BIKE RENTAL

SAMEER PANDEY

APRIL 02, 2019

Contents

No table of contents entries found.

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 Variables

There are 731 Observations and 16 variables in our data in which 13 are independent variables and 3 (Casual, Registered and Count) are dependent variables. Since the type of target variable is continuous, this is a regression problem.

Variable Information:

1. **instant**: Record index
2. **dteday**: Date
3. **season**: Season (1:springer, 2:summer, 3:fall, 4:winter)
4. **yr**: Year (0: 2011, 1:2012)
5. **mnth**: Month (1 to 12)
6. **holiday**: weather day is holiday or not (extracted from Holiday Schedule)
7. **weekday**: Day of the week
8. **workingday**: If day is neither weekend nor holiday is 1, otherwise is 0.
9. **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
10. **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)
11. **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)
12. **hum**: Normalized humidity. The values are divided to 100 (max)
13. **windspeed**: Normalized wind speed. The values are divided to 67 (max)
14. **casual**: count of casual users
15. **registered**: count of registered users
16. **cnt**: count of total rental bikes including both casual and registered

1.3 Sample Data

instant	dteday	season	yr	mnth	holiday	weekday
1	01-01-2011	1	0	1	0	6
2	02-01-2011	1	0	1	0	0
3	03-01-2011	1	0	1	0	1
4	04-01-2011	1	0	1	0	2
5	05-01-2011	1	0	1	0	3
6	06-01-2011	1	0	1	0	4

Table 1.1: Bike Rental Sample Data (Columns: 1-7)

workingday	weathersit	temp	atemp	hum	windspeed
0	2	0.344167	0.363625	0.805833	0.160446
0	2	0.363478	0.353739	0.696087	0.248539
1	1	0.196364	0.189405	0.437273	0.248309
1	1	0.2	0.212122	0.590435	0.160296
1	1	0.226957	0.22927	0.436957	0.1869
1	1	0.204348	0.233209	0.518261	0.0895652

Table 1.2: Bike Rental Sample Data (Columns: 8-13)

casual	registered	cnt
331	654	985
131	670	801
120	1229	1349
108	1454	1562
82	1518	1600
88	1518	1606

Table 1.3: Bike Rental Sample Data (Columns: 14-16)

1.4 Unique count

Below figure shows the unique count of all the variables present in the data.

List of columns and their number of unique values -

```
ID                                 36
Reason for absence                 28
Month of absence                   13
Day of the week                     5
Seasons                             4
Transportation expense             24
Distance from Residence to Work    25
Service time                       18
Age                                22
Work load Average/day              38
```

Hit target	13
Disciplinary failure	2
Education	4
Son	5
Social drinker	2
Social smoker	2
Pet	6
Weight	26
Height	14
Body mass index	17
Absenteeism time in hours	19

Chapter 2: Methodology

2.1 Pre – Processing

A predictive model requires that we look at the data before we start to create a model. However, in data mining, looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is known as Exploratory Data Analysis. In this project we look at the distribution of categorical variables and continuous variables. We also look at the missing values in the data and the outliers present in the data.

Remember the quality of our inputs decide the quality of your output. So, once we have got our business hypothesis ready, it makes sense to spend lot of time and efforts here. Data exploration, cleaning and preparation can take up to 70% of our total project time.

Below are the steps involved to understand, clean and prepare our data for building model:

1. Variable Identification
2. Univariate Analysis
3. Bi-variate Analysis
4. Missing values treatment
5. Outlier treatment
6. Feature Selection
7. Feature scaling

2.2 Variable Identification

From EDA we have concluded that there are 10 continuous variable and 10 categorical variable and one continuous target variable.

Target Variable = Absenteeism.time.in.hours

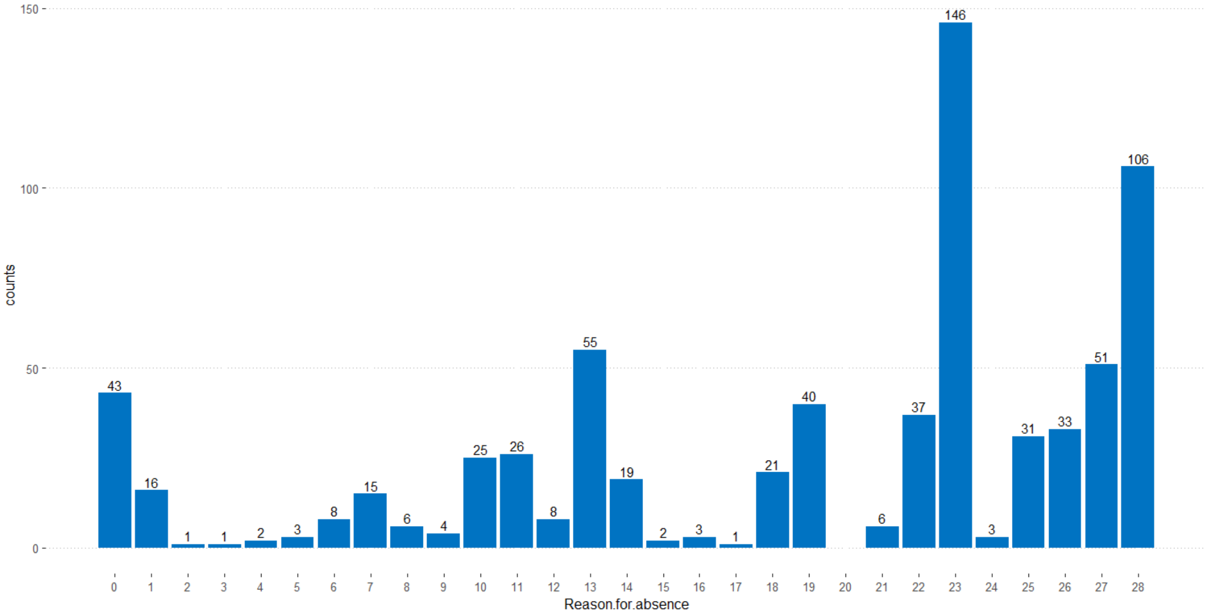
Continuous variables	Categorical variables
Transportation.expense	Reason.for.absence
Distance.from.Residence.to.Work	Month.of.absence
Age	Day.of.the.week
Transportation expense	Seasons
Work.load.Average.day.	Disciplinary.failure
Hit.target	Education
Weight	Son
Height	Social.drinker
Body.mass.index	Social.smoker
Service. Time	Pet

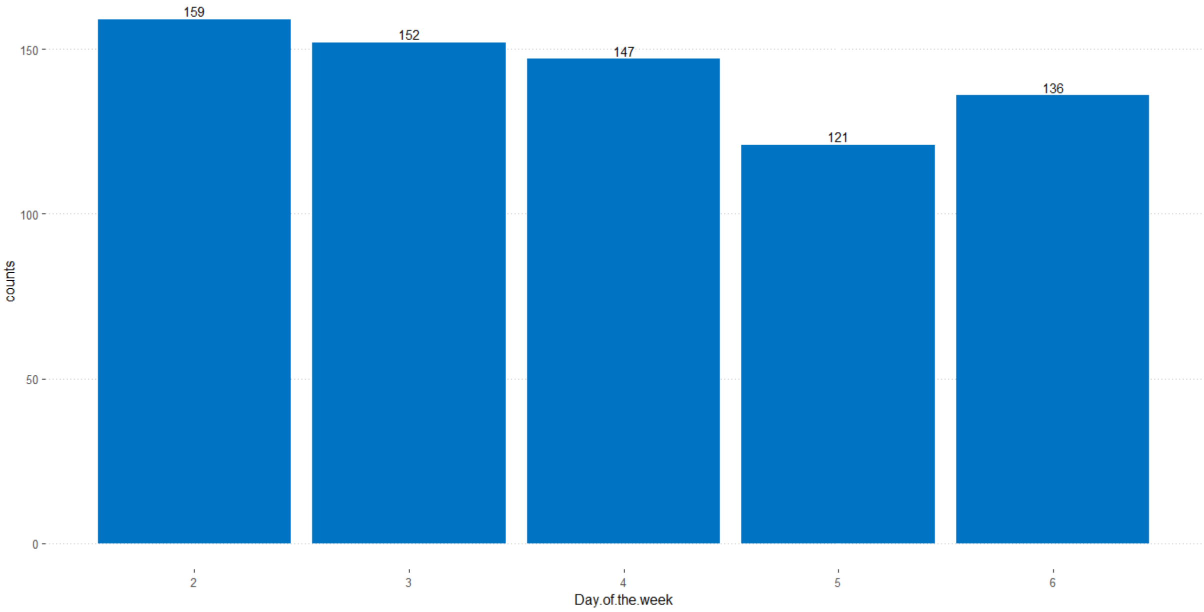
Table 1.4: Employee Absenteeism Variable Category

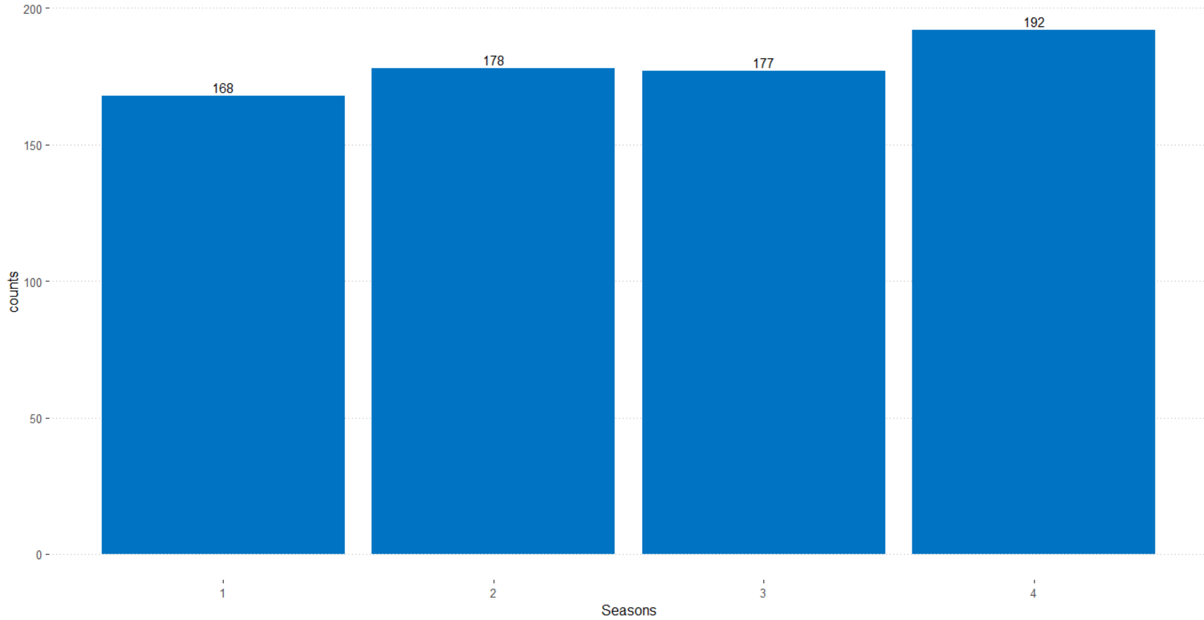
2.3 Univariate Analysis

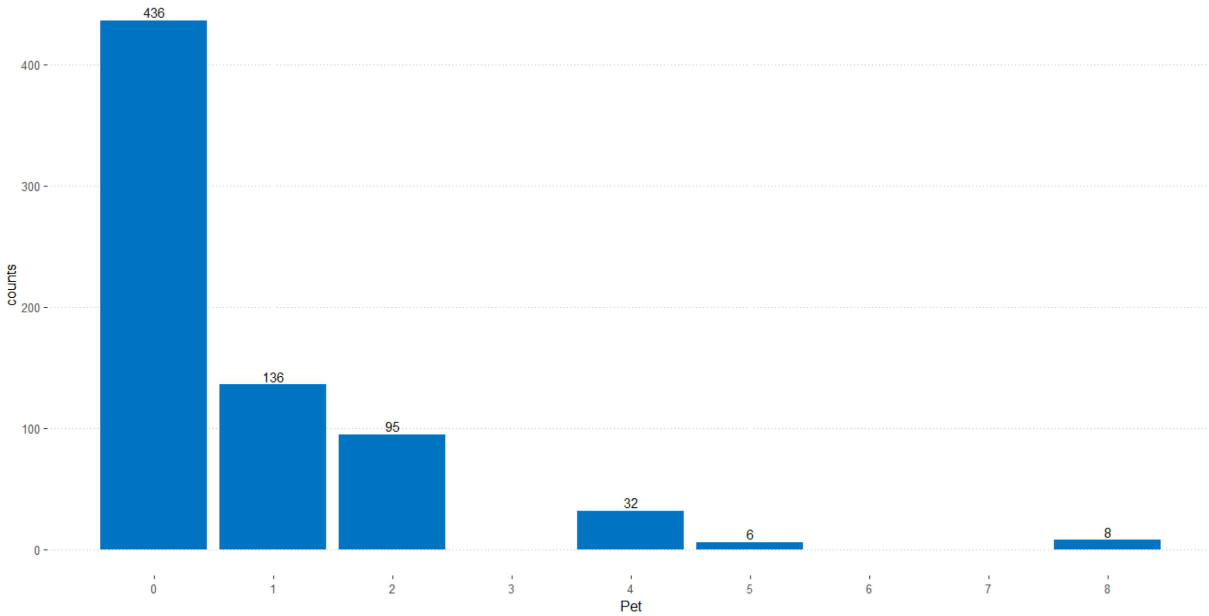
At this stage, we explore variables one by one. Method to perform uni-variate analysis will depend on whether the variable type is categorical or continuous. Let's look at these methods and statistical measures for categorical and continuous variables individually:

