cnames[i],x="cnt")+

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

}

# %% [code]

cnames

# %% [code]

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

# %% [code]

gridExtra::grid.arrange(gn3,gn4,ncol=3)

# %% [code]

gridExtra::grid.arrange(gn5,ncol=2) # it can be seen that hum and windspeed variable both have outliers.

# %% [code]

#load the DMwR library

library(DMwR)

#create subset for windspeed and humidity variable

wind\_hum <- subset(bike\_day,select = c('windspeed','hum'))

#column names of wind\_hum

cnames <- colnames(wind\_hum)

for(i in cnames){

val = wind\_hum[,i][wind\_hum[,i] %in% boxplot.stats(wind\_hum[,i])$out] #outlier values

wind\_hum[,i][wind\_hum[,i] %in% val]= NA # Replace outliers with NA

}

#Imputating the missing values using mean imputation method

wind\_hum$windspeed[is.na(wind\_hum$windspeed)] <- mean(wind\_hum$windspeed,na.rm=T)

wind\_hum$hum[is.na(wind\_hum$hum)] <- mean(wind\_hum$hum,na.rm=T)

# %% [code]

#Remove the windspeed and hum variable in order to replace imputated data

new\_day <- subset(bike\_day,select = -c(windspeed,hum))

#Combined new\_day and wind\_hum data frames

bike\_day <- cbind(new\_day,wind\_hum)

head(bike\_day,5)

# %% [code]

#Missing values in dataset

miss\_val <- data.frame(apply(bike\_day,2,function(x){sum(is.na(x))}))

names(miss\_val)[1]='missing\_val'

miss\_val

# %% [code]

# correlation matrix to determine relationship between continuous variables.

library(corrgram)

corrgram(bike\_day[,numeric\_index], order = F,

upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")

# %% [markdown]

# It can be observed that temp and atemp are highly correlated so we will ignore atemp variable for further analysis.

# %% [code]

#dimension reduction.

bike\_day = subset(bike\_day,select = -c(atemp))

# %% [code]

dim(bike\_day) # we have dropped a temp variable

# %% [code]

library(rpart)

library(MASS)

#divide the data into train and test data.

train\_index = sample(1:nrow(bike\_day), 0.8 \* nrow(bike\_day))

train = bike\_day[train\_index,]

test = bike\_day[-train\_index,]

# %% [code]

# applying decision tree regression model

fit = rpart(cnt ~ ., data = train, method = "anova")

# %% [code]

prediction\_DT = predict(fit, test[,-11])

# %% [code]

prediction\_DT

# %% [code]

#define mape function : MAPE

mape = function(y, yhat){mean(abs((y-yhat)/y))\*100}

mape(test[,11], prediction\_DT)

# %% [code]

str(bike\_day)

# %% [code]

cnames= c("dteday","season","mnth","weekday","weathersit")

bike\_enco=bike\_day[,cnames]

cnt=data.frame(bike\_day$cnt)

names(cnt)[1]="cnt"

bike\_enco <- fastDummies::dummy\_cols(bike\_enco)

bike\_enco= subset(bike\_enco,select = -c(dteday,season,mnth,weekday,weathersit))

d3 = cbind(bike\_enco,bike\_day)

d3= subset(d3,select = -c(dteday,season,mnth,weekday,weathersit,cnt))

bike\_enco=cbind(d3,cnt)

# %% [code]

#divide encoded data into test sets

#divide the data into train and test data.

encoded\_index = sample(1:nrow(bike\_enco), 0.8 \* nrow(bike\_enco))

train\_enco = bike\_enco[encoded\_index,]

test\_enco = bike\_enco[-encoded\_index,]

dim(test\_enco)

# %% [code]

library(randomForest)

RF\_model = randomForest(cnt ~ ., train, importance = TRUE, ntree = 200)

predictions\_RF = predict(RF\_model, test[,-11])

plot(RF\_model)

# %% [code]

mape(test[,11], predictions\_RF)

# %% [code]

test\_enco

# %% [code]

#linear regression model

library(usdm)

lm\_model = lm(cnt ~ ., data = train\_enco)

summary(lm\_model)

# %% [code]

prediction\_Lr = predict(lm\_model, test\_enco[,-34])

# %% [code]

prediction\_Lr

# %% [code]

mape(test\_enco[,34], prediction\_Lr)

# %% [code]

#from the mape function we can see that, of all the models Random Forest has the least abosulute error. So we choose Random Forest model as the best model for the given data.

test\_input = test$cnt

Bike\_rental\_predict = data.frame(test\_input, predictions\_RF)

write.csv(Bike\_rental\_predict,'Bike\_Renting\_R.CSV',row.names=F)

Bike\_rental\_predict

# %% [code]