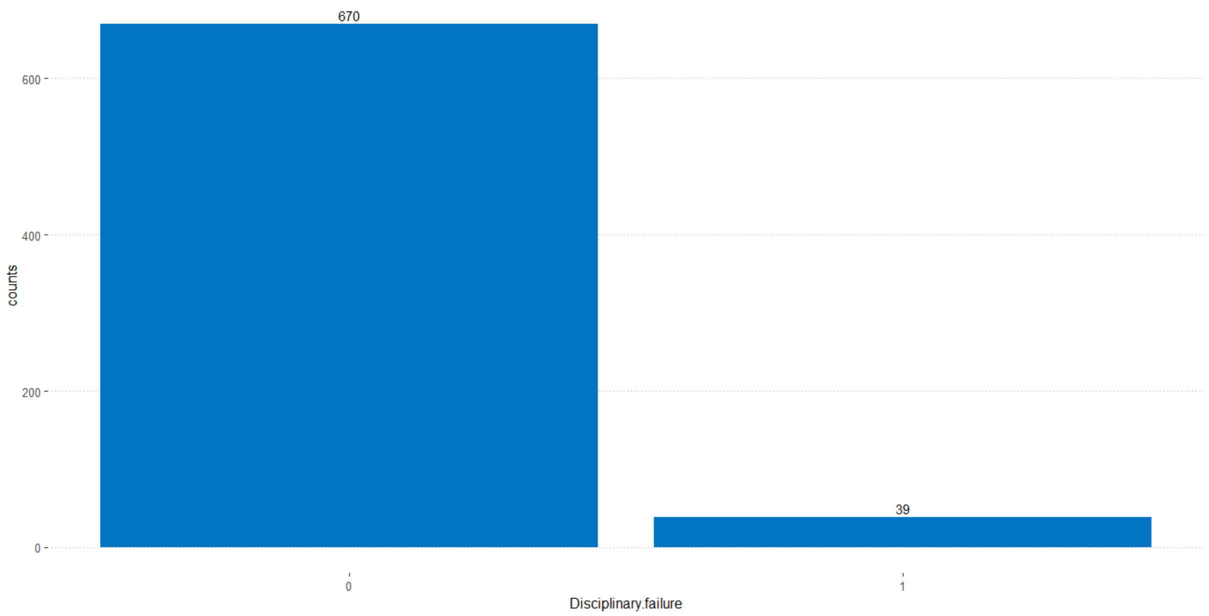
Categorical Variables

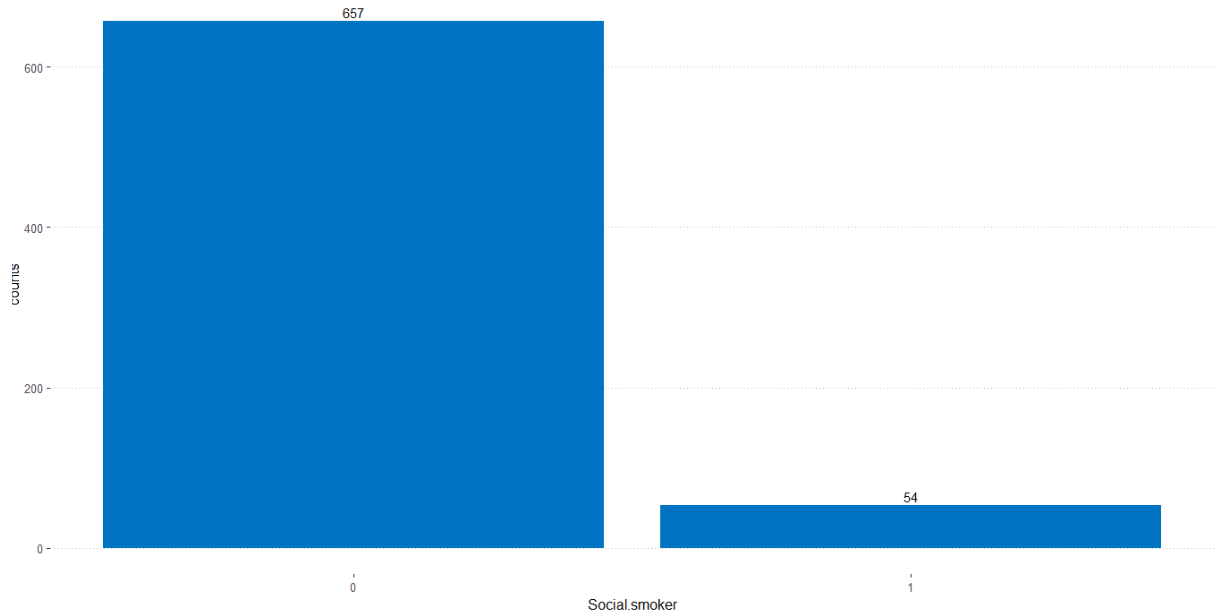


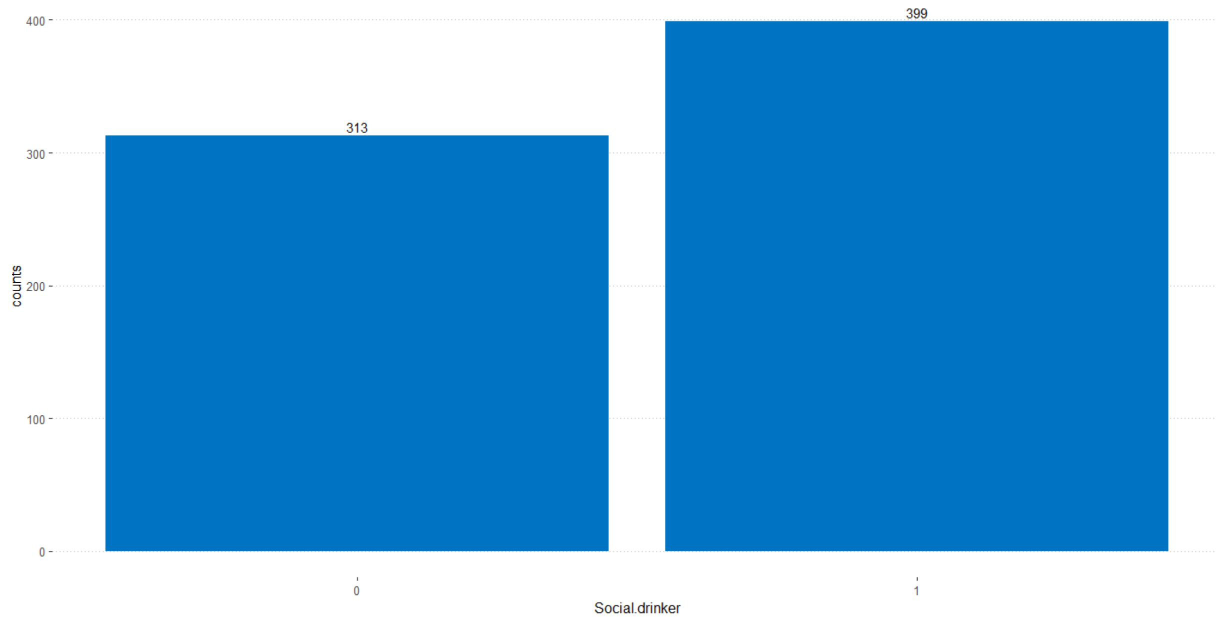


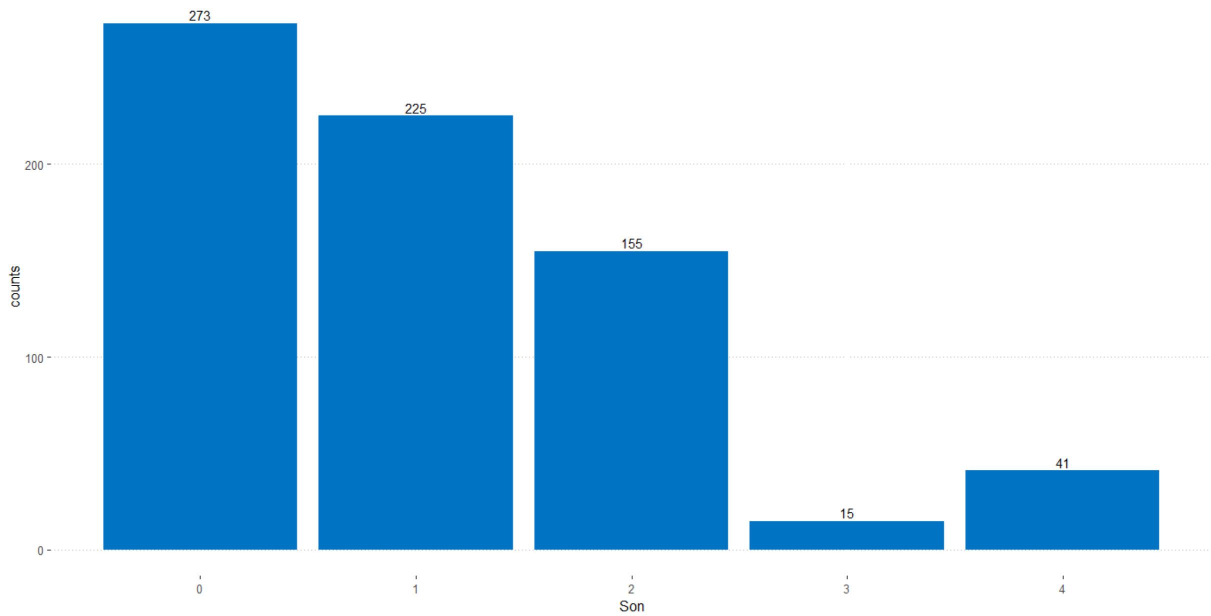

200
150
100
50
0
counts
168
178
177
192
1
2
3
4
Seasons

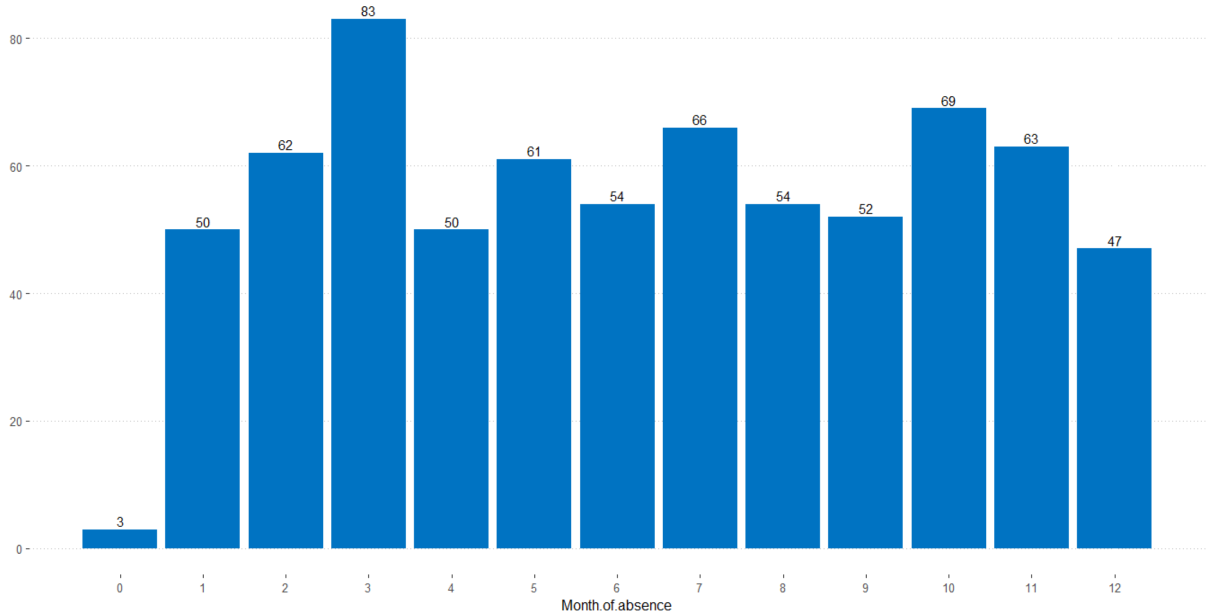

400
300
200
100
0
counts
436
136
95
32
6
8
0
1
2
3
4
5
6
7
8
Pet

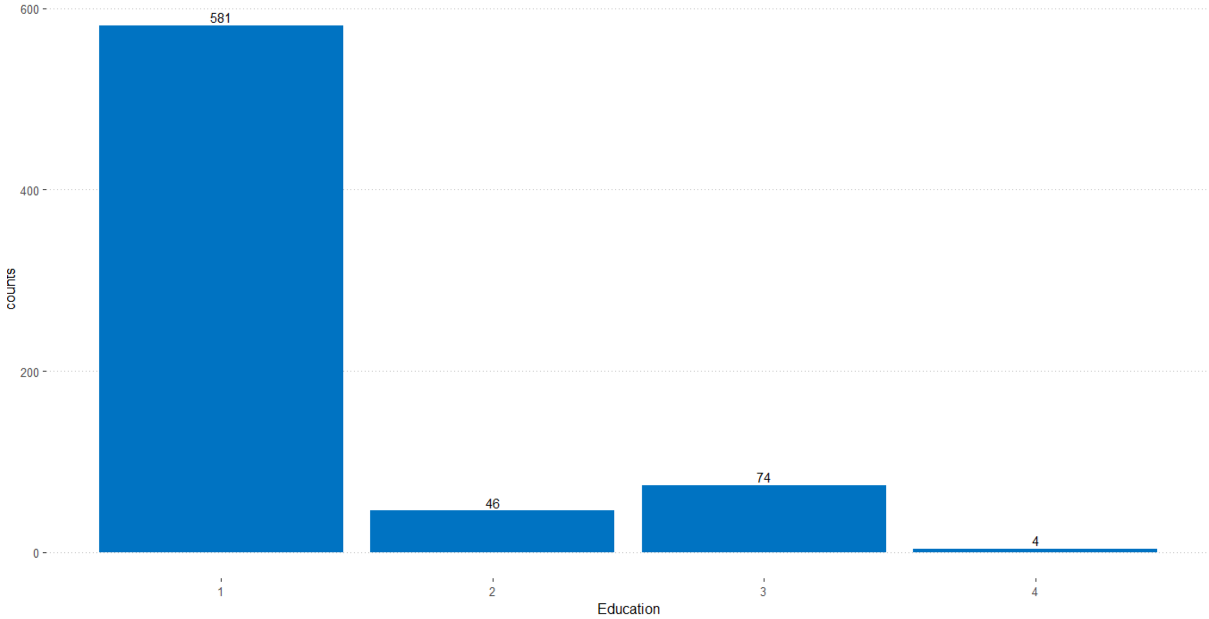

600
400
200
0
counts
670
39
0
1
Disciplinary.failure


657
54
600
400
200
0
counts
0
1
Social.smoker

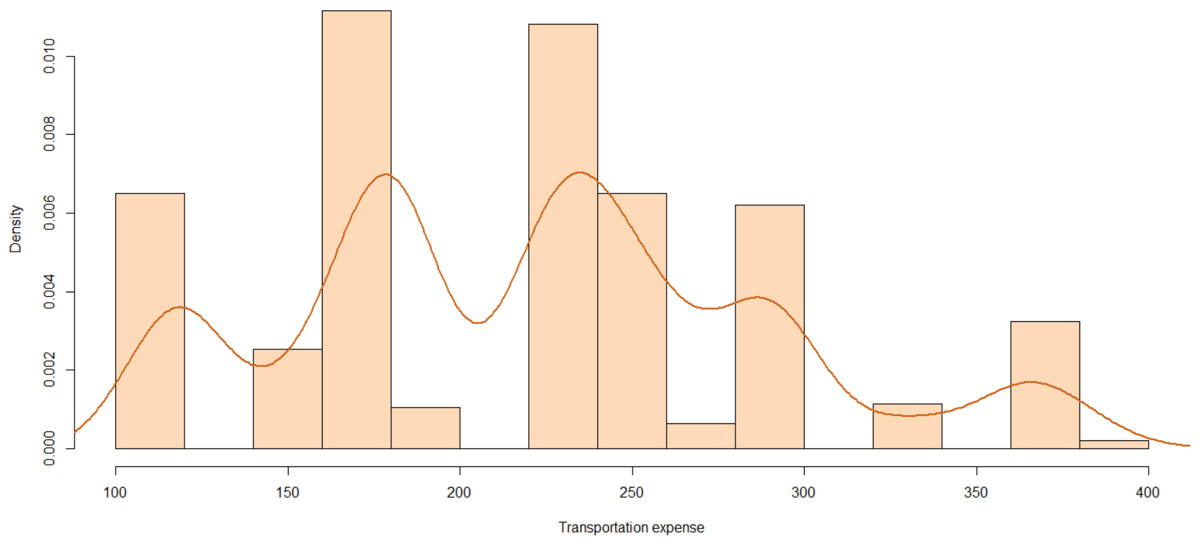

399
313
400
300
200
100
0
counts
0
1
Social.drinker

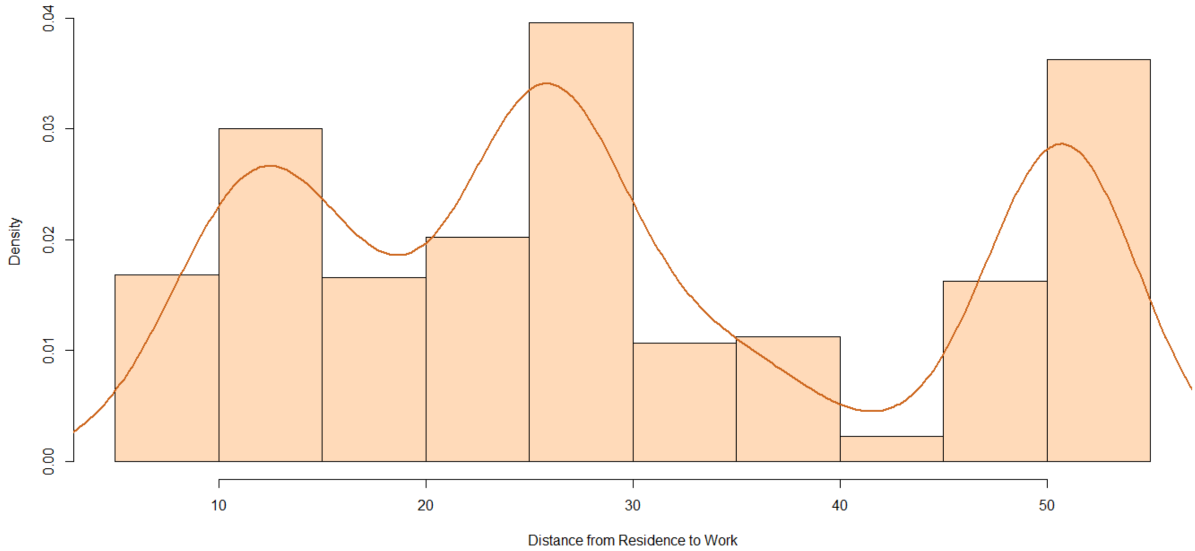

273
225
155
15
41
200
100
0
counts
0
1
2
3
4
Son

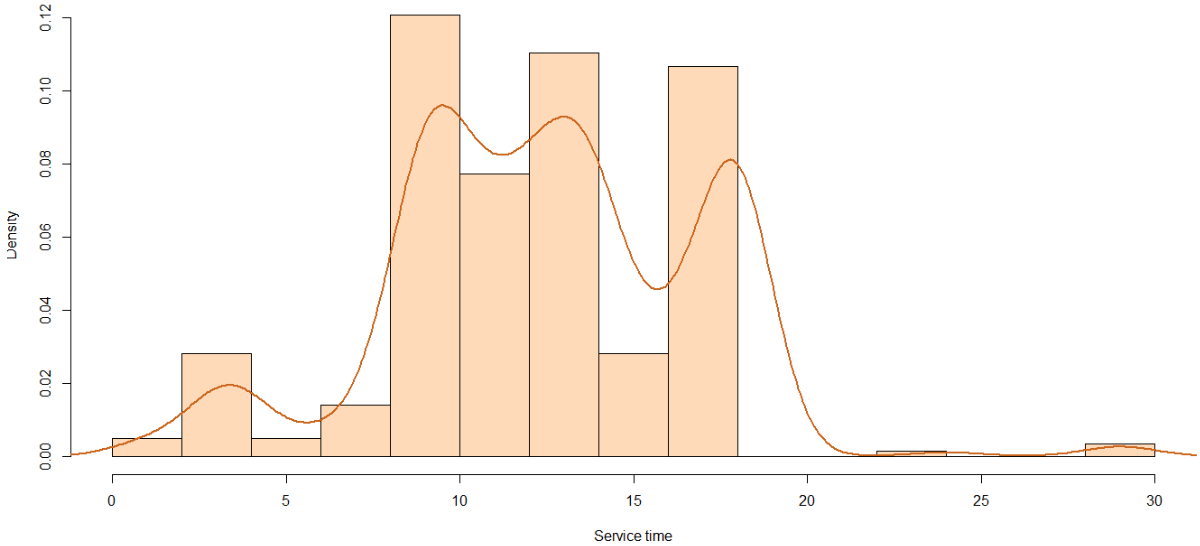


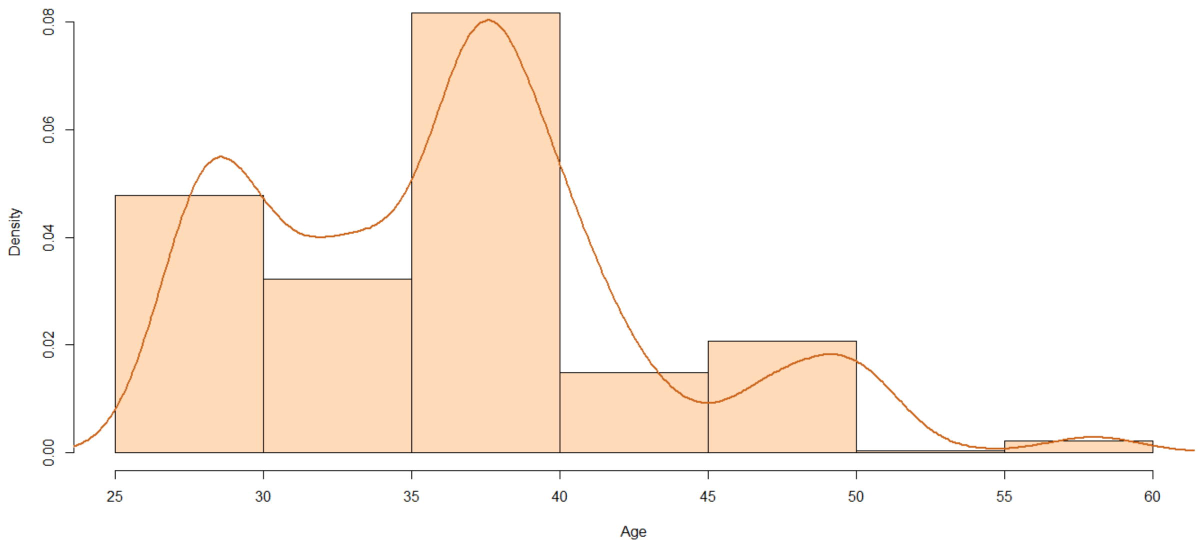


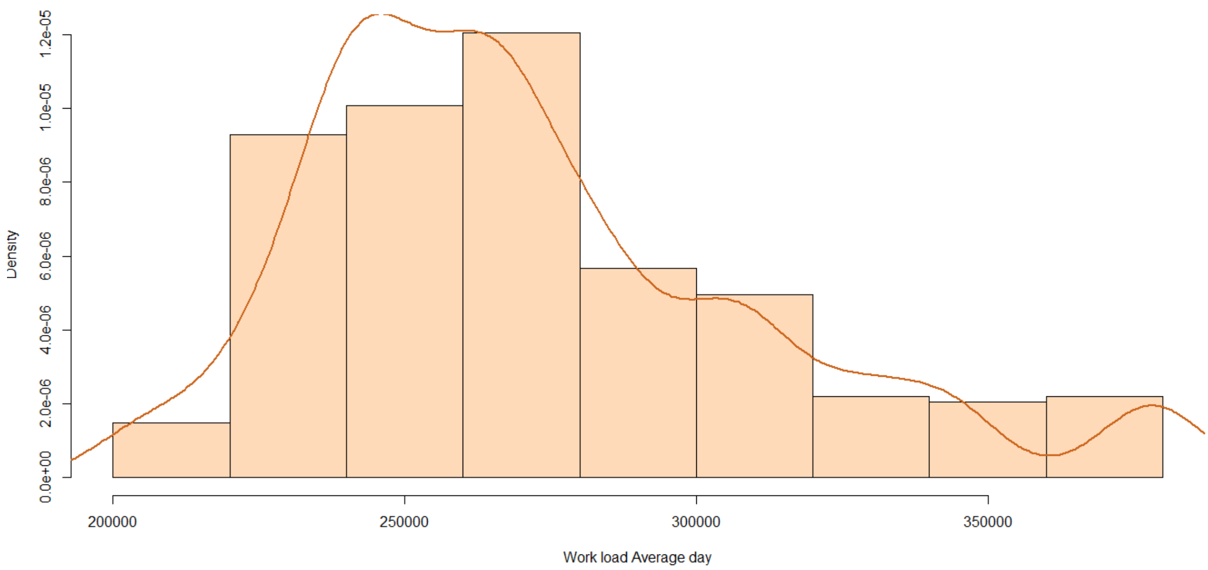
Continuous Variables

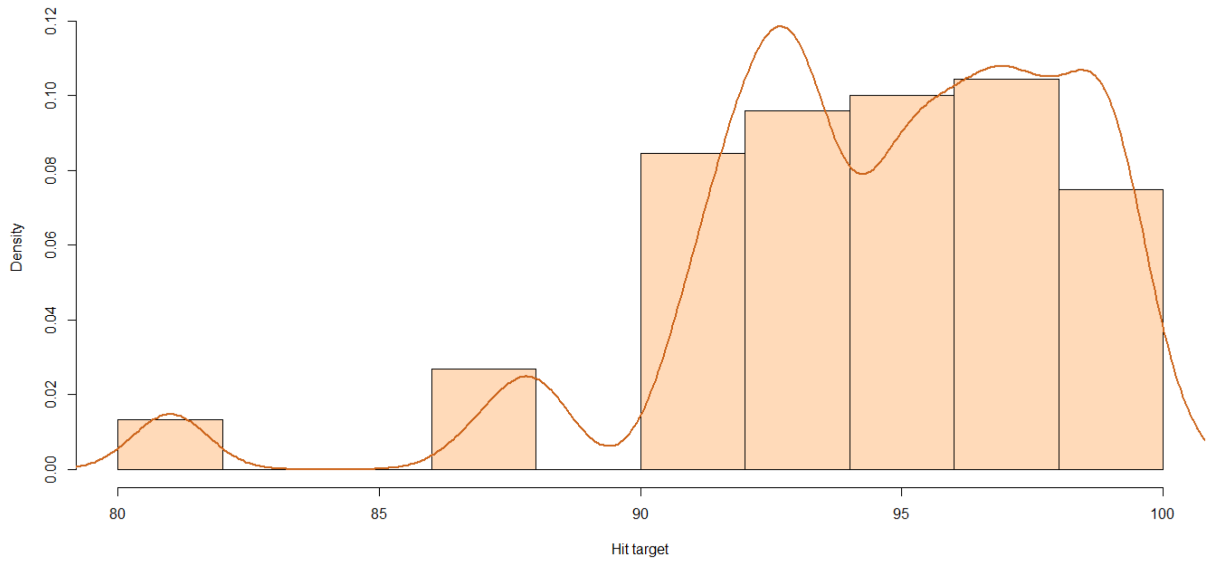


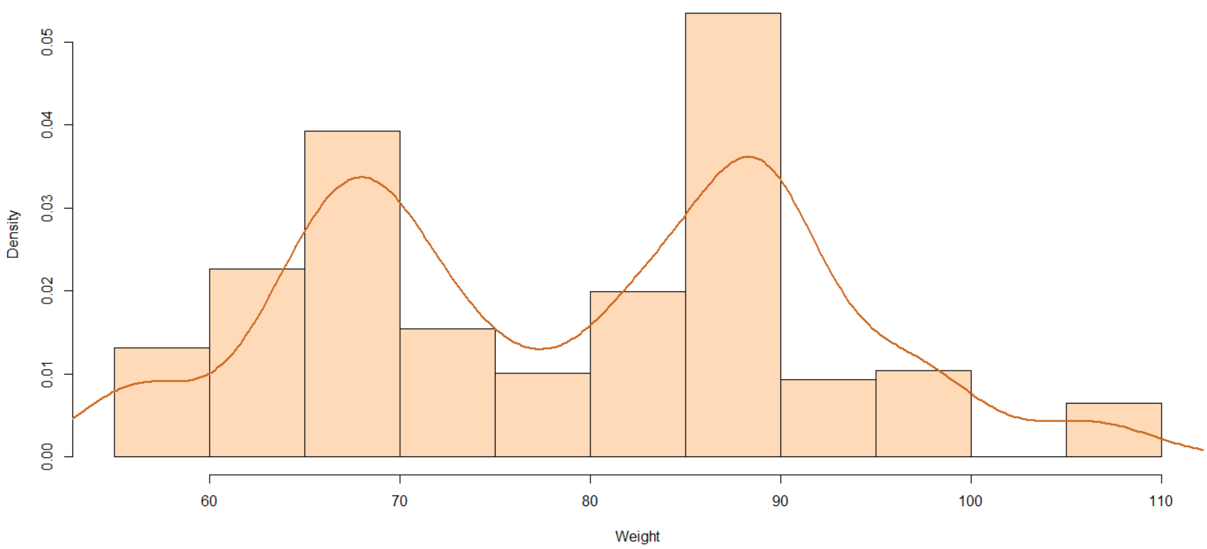

Density
0.00
0.01
0.02
0.03
0.04
10
20
30
40
50
Distance from Residence to Work

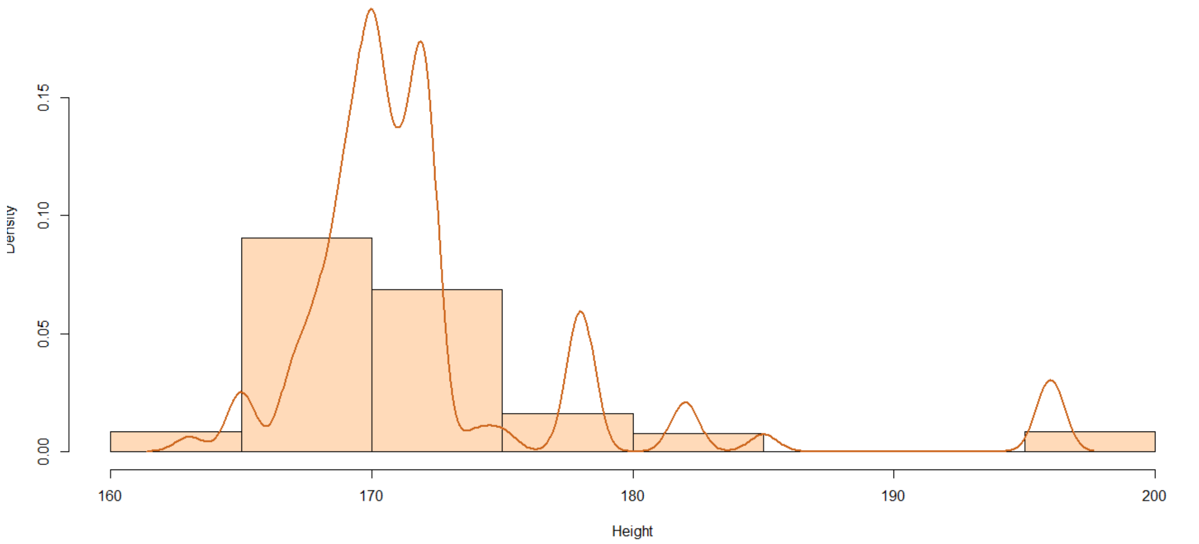

Density
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0
5
10
15
20
25
30
Service time

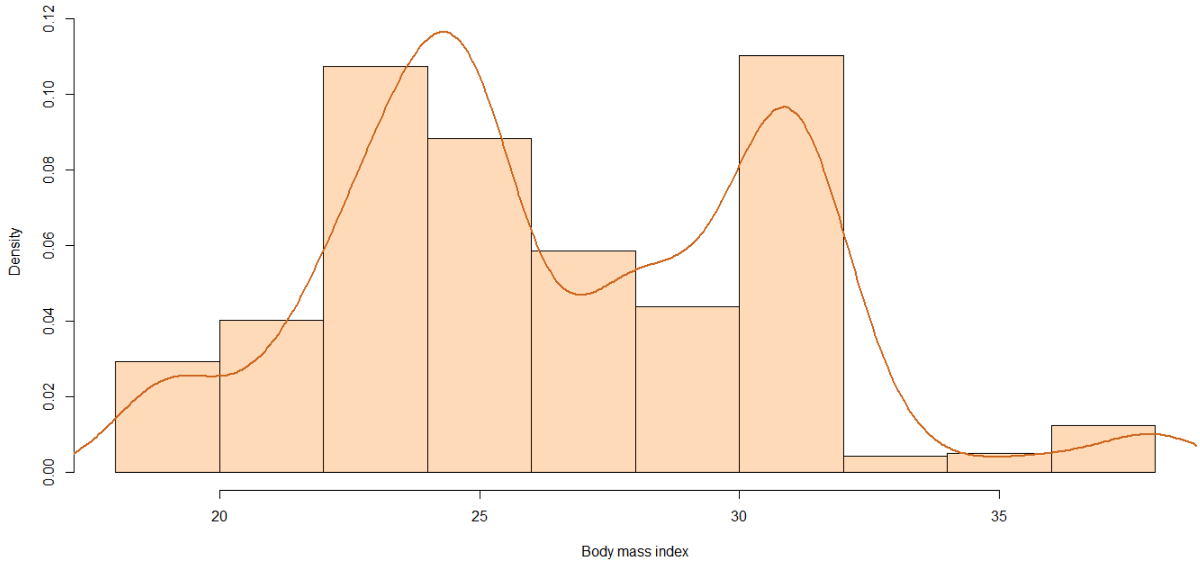

Density
0.00
0.02
0.04
0.06
0.08
25
30
35
40
45
50
55
60
Age


Density
0.0e+00
2.0e-06
4.0e-06
6.0e-06
8.0e-06
1.0e-05
1.2e-05
200000
250000
300000
350000
Work load Average day

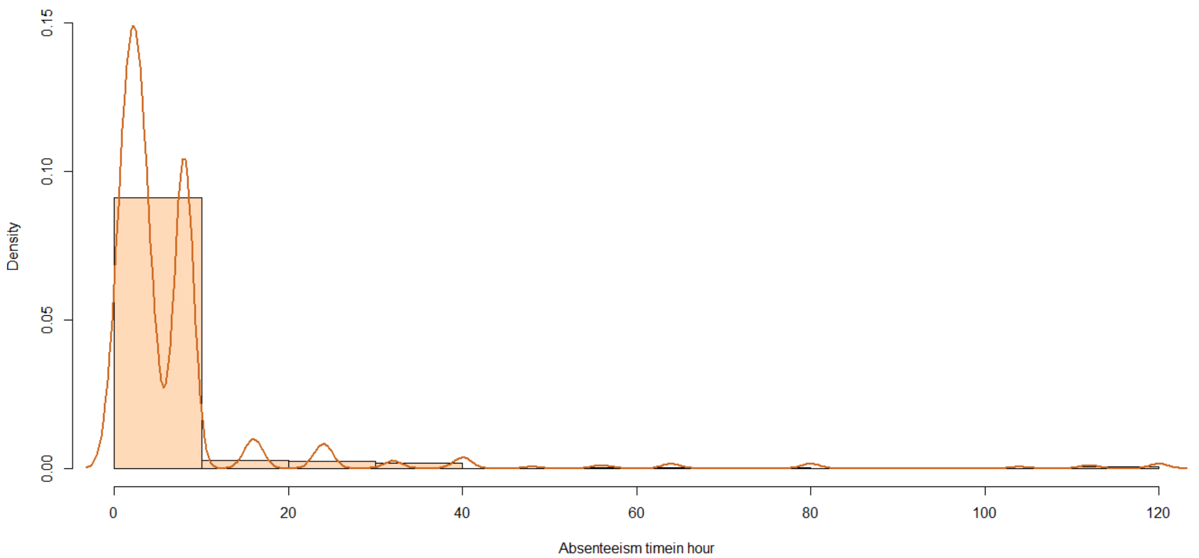

Density
0.00
0.02
0.04
0.06
0.08
0.10
0.12
80
85
90
95
100
Hit target


Density
0.00
0.01
0.02
0.03
0.04
0.05
60
70
80
90
100
110
Weight



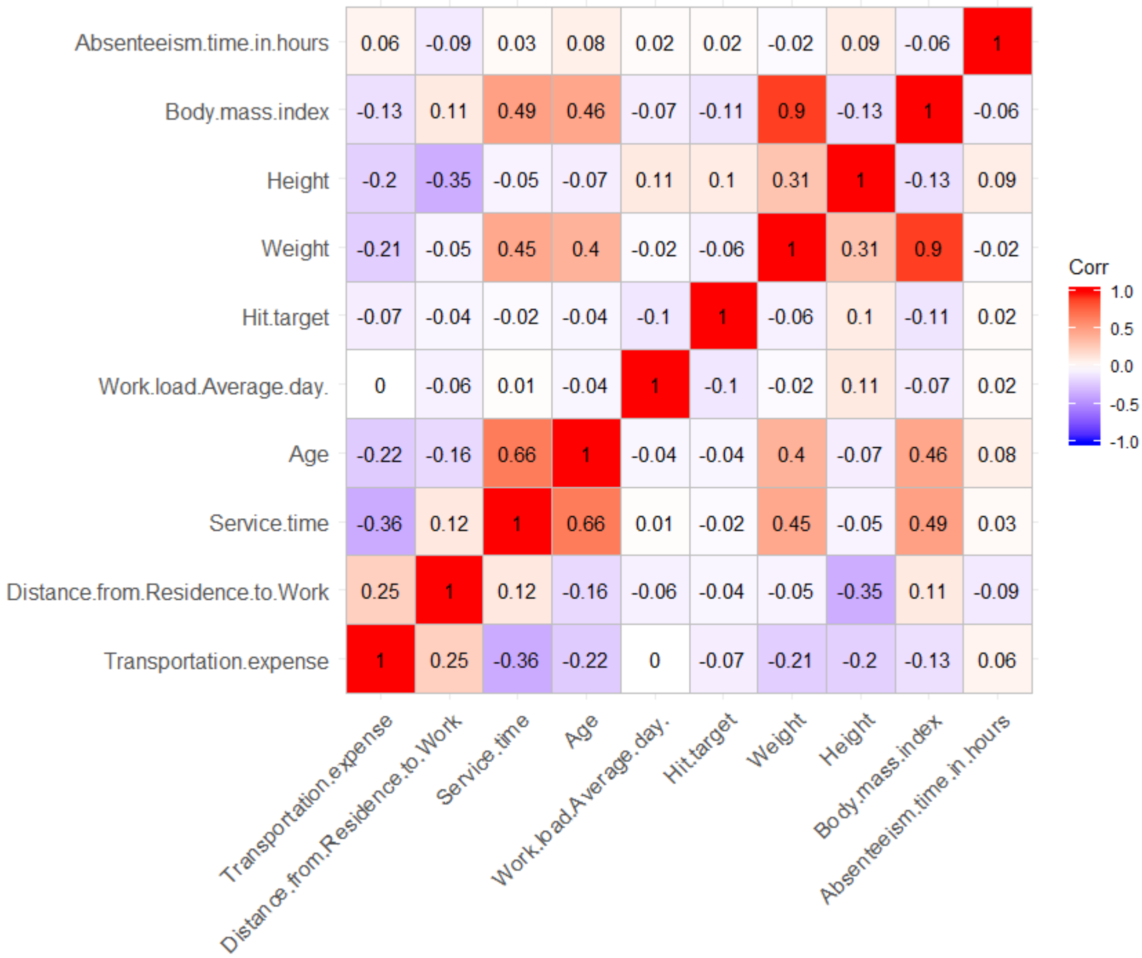


Target Variable

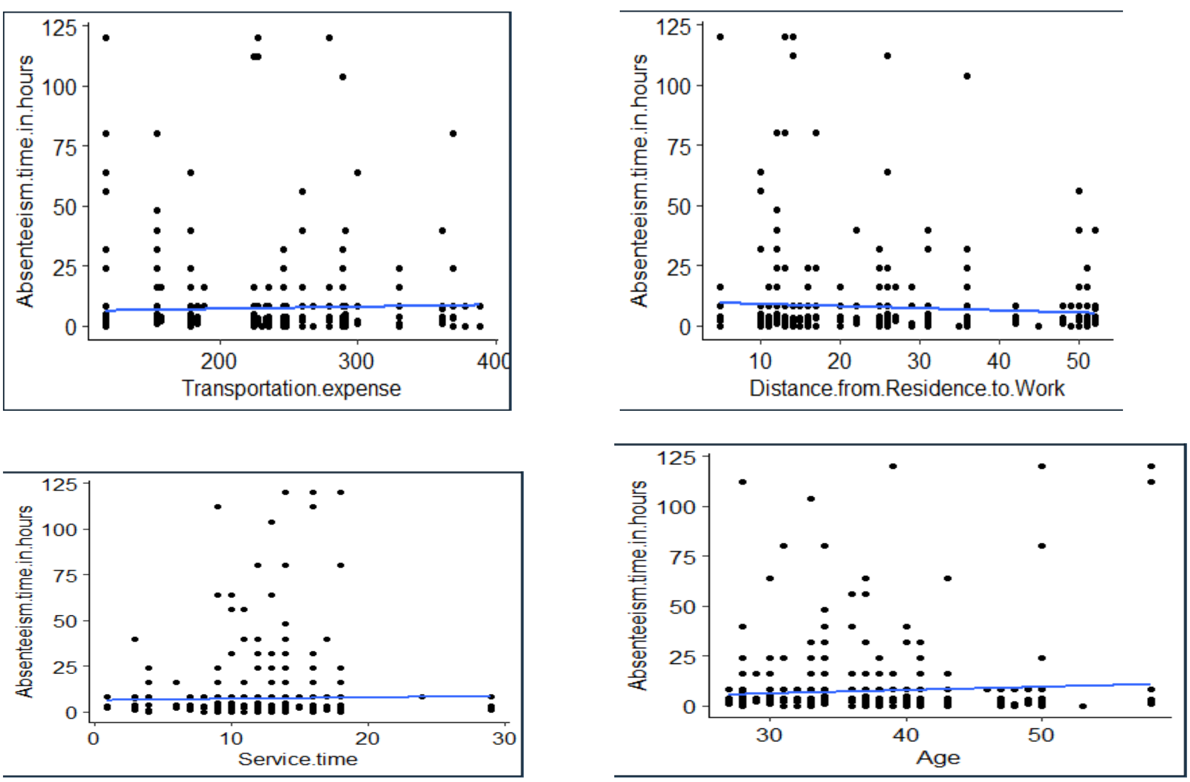


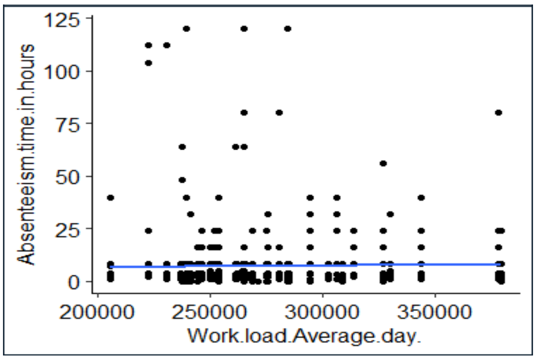
2.4 Bi-variate Analysis

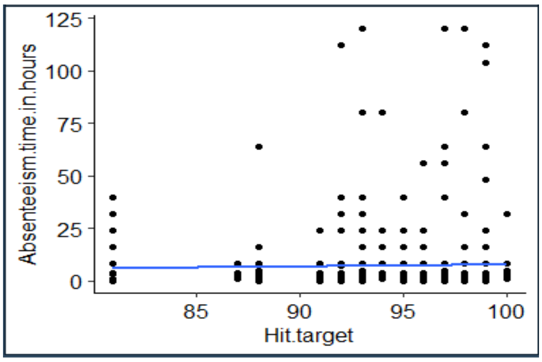
Correlation Plot of Continuous Variables

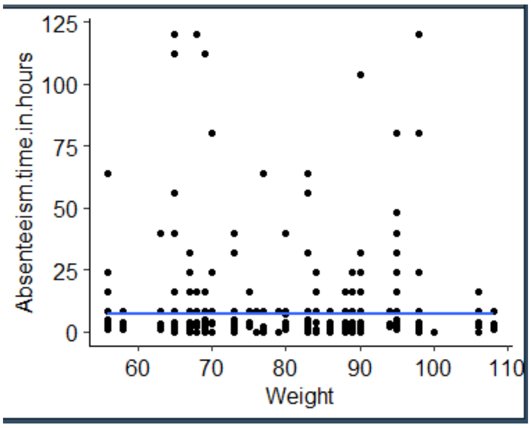


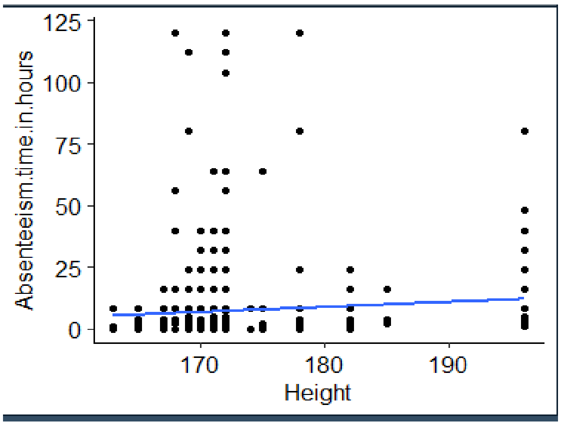
Scatter Plot B/W Continuous Predictor and Target Variable

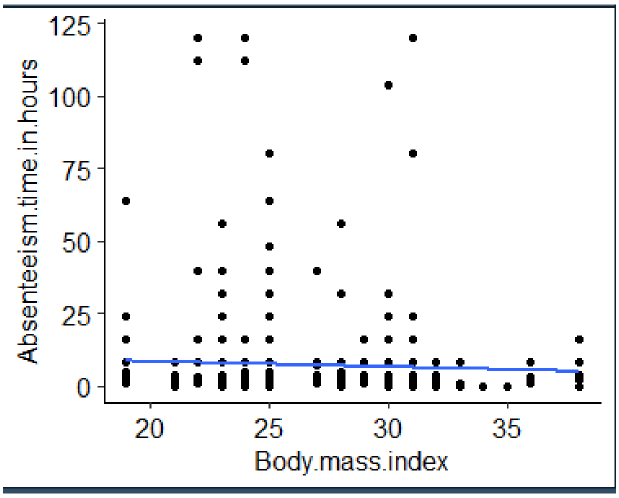




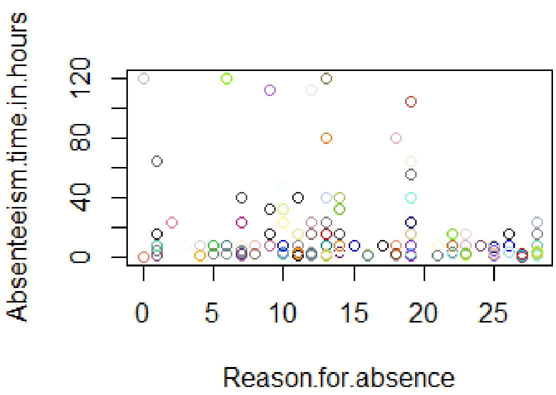


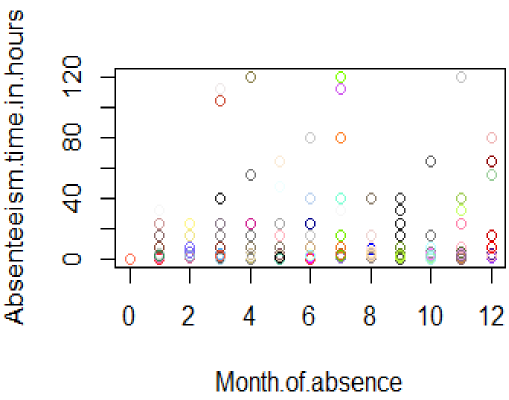


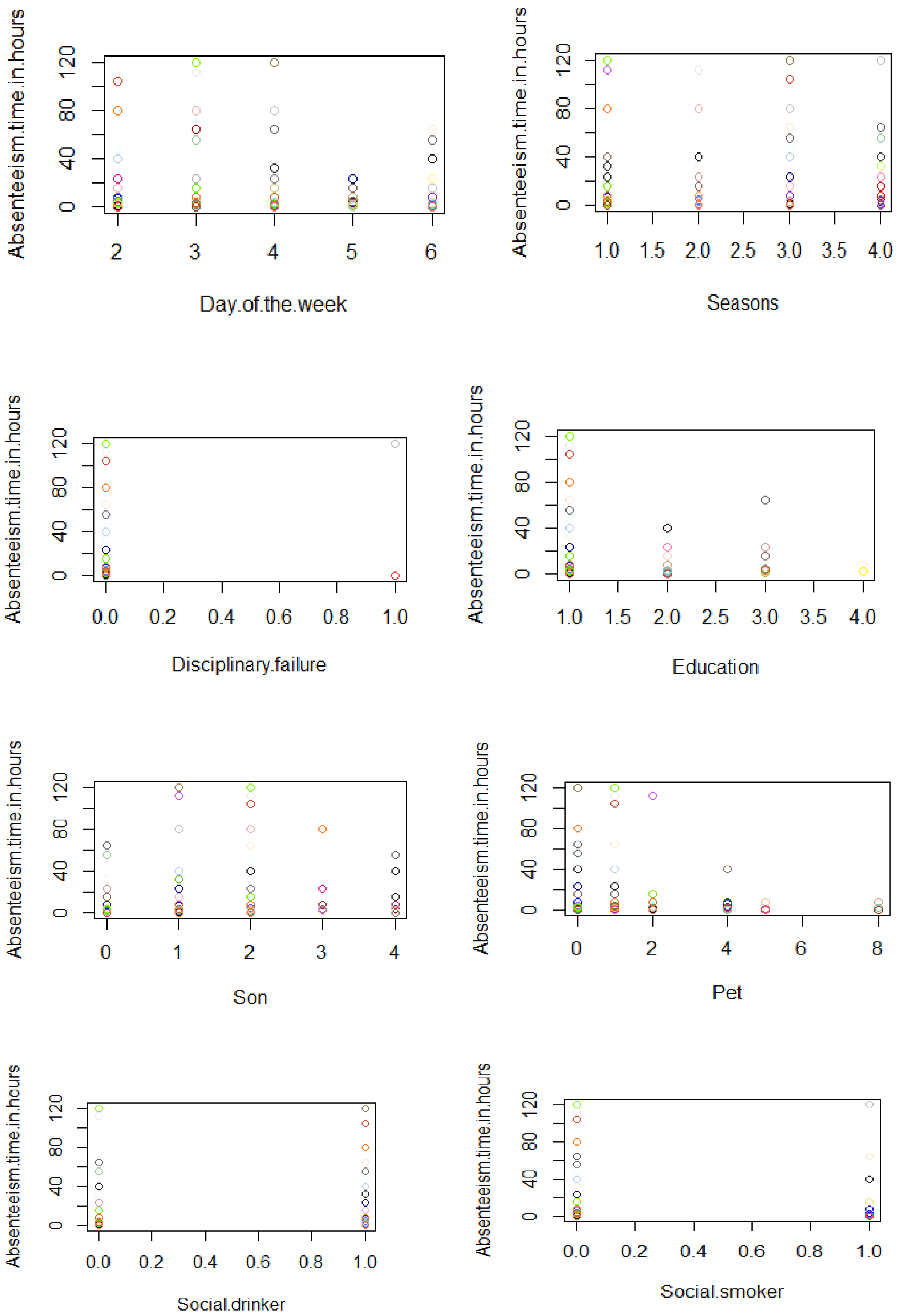




Plot B/W Categorical Predictor and Target Variable






Absenteeism.time.in.hours
Day.of.the.week
Seasons
Disciplinary.failure
Education
Son
Pet
Social.drinker
Social.smoker

2.5 Missing Value Analysis

In statistics, missing data or missing values occur when no data value is stored for the variable in an observation. Missing values are a common occurrence in data analysis. These values can have a significant impact on the results or conclusions that would be drawn from these data. If a variable has more than 30% of its values missing, then those values can be ignored, or the column itself is ignored. In our case, none of the columns have a high percentage of missing values. The maximum missing percentage is 4.18% i.e., Body Mass Index column. The missing values have been computed using KNN computation method.

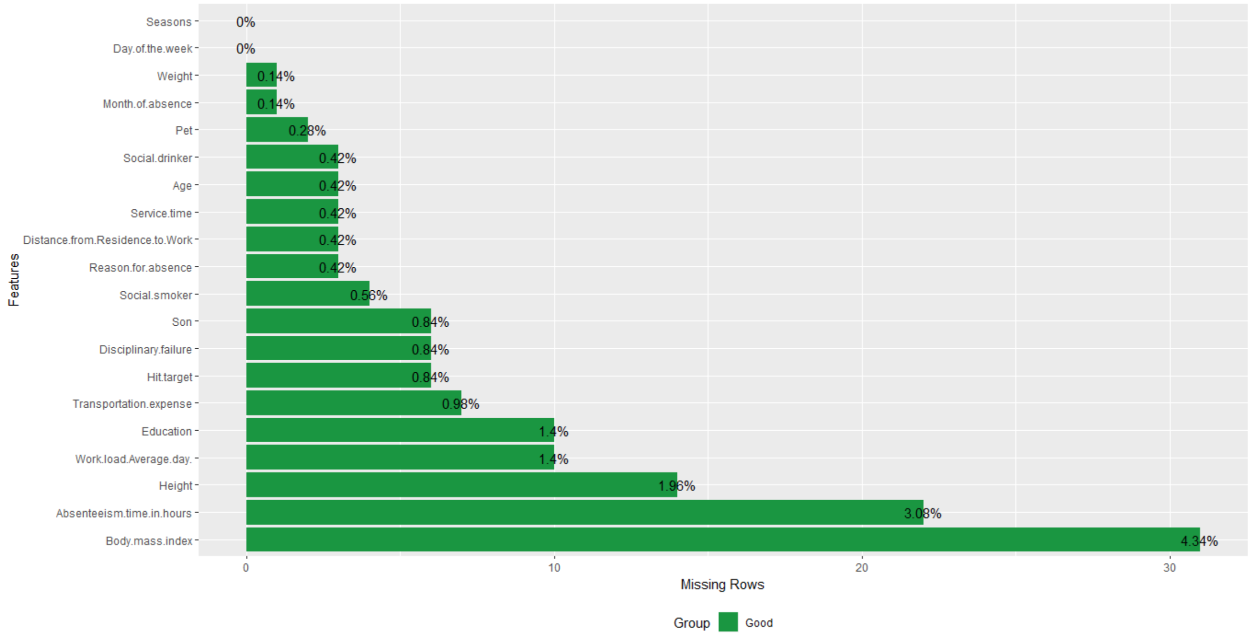
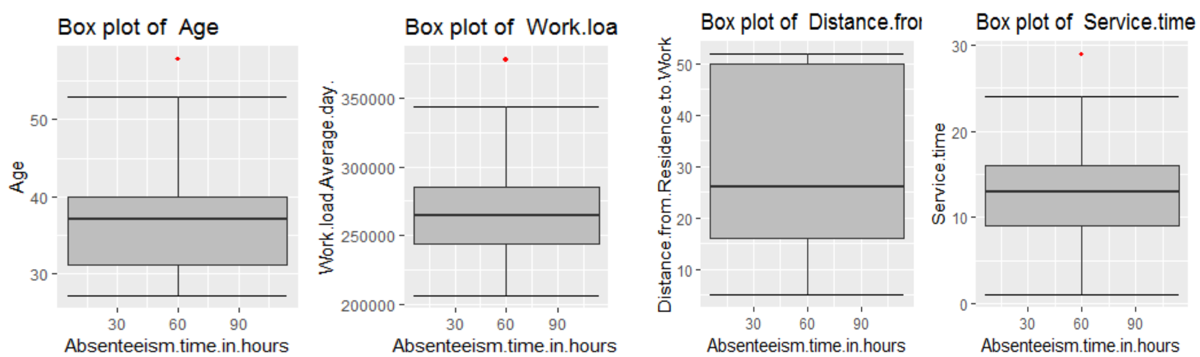


Figure: Missing Values by percentage in each column

2.6 Outlier Analysis

It can be observed from the distribution of variables that almost none of the variables are normally distributed. The skew in these distributions can be explained by the presence of outliers and extreme values in the data. One of the steps in pre-processing involves the detection and removal of such outliers. In this project, we use boxplot to visualize and remove outliers. Any value lying outside of the lower and upper whisker of the boxplot are outliers.
In figure we have plotted the boxplots of the 10 predictor variables with respect to **Absenteeism time in hour**.



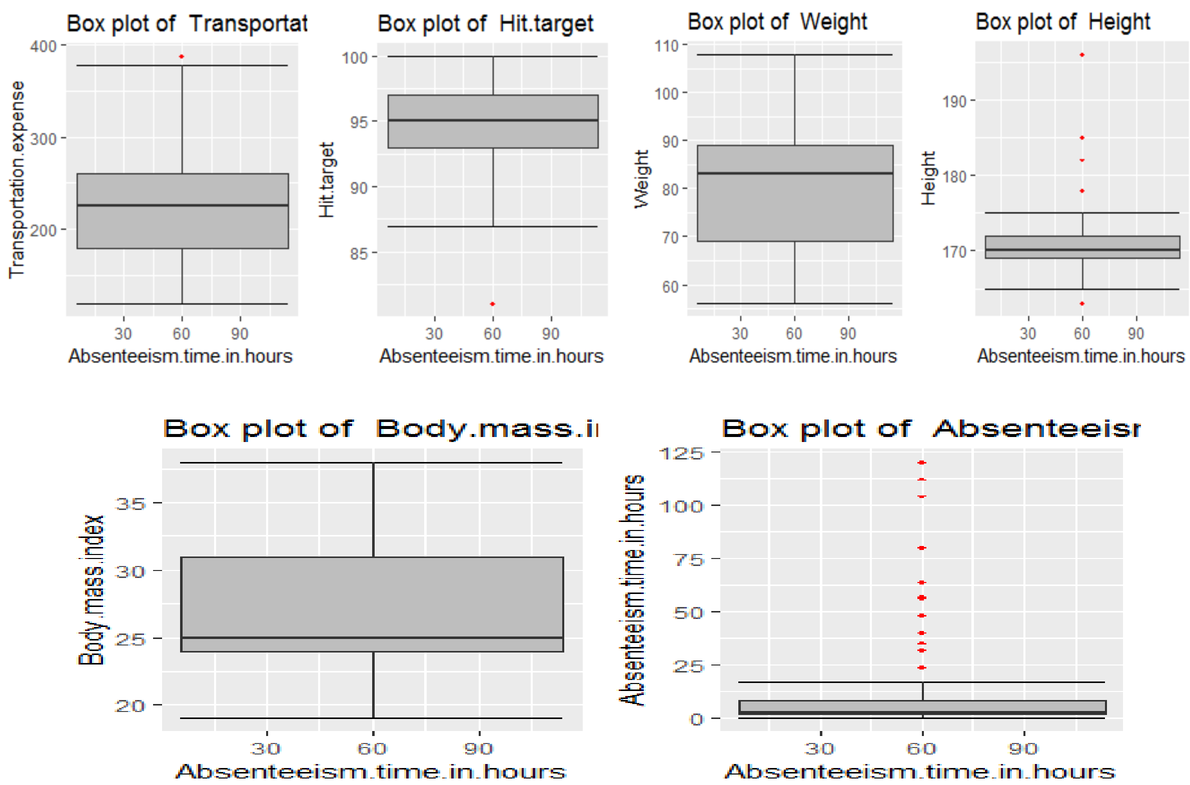


Figure – Boxplots of continuous variables with outliers

2.7 Feature Selection

Feature Selection reduces the complexity of a model and makes it easier to interpret. It also reduces overfitting. Features are selected based on their scores in various statistical tests for their correlation with the outcome variable. Correlation plot is used to find out if there is any multicollinearity between variables. The highly collinear variables are dropped and then the model is executed. Also from our business understanding we also remove those columns which might not be contributing to the dataset for example Height Column.

From correlation analysis we have found that Weight and Body Mass Index has high correlation (>0.7), so we have excluded the Wight column from our dataset

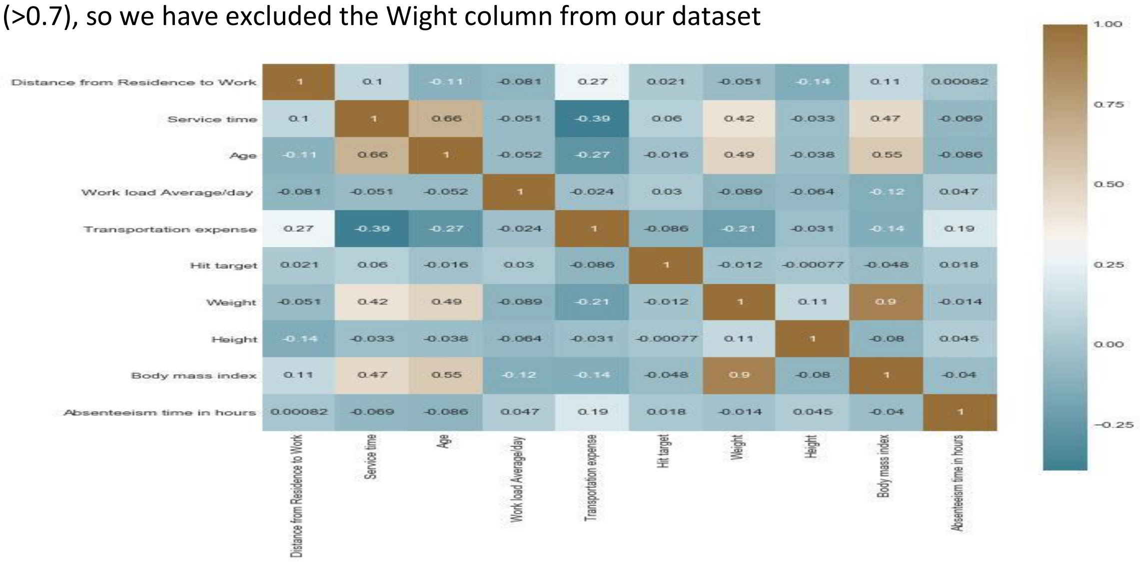


Figure – Correlation plot of Continuous variables

2.8 Feature Scaling

Feature scaling is a method used to standardize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data pre-processing step.

Most classifiers calculate the distance between two points by the Euclidean distance. If one of the features has a broad range of values, the distance will be governed by this feature. Therefore, the range of all features should be scaled so that each feature contributes proportionately to the model and our result is not biased towards the variable greater in magnitude.
We are using **Auto scaling or z-transformation** as the scaling procedure which results in a zero mean and unit variance of any descriptor variable.

	Distance from Residence to Work	Service time	Age	Work load Average/day	Transportation expense	Hit target	Body mass index
0	0.427479	0.133232	-0.522597	-0.85201	1.034678	0.680946	0.781660
1	-1.122520	1.333188	2.271209	-0.85201	-1.551711	0.680946	1.015842
2	1.438348	1.333188	0.299111	-0.85201	-0.629081	0.680946	1.015842
3	-1.661650	0.373223	0.463452	-0.85201	0.883427	0.680946	-0.623430
4	0.427479	0.133232	-0.522597	-0.85201	1.034678	0.680946	0.781660

Table - Scaled Continuous variables

2.9 Principal Component Analysis (PCA)

Principal component analysis is a method of extracting important variables (in form of components) from a large set of variables available in a data set. It extracts low dimensional set of features from a high dimensional data set with a motive to capture as much information as possible.

After creating dummy variable of categorical variables, the data would have 81 columns and 740(In Python)/715(In R) observations. This high number of columns leads to bad accuracy.

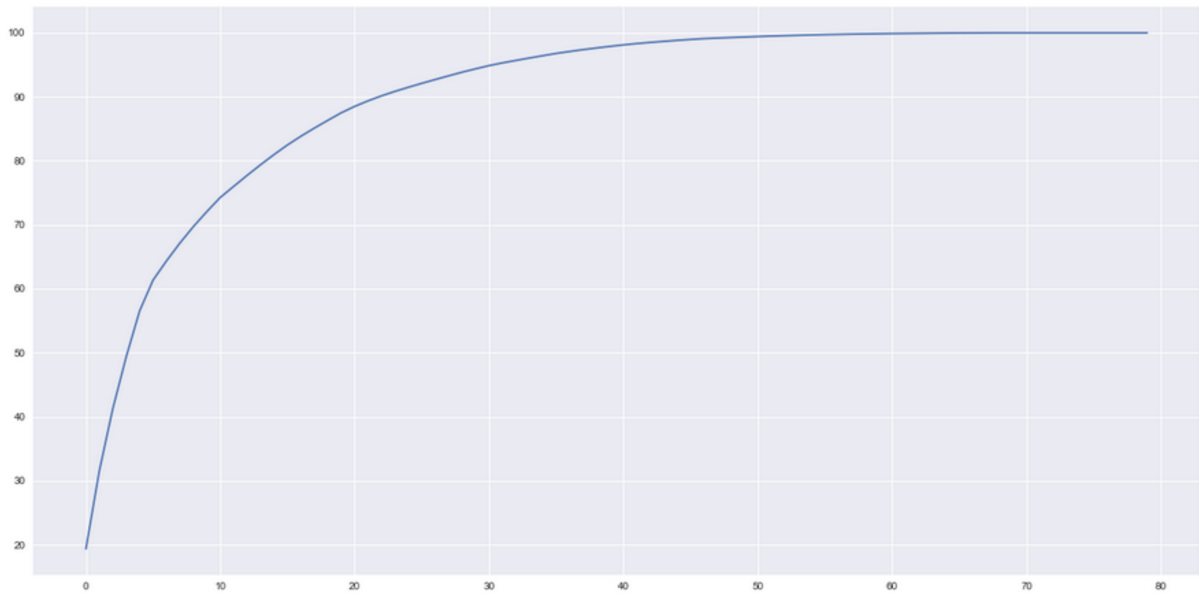


Figure - PCA plot for variables

After applying PCA algorithm and observing the above Cumulative Scree Plot, it can be observed that almost 95% of the data can be explained by 45 variables out of 80. Hence, we choose only 45 variables as input to the models.

Chapter 3: Modelling

3.1 Model Selection

After a thorough pre-processing we will be using some regression models on our processed data to predict the target variable. The target variable in our model is a continuous variable i.e., Absenteeism time in hours. Hence the models that we choose are Decision Tree and Random Forest, XG Boost and Linear Regression. The error metric chosen for the given problem statement is Root Mean Square Error (RMSE), Mean Absolute Error (MAE). We will select our best fit model by comparing both the statistical values.

3.2 Decision Tree

Decision Tree algorithm belongs to the family of supervised learning algorithms. Decision trees are used for both classification and regression problems.

A decision tree is a tree where each node represents a feature (attribute), each link (branch) represents a decision (rule) and each leaf represents an outcome (categorical or continues value). The general motive of using Decision Tree is to create a training model which can be used to predict class or value of target variables by learning decision rules inferred from prior data (training data).

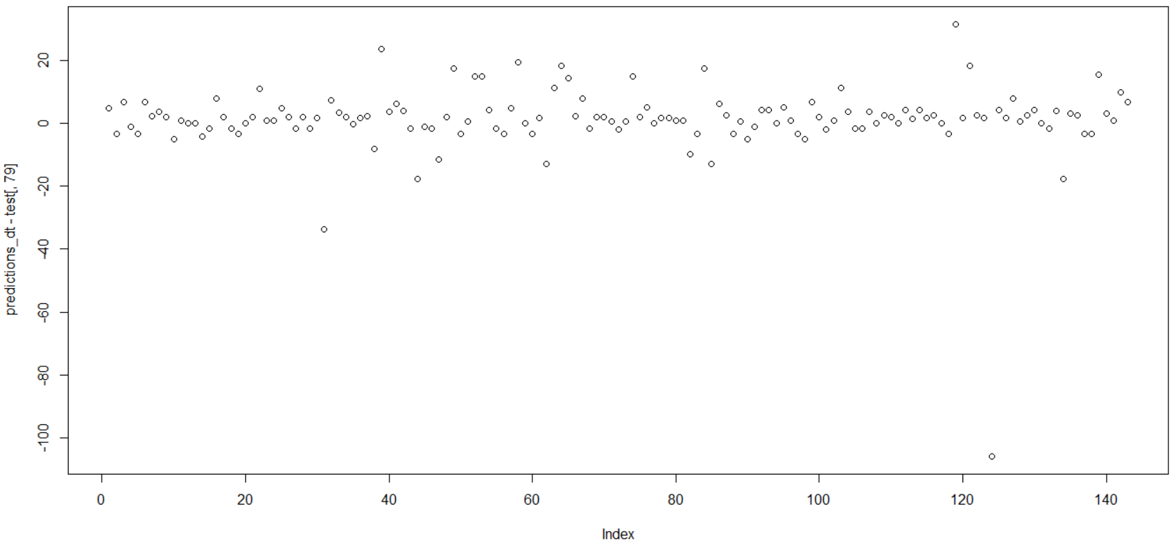


Figure – Residual Plot of (predicted values-actual values) for Decision Tree

The MAE, RMSE, MSE & R^2 values for the given project in R and Python are:

DECISION TREE	MAE	RMSE	MSE	R^2
R	5.658996	11.717496	137.299703	Not Calculated
PYTHON	5.409	12.35	Not Calculated	0.0484

3.3 Random Forest

Random Forest is a supervised learning algorithm. Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. It can be used for both classification and regression problems. The method of combining trees is known as an ensemble method. Ensemble is nothing but a combination of weak learners (individual trees) to produce a strong learner.
The number of decision trees used for prediction in the forest is 500.

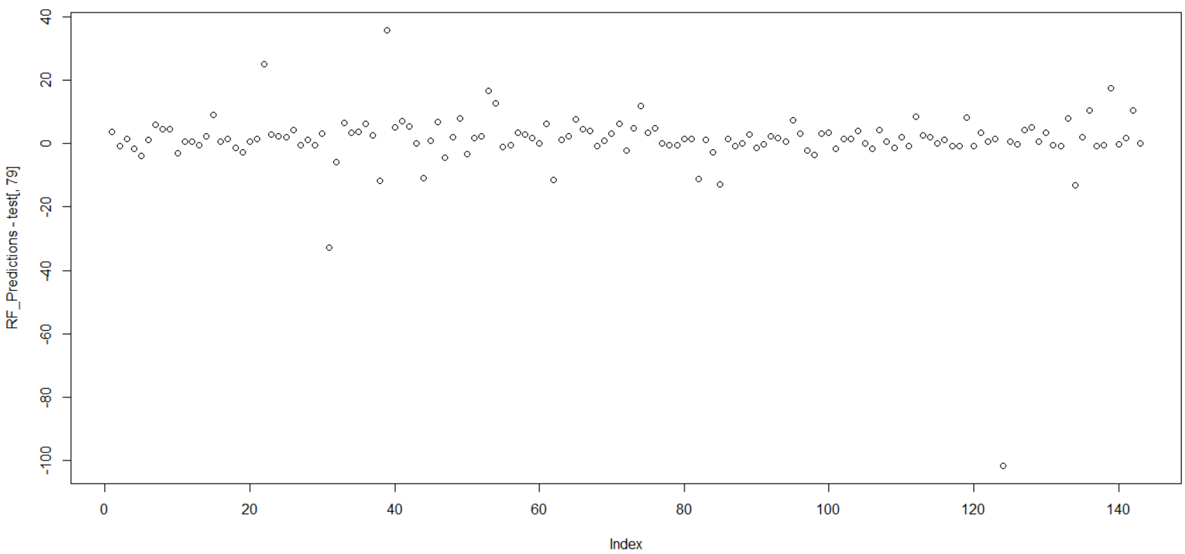


Figure – Residual Plot of (predicted values-actual values) for Random Forest

The MAE, RMSE, MSE & R^2 values for the given project in R and Python are:

DECISION TREE	MAE	RMSE	MSE	R^2
R	4.706784	10.819635	117.064493	Not Calculated
PYTHON	4.696	11.743	Not Calculated	0.140

3.4 XG Boost

XG Boost (Extreme Gradient Boosting) is an advanced and more efficient implementation of Gradient Boosting Algorithm discussed in the previous section.

Advantages over Other Boosting Techniques

- It is 10 times faster than the normal Gradient Boosting as it implements parallel processing. It is highly flexible as users can define custom optimization objectives and evaluation criteria, has an inbuilt mechanism to handle missing values.
- Unlike gradient boosting which stops splitting a node as soon as it encounters a negative loss, XG Boost splits up to the maximum depth specified and prunes the tree backward and removes splits beyond which there is an only negative loss.

Extreme gradient boosting can be done using the XG Boost package in R and Python.

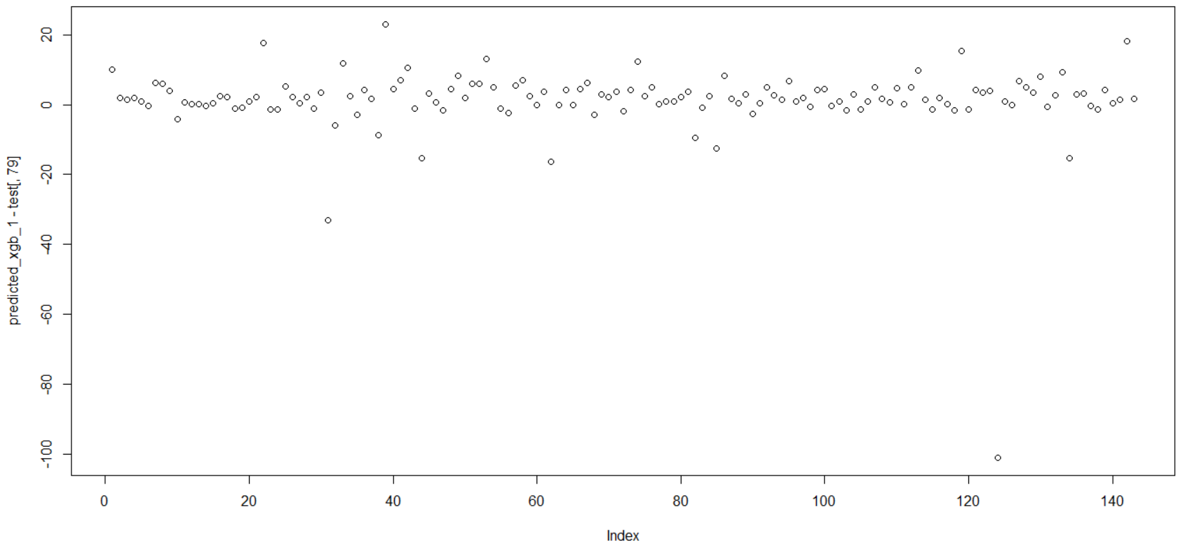


Figure – Residual Plot of (predicted values-actual values) for XG Boost

The MAE, RMSE, MSE & R^2 values for the given project in R and Python are:

DECISION TREE	MAE	RMSE	MSE	R^2
R	4.815342	10.635997	112.124428	Not Calculated
PYTHON	4.253	12.124	Not Calculated	0.0843

3.5 Linear Regression

Multiple linear regression is the most common form of linear regression analysis. Multiple linear regression is used to explain the relationship between one continuous dependent variable and two or more independent variables. The independent variables can be continuous or categorical.

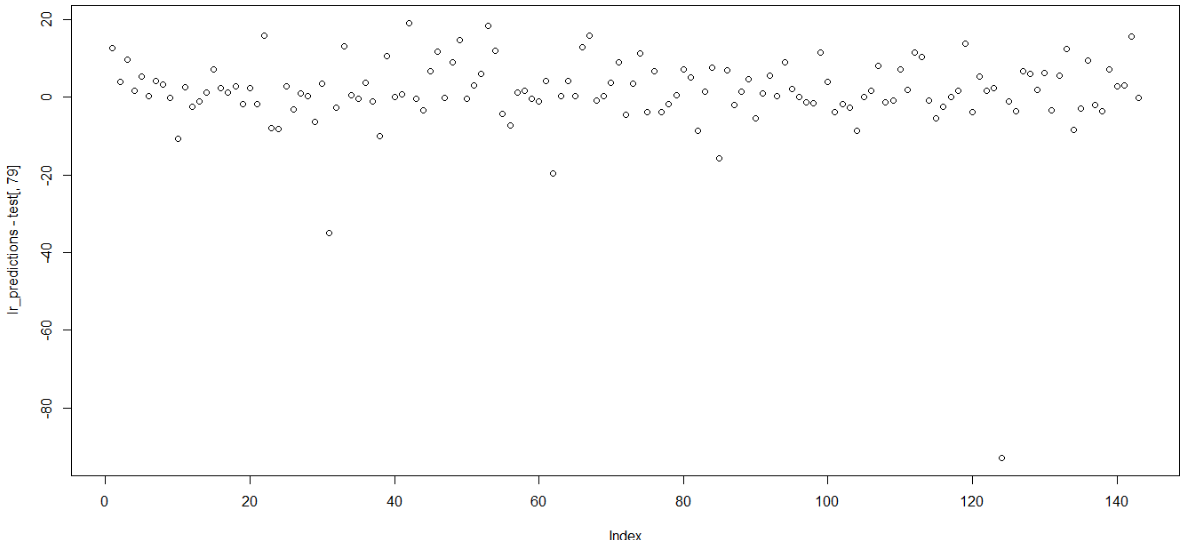


Figure - Residual Plot of (predicted values-actual values) for Linear Regression

The MAE, RMSE, MSE & R^2 values for the given project in R are:-

DECISION TREE	MAE	RMSE	MSE	R^2
R	5.47771	13.38506	179.15995	Not Calculated

Chapter 4: Conclusion

4.1 Model Evaluation

In the previous chapter we have seen the Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Squared Error (MSE) and R-Squared Value of different models.

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors). Residuals are a measure of how far from the regression line data points are, RMSE is a measure of how spread out these residuals are. In other words, it tells you how concentrated the data is around the line of best fit.

Mean Absolute Error (MAE): MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It's the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight.
Mean Squared Error (MSE): MSE basically measures average squared error of our predictions. For each point, it calculates square difference between the predictions and the target and then average those values.
The higher this value, the worse the model is. It is never negative, since we're squaring the individual prediction-wise errors before summing them, but would be zero for a perfect model.

R-squared is a relative measure of fit, RMSE is an absolute measure of fit. As the square root of a variance, RMSE can be interpreted as the standard deviation of the unexplained variance and has the useful property of being in the same units as the response variable. Lower values of RMSE and higher value of R-Squared Value indicate better fit.

Comparison

Similarities: Both MAE and RMSE express average model prediction error in units of the variable of interest. Both metrics can range from 0 to ∞ and are indifferent to the direction of errors. They are negatively-oriented scores, which means lower values are better.

Differences: Taking the square root of the average squared errors has some interesting implications for RMSE. Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the RMSE should be more useful when large errors are particularly undesirable. The three tables below show examples where MAE is steady and RMSE increases as the variance associated with the frequency distribution of error magnitudes also increases.

4.2 Model Selection

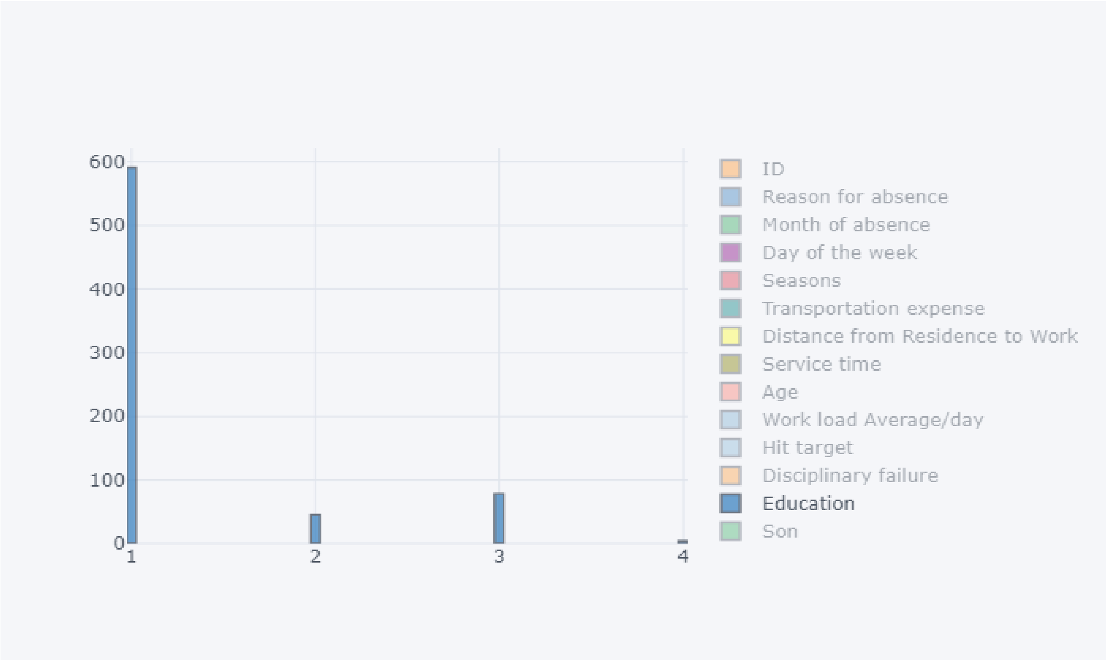
From the observation of all RMSE Value and MSE Value and Residual Plot we have concluded that **XG Boost** has comparable minimum value of RMSE and MSE.

4.3 Solutions of Problem Statement

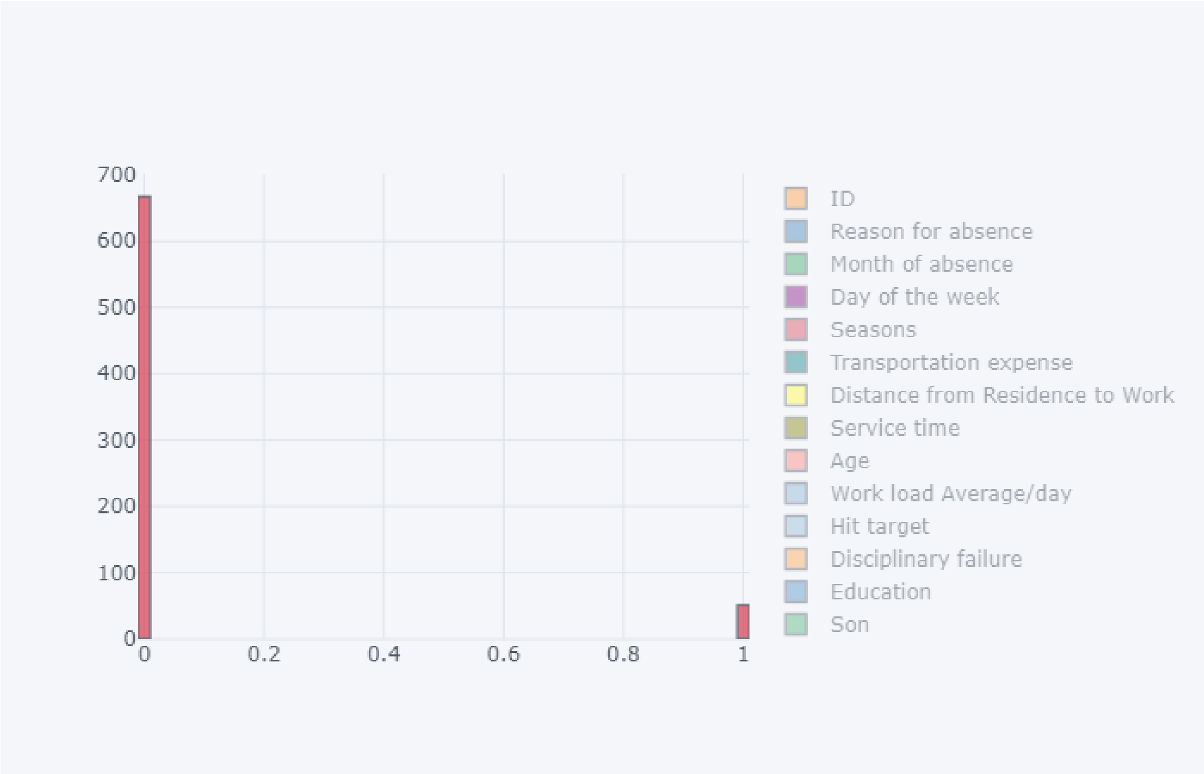
4.3.1 What changes company should bring to reduce the number of absenteeism?

Solution:

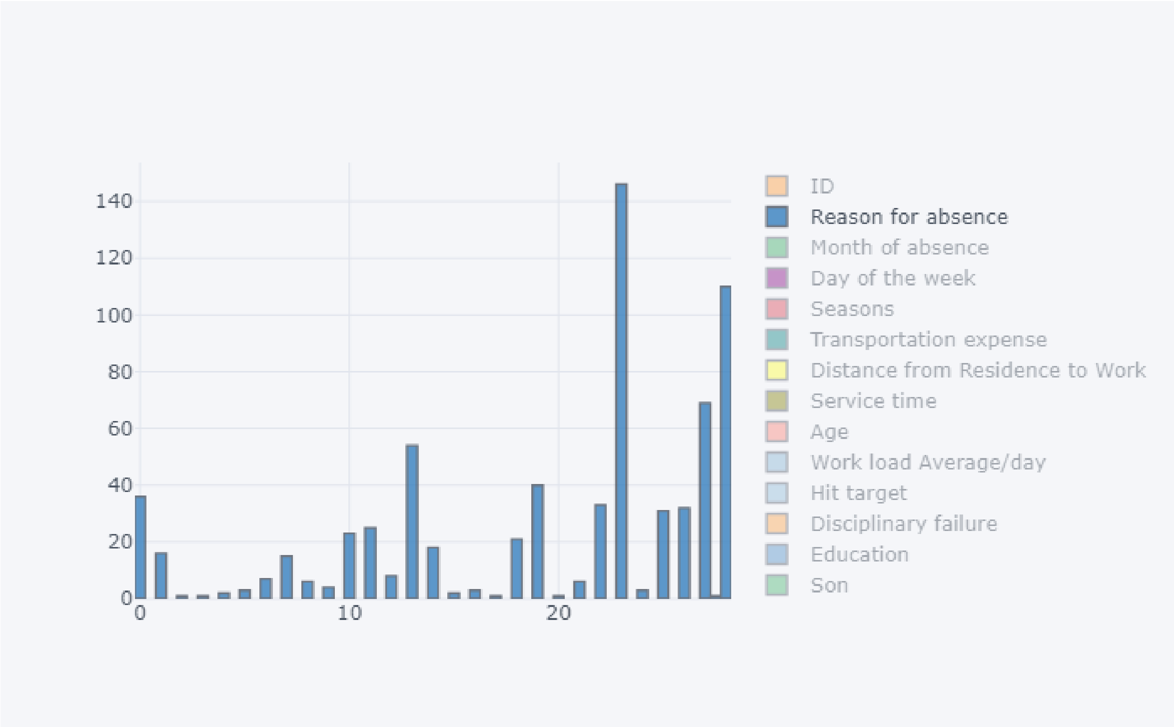
1. It is observed that employee with low education have maximum absentee time.



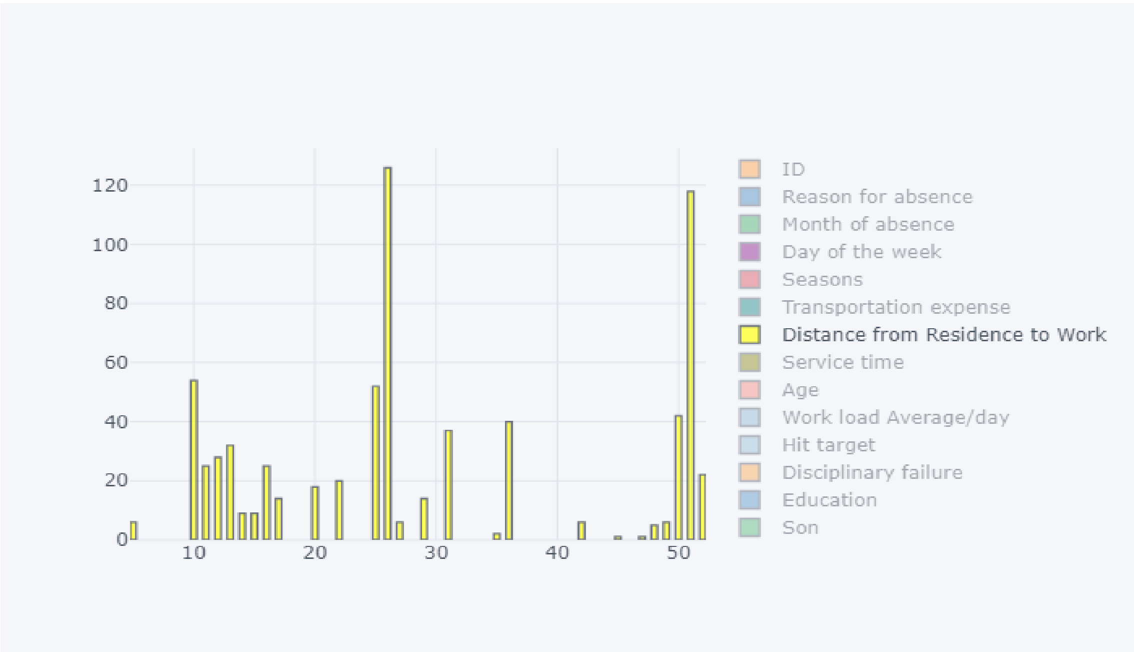
2. Employees who are social smoker have more absentee hour than who are not social smoker.



3. Most often Reason for absence are medical consultation and dental consultation, company should take care of it.



4. Employees who has Distance from Residence to Work high more tends to absent more.



4.3.2 How much losses every month can we project in 2011 if same trend of absenteeism continues?

Solution:

Considering the losses to be the absenteeism time in hours, if the same trend of absenteeism continues, then the total losses per month is as shown in the graph below.

Employees are absent the most in the month of March, with total Absenteeism hours equal to 458.2 hours. Employees are absent the least in the month of January, with total Absenteeism hours equal to 173.6.

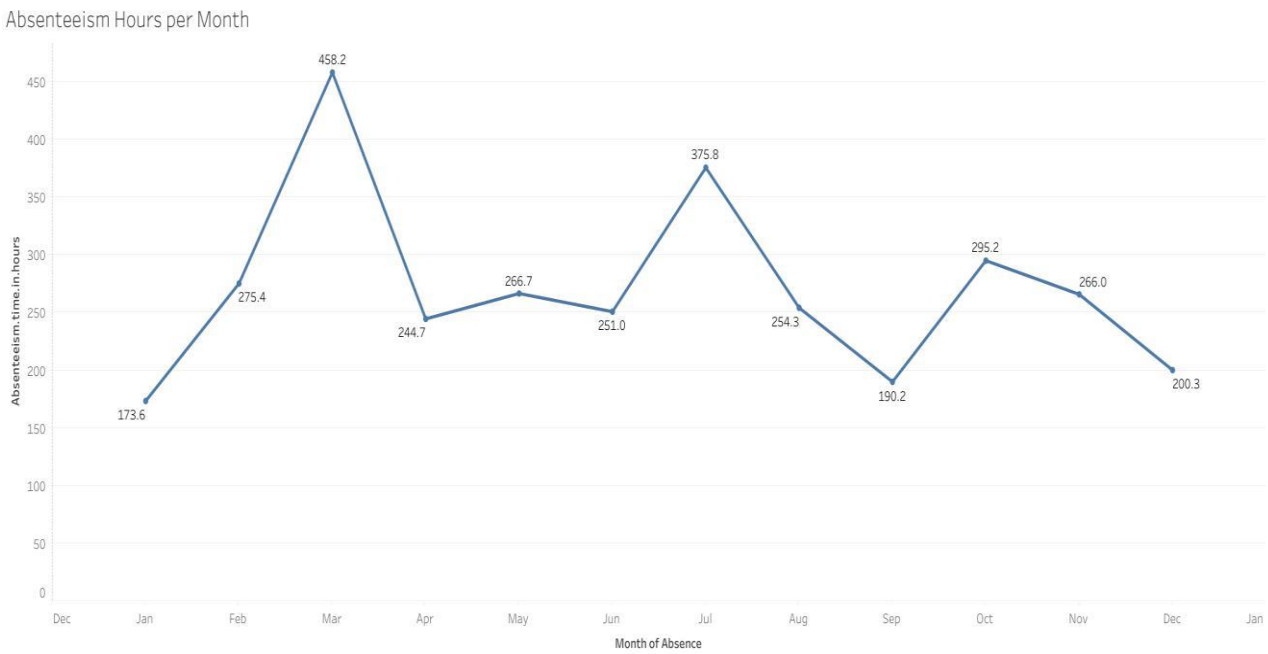


Fig - Absenteeism Hours per Month

Below table shows the monthly losses of absenteeism hours:

Month	Absent Hours
January	173.6
February	275.4
March	458.2
April	244.7
May	266.7
June	251
July	375.8
August	254.3
September	190.2
October	295.2
November	266
December	200.3

Chapter 5: R Code

```
#Remove all objects stored
rm(list = ls())



#set working directory
setwd("E:/R/Project")


#Load .xls file in R with gdata package
library(gdata)
library(dplyr)
library(tidyverse)


#Loading .xls file by removing first column i.e. ID Column
df <- read.xls("Absenteeism_at_work_Project.xls")[, -1] #Since employee ID column is not useful for
our Analysi
str(df)


###############Changing Variable type########################
#Removing commas(,) from Work.load.Average.day. column
df$Work.load.Average.day. <- as.character(df$Work.load.Average.day.)
df$Work.load.Average.day.<- gsub("\\,", "", df$Work.load.Average.day.)
df$Work.load.Average.day. <- as.integer(df$Work.load.Average.day.)
dim(df)
str(df)

#Converting our original dataset df into tibble just for better understanding
Absenteeism_Data <- as_tibble(df)
#Removing Duplicate rows from  data
Absenteeism_Data <- Absenteeism_Data %>% distinct()
dim(Absenteeism_Data)
str(Absenteeism_Data)
summary(Absenteeism_Data)
View(Absenteeism_Data)


########################################Exploratory Data
Analysis#####################################################
library(DataExplorer)
library(ggplot2)
library(ggpubr)
theme_set(theme_pubr())

#Checking the dimension of the input dataset and the type of variables
dim(Absenteeism_Data)
plot_str(Absenteeism_Data)
```

```
continuous_vars = c("Transportation.expense", "Distance.from.Residence.to.Work", "Service.time", 
"Age", 'Transportation expense',
              "Work.load.Average.day.", "Hit.target", "Weight", "Height", "Body.mass.index", 
"Absenteeism.time.in.hours")

categorical_vars = c("Reason.for.absence","Month.of.absence" ,"Day.of.the.week",
               "Seasons","Disciplinary.failure", "Education", "Son",
               "Social.drinker", "Social.smoker","Pet")



       #######################Univariate Analysis######################

                    ##For Categorical Data##


#Reason.for.absence
 cat_count <- Absenteeism_Data %>% group_by(Reason.for.absence) %>% summarise(counts = n())
  ggplot(cat_count, aes(x = Reason.for.absence, y = counts)) +
    geom_bar(fill = "#0073C2FF", stat = "identity") + scale_x_continuous(breaks = 0:28) +
    geom_text(aes(label = counts), vjust = -0.3) + 
    theme_pubclean()

#Month.of.absence
  cat_count_1 <- Absenteeism_Data %>% group_by(Month.of.absence) %>% summarise(counts = n())
  ggplot(cat_count_1, aes(x = Month.of.absence, y = counts)) +
    geom_bar(fill = "#0073C2FF", stat = "identity") + scale_x_continuous(breaks = 0:12) +
    geom_text(aes(label = counts), vjust = -0.3) + 
    theme_pubclean()

  #Day.of.the.week
  cat_count_2 <- Absenteeism_Data %>% group_by(Day.of.the.week) %>% summarise(counts = n())
  ggplot(cat_count_2, aes(x = Day.of.the.week, y = counts)) +
    geom_bar(fill = "#0073C2FF", stat = "identity") +
    geom_text(aes(label = counts), vjust = -0.3) + 
    theme_pubclean()
  #Seasons
  cat_count_3 <- Absenteeism_Data %>% group_by(Seasons) %>% summarise(counts = n())
  ggplot(cat_count_3, aes(x = Seasons, y = counts)) +
    geom_bar(fill = "#0073C2FF", stat = "identity") +
    geom_text(aes(label = counts), vjust = -0.3) + 
    theme_pubclean()
  #Disciplinary.failure
  cat_count_4 <- Absenteeism_Data %>% group_by(Disciplinary.failure) %>% summarise(counts = 
n())
  ggplot(cat_count_4, aes(x = Disciplinary.failure, y = counts)) +
    geom_bar(fill = "#0073C2FF", stat = "identity") + scale_x_continuous(breaks = 0:1) +
    geom_text(aes(label = counts), vjust = -0.3) + 
    theme_pubclean()
  #Education
  cat_count_5 <- Absenteeism_Data %>% group_by(Education) %>% summarise(counts = n())
  ggplot(cat_count_5, aes(x = Education, y = counts)) +
    geom_bar(fill = "#0073C2FF", stat = "identity") +
    geom_text(aes(label = counts), vjust = -0.3) + 
    theme_pubclean()
```

```
#Social.drinker
cat_count_6 <- Absenteeism_Data %>% group_by(Social.drinker) %>% summarise(counts = n())
ggplot(cat_count_6, aes(x = Social.drinker, y = counts)) +
  geom_bar(fill = "#0073C2FF", stat = "identity") + scale_x_continuous(breaks = 0:1) +
  geom_text(aes(label = counts), vjust = -0.3) +
  theme_pubclean()
#Social.smoker
cat_count_7 <- Absenteeism_Data %>% group_by(Social.smoker) %>% summarise(counts = n())
ggplot(cat_count_7, aes(x = Social.smoker, y = counts)) +
  geom_bar(fill = "#0073C2FF", stat = "identity") + scale_x_continuous(breaks = 0:1) +
  geom_text(aes(label = counts), vjust = -0.3) +
  theme_pubclean()
#Son
cat_count_8 <- Absenteeism_Data %>% group_by(Son) %>% summarise(counts = n())
ggplot(cat_count_8, aes(x = Son, y = counts)) +
  geom_bar(fill = "#0073C2FF", stat = "identity") +
  geom_text(aes(label = counts), vjust = -0.3) +
  theme_pubclean()
#Pet
cat_count_9 <- Absenteeism_Data %>% group_by(Pet) %>% summarise(counts = n())
ggplot(cat_count_9, aes(x = Pet, y = counts)) +
  geom_bar(fill = "#0073C2FF", stat = "identity") + scale_x_continuous(breaks = 0:8) +
  geom_text(aes(label = counts), vjust = -0.3) +
  theme_pubclean()




##FOR CONTINUOUS VARIABLE

 #Transportation.expense

  hist(Absenteeism_Data$Transportation.expense,
      col = "peachpuff",
      border = "black",
      prob = TRUE,
      xlab = "Transportation expense",
      main = NULL)
  lines(density(Absenteeism_Data$Transportation.expense, na.rm = TRUE), lwd = 2, col =
"chocolate3")

  #Distance.from.Residence.to.Work
  hist(Absenteeism_Data$Distance.from.Residence.to.Work,
      col = "peachpuff",
      border = "black",
      prob = TRUE,
      xlab = "Distance from Residence to Work",
      main = NULL)
  lines(density(Absenteeism_Data$Distance.from.Residence.to.Work, na.rm = TRUE), lwd = 2, col =
"chocolate3")

  #Service.time
  hist(Absenteeism_Data$Service.time,
      col = "peachpuff",
      border = "black",
```

```
    prob = TRUE,
    xlab = "Service time",
    main = NULL)
lines(density(Absenteeism_Data$Service.time, na.rm = TRUE), lwd = 2, col = "chocolate3")

#Age
hist(Absenteeism_Data$Age,
    col = "peachpuff",
    border = "black",
    prob = TRUE,
    xlab = "Age",
    main = NULL)
lines(density(Absenteeism_Data$Age, na.rm = TRUE), lwd = 2, col = "chocolate3")

#Work.load.Average.day.
hist(Absenteeism_Data$Work.load.Average.day.,
    col = "peachpuff",
    border = "black",
    prob = TRUE,
    xlab = "Work load Average day",
    main = NULL)
lines(density(Absenteeism_Data$Work.load.Average.day., na.rm = TRUE), lwd = 2, col =
"chocolate3")

#Hit.target
DENS_0 <- density(Absenteeism_Data$Hit.target, na.rm = TRUE)
YMax_0 <- max(DENS_0$y)
hist(Absenteeism_Data$Hit.target,
    col = "peachpuff",
    border = "black",
    ylim = c(0, YMax_0),
    prob = TRUE,
    xlab = "Hit target",
    main = NULL)
lines(DENS_0, lwd = 2, col = "chocolate3")

#Weight
hist(Absenteeism_Data$Weight,
    col = "peachpuff",
    border = "black",
    prob = TRUE,
    xlab = "Weight",
    main = NULL)
lines(density(Absenteeism_Data$Weight, na.rm = TRUE), lwd = 2, col = "chocolate3")

#Height
DENS_1 <- density(Absenteeism_Data$Height, na.rm = TRUE)
YMax_1 <- max(DENS_1$y)
hist(Absenteeism_Data$Height,
    col = "peachpuff",
    border = "black",
    ylim = c(0, YMax_1),
    prob = TRUE,
    xlab = "Height",
    main = NULL)
```

```
lines(DENS_1, lwd = 2, col = "chocolate3")

#Body.mass.index
DENS_2 <- density(Absenteeism_Data$Body.mass.index, na.rm = TRUE)
YMax_2 <- max(DENS_2$y)
hist(Absenteeism_Data$Body.mass.index,
    col = "peachpuff",
    border = "black",
    ylim = c(0, YMax_2),
    prob = TRUE,
    xlab = "Body mass index",
    main = NULL)
lines(density(Absenteeism_Data$Body.mass.index, na.rm = TRUE), lwd = 2, col = "chocolate3")

#Absenteeism.time.in.hours
DENS <- density(Absenteeism_Data$Absenteeism.time.in.hours, na.rm = TRUE)
YMax <- max(DENS$y)
hist(Absenteeism_Data$Absenteeism.time.in.hours,
    col = "peachpuff",
    border = "black",
    ylim = c(0, YMax),
    prob = TRUE,
    xlab = "Absenteeism timein hour",
    main = "Target Variable")
lines(DENS, lwd = 2, col = "chocolate3")


        ########################Bivariate Analysis#######################

library(ggcorrplot)
library(dlookr)
# Correlation Plot of Numerical Variables
Absenteeism_Data[ ,c(5:10,17:20)] %>%
  select_if(is.numeric) %>% na.omit() %>%
  cor() %>%
  ggcorrplot(lab = T)

## Compute a correlation matrix
corr_1 <- Absenteeism_Data[, c(1:4, 20)] %>% na.omit() %>% cor()
corr_1
corr_2 <- Absenteeism_Data[, c(5:6, 20)] %>% na.omit() %>% cor()
corr_2
corr_3 <- Absenteeism_Data[, c(7:10, 20)] %>% na.omit() %>% cor()
corr_3
corr_4 <- Absenteeism_Data[, c(11:14, 20)] %>% na.omit() %>% cor()
corr_4
corr_5 <- Absenteeism_Data[, c(15:19, 20)] %>% na.omit() %>% cor()
corr_5


          #####Grouped Descriptive Statistics#####

            ##Grouped Numerical Variables##
#Transportation.expense
ggplot(Absenteeism_Data, aes(x=Transportation.expense, y=Absenteeism.time.in.hours )) +
```

```
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)
#Distance.from.Residence.to.Work
ggplot(Absenteeism_Data, aes(x=Distance.from.Residence.to.Work, y=Absenteeism.time.in.hours ))
+
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)
#Service.time
ggplot(Absenteeism_Data, aes(x=Service.time, y=Absenteeism.time.in.hours )) +
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)
#Age
ggplot(Absenteeism_Data, aes(x=Age, y=Absenteeism.time.in.hours )) +
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)
#Work.load.Average.day.
ggplot(Absenteeism_Data, aes(x=Work.load.Average.day., y=Absenteeism.time.in.hours )) +
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)
#Hit.target
ggplot(Absenteeism_Data, aes(x=Hit.target, y=Absenteeism.time.in.hours )) +
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)
#Weight
ggplot(Absenteeism_Data, aes(x=Weight, y=Absenteeism.time.in.hours )) +
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)
#Height
ggplot(Absenteeism_Data, aes(x=Height, y=Absenteeism.time.in.hours )) +
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)
#Body.mass.index
ggplot(Absenteeism_Data, aes(x=Body.mass.index, y=Absenteeism.time.in.hours )) +
  geom_point(na.rm = T)+
  geom_smooth(method=lm, se=FALSE, na.rm = T)


##Grouped Categorical Variables##
#Reason.for.absence
plot(Absenteeism.time.in.hours~Reason.for.absence, data = Absenteeism_Data, col = colors())

#Month.of.absence
plot(Absenteeism.time.in.hours~Month.of.absence, data = Absenteeism_Data, col = colors())

#Day.of.the.week
plot(Absenteeism.time.in.hours~Day.of.the.week, data = Absenteeism_Data, col = colors())
#Seasons
plot(Absenteeism.time.in.hours~Seasons, data = Absenteeism_Data, col = colors())
#Disciplinary.failure
plot(Absenteeism.time.in.hours~Disciplinary.failure, data = Absenteeism_Data, col = colors())
#Education
plot(Absenteeism.time.in.hours~Education, data = Absenteeism_Data, col = colors())
#Son
plot(Absenteeism.time.in.hours~Son, data = Absenteeism_Data, col = colors())
#Social.drinker
```

```
plot(Absenteeism.time.in.hours~Social.drinker, data = Absenteeism_Data, col = colors())
#Social.smoker
plot(Absenteeism.time.in.hours~Social.smoker, data = Absenteeism_Data, col = colors())
#Pet
plot(Absenteeism.time.in.hours~Pet, data = Absenteeism_Data, col = colors())


        #####################Missing Value Treatment#####################

#Visualizing Missing Values for each variable
plot_missing(Absenteeism_Data)
#Checking missing values in each columns with their missing percentages
profile_missing(Absenteeism_Data)
#Imputing missing values using MICE package
library(mice)
library(VIM)
imp.Absenteeism_Data <- mice(Absenteeism_Data, m=5, maxit = 50, seed = 500)
summary(imp.Absenteeism_Data)
stripplot(imp.Absenteeism_Data, pch = 20, cex = 1.2)
imp.Absenteeism_Data$imp$Reason.for.absence
#Replacing missing values with imputed values
Absenteeism_Data_complete <- complete(imp.Absenteeism_Data, 1)

#########################################Outlier
Analysis#########################################
library(Hmisc)

#Visualizing outlier using boxplots
 #selecting only continuous variable
numeric_data = Absenteeism_Data_complete[ ,c(5:10,17:20)]


cnames = colnames(numeric_data)
for (i in 1:length(cnames))
{
  assign(paste0("gn",i), ggplot(aes_string(y = (cnames[i])), data =
subset(Absenteeism_Data_complete))+
      stat_boxplot(geom = "errorbar", width = 0.5) +
      geom_boxplot(outlier.colour="red", fill = "grey" ,outlier.shape=18,
            outlier.size=2, notch=FALSE) +
      theme(legend.position="bottom")+
      labs(y=cnames[i])+
      ggtitle(paste("Box plot for",cnames[i])))
}
## Plotting plots together #2, 6, 7, 9
gridExtra::grid.arrange(gn1,gn2,gn3,ncol=3)
gridExtra::grid.arrange(gn4,gn5,gn6,ncol=3)
gridExtra::grid.arrange(gn7,gn8,gn9, gn10, ncol=4)

#Investigating summary of continuous variables to look for outlier
summary(Absenteeism_Data_complete[ ,5:9])
summary(Absenteeism_Data_complete[ ,c(10,17:20)])

#replacing all outliers with NA's
  #Transportation.expense
```

```
range_T <- 260 + 1.5*IQR(Absenteeism_Data_complete$Transportation.expense)
Absenteeism_Data_complete$Transportation.expense[Absenteeism_Data_complete$Transportatio
n.expense > range_T] <- NA
  #Service.time
range_ST <- 16 + 1.5*IQR(Absenteeism_Data_complete$Service.time)
Absenteeism_Data_complete$Service.time[Absenteeism_Data_complete$Service.time > range_ST]
<- NA
  #Age
range_A <- 40 + 1.5*IQR(Absenteeism_Data_complete$Age)
Absenteeism_Data_complete$Age[Absenteeism_Data_complete$Age > range_A] <- NA
 #Work.load.Average.day.
range_WL <- 294217 + 1.5*IQR(Absenteeism_Data_complete$Work.load.Average.day.)
Absenteeism_Data_complete$Work.load.Average.day.[Absenteeism_Data_complete$Work.load.Av
erage.day. > range_WL] <- NA
  #Month.of.absence has month 0 which is not possible as month ranges from 1-12
Absenteeism_Data_complete$Month.of.absence[Absenteeism_Data_complete$Month.of.absence
== 0] <- NA

sum(is.na(Absenteeism_Data_complete))

#Imputing NA's
imp.Absenteeism_Data_complete <- mice(Absenteeism_Data_complete, m=5, maxit = 50, seed =
500)
imp.Absenteeism_Data_complete$imp$Month.of.absence
#stripplot(imp.Absenteeism_Data_complete, pch = 20, cex = 1.2)
Absenteeism_Data_complete_OA <- complete(imp.Absenteeism_Data_complete, 1)

#Visualizing to check for utliers after oulier treatment from some continuous variables
plot_outlier(Absenteeism_Data_complete_OA)

#We have replaced outliers for some variables with NA's then we replaced them with imputed
values.
#Apart from that all values which are showing as outlier are accepted for our business purpose in the
dataset.
#We are not removing outliers from target variables as we don't know exactly wether these values
are outlier or not.

#########################################Feature
Selection############################################
## Correlation Plot
library(corrgram )
library(VIF)
library(usdm)
corrgram(Absenteeism_Data_complete_OA[ ,c(5:10,17:20)], order = F,
        upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")

#Check for multicollinearity using VIF
vifcor(Absenteeism_Data_complete_OA[ ,c(5:10,17:20)])
#Removing Weight variable as a part of our feature selection
Absenteeism_Data_complete_OA_FS = subset(Absenteeism_Data_complete_OA, select = -Weight)
#Checking VIF after removing Weight Column
vifcor(Absenteeism_Data_complete_OA_FS[ ,c(5:10,17:19)])

###################################Feature
Scaling################################################
```

```
#Scaling Continuous variables by z-score standardization technique
Absenteeism_Data_complete_OA_FS[ ,c(5:10,17:18)] <-
as.data.frame(scale(Absenteeism_Data_complete_OA_FS[ ,c(5:10,17:18)]))


#Saving the Complete clean pre-processed data
library(xlsx)
write.xlsx(Absenteeism_Data_complete_OA_FS, file = "Absenteeism_Clean_Data_new.xlsx",
sheetName="Sheet1", col.names=TRUE, row.names=F, append=FALSE)




################################################ Model
Developement#######################################################
emp_abs <- read.xlsx("Absenteeism_Clean_Data_new.xlsx",sheetName="Sheet1")
str(emp_abs)
# Creating dummy variables for categorical variables
library(mlr)
library(dummies)
df_0 = dummy.data.frame(emp_abs, categorical_vars,sep = ".")
#Divide the data into train and test
set.seed(123)
train_index = sample(1:nrow(df_0), 0.8 * nrow(df_0))
train = df_0[train_index,]
test = df_0[-train_index,]

####################################################Decision
Tree######################################################

#    mae       rmse       mse
# 5.658996  11.717496 137.299703

#Load Libraries
library(rpart)
library(DMwR)

# ##rpart for regression
fit = rpart(Absenteeism.time.in.hours ~ ., data = train, method = "anova")
#Predict for new test cases
predictions_dt = predict(fit, test[,-79])
#Error metric evaluation
regr.eval(test[ , 79], predictions_dt, stats = c('mae', 'rmse','mse'))
plot(predictions_dt - test[ , 79])

###############################################Random
Forest###################################################

#   mae       rmse       mse
# 4.706784  10.819635 117.064493

library(randomForest)
library(inTrees)
```

```
set.seed(321)
###Random Forest
RF_model = randomForest(Absenteeism.time.in.hours ~ ., train, importance = TRUE, ntree = 500)
#Extract rules fromn random forest
#transform rf object to an inTrees' format
treeList = RF2List(RF_model)

# #Extract rules
exec = extractRules(treeList, train[,-79])  # R-executable conditions

# #Visualize some rules
exec[1:2,]

# #Make rules more readable:
readableRules = presentRules(exec, colnames(train))
readableRules[1:2,]

# #Get rule metrics
ruleMetric = getRuleMetric(exec, train[,-79], train$Absenteeism.time.in.hours)  # get rule metrics

# #evaulate few rules
ruleMetric[1:2,]

#Presdict test data using random forest model
RF_Predictions = predict(RF_model, test[,-79])
#Error metric evaluation
regr.eval(test[ , 79], RF_Predictions, stats = c('mae', 'rmse','mse'))
plot(RF_Predictions - test[ , 79])

############################################XG
Boost##########################################################
#    mae       rmse        mse
# 4.815342  10.635997 112.124428

# Fitting model
library(caret)
set.seed(444)
TrainControl <- trainControl( method = "repeatedcv", number = 10, repeats = 4)
##Train the model using training data
model_xboost_1 <- train(Absenteeism.time.in.hours ~ ., data = train, method = "xgbTree", trControl
= TrainControl,verbose = FALSE)
#Predict the test cases
predicted_xgb_1 <- predict(model_xboost_1, test[, -79])
regr.eval(test[ , 79], predicted_xgb_1, stats = c('mae', 'rmse', 'mse'))
plot(predicted_xgb_1 - test[ , 79])


################################################Linear
Regression############################

#   mae      rmse       mse
# 5.47771  13.38506 179.15995

##Train the model using training data
lr_model = lm(formula = Absenteeism.time.in.hours~., data = train)
```

```
#Get the summary of the model
summary(lr_model)

#Predict the test cases
lr_predictions = predict(lr_model, test[,-79])
#Error metric evaluation
regr.eval(test[ , 79], lr_predictions, stats = c('mae', 'rmse', 'mse'))
plot(lr_predictions - test[ , 79])
```

References

1. For Data Cleaning and Model Development - https://edwisor.com/career-data-scientist
2. For PCA - https://www.analyticsvidhya.com/blog/2016/03/practical-guide-principal-component-analysis-python/
3. For Visualization – Analytics Vidya, Youtube, Udemy, Stackoveflow,