

Employee Absenteeism

*Prakash B*

*05 May 2019*

# Contents

|   |    |
|---|----|
| 1. Introduction                                 |    |
| 1.1 Problem Statement. ....                     | 3  |
| 1.2 Data. ....                                  | 3  |
| 2. Methodology                                  |    |
| 2.1 Pre – Processing. ....                      | 6  |
| 2.2 Missing Value Analysis. ....                | 6  |
| 2.3 Distribution of Continuous variables ....   | 10 |
| 2.4 Distribution of Categorical variables. .... | 13 |
| 2.5 Outlier Analysis. ....                      | 14 |
| 2.6 Feature Selection. ....                     | 18 |
| 2.7 Feature Scaling. ....                       | 19 |
| 2.8 Principal Component Analysis (PCA) ....     | 19 |
| 3. Modelling                                    |    |
| 3.1 Model Selection. ....                       | 20 |
| 3.2 Decision Tree. ....                         | 20 |
| 3.3 Random Forest. ....                         | 20 |
| 3.4 Linear Regression. ....                     | 21 |
| 4. Conclusion                                   |    |
| 4.1 Model Evaluation. ....                      | 22 |
| 4.2 Model Selection. ....                       | 22 |
| 4.3 Solution of Problem Statement. ....         | 22 |
| 5. Appendix                                     |    |
| 5.1 Figures. ....                               | 27 |
| 6. R code. ....                                 | 36 |
| Python code ....                                | 44 |

# Chapter 1: Introduction

## 1.1 Problem Statement

XYZ is a courier company. As we appreciate that human capital plays an important role in collection, transportation and delivery. The company is passing through genuine issue of Absenteeism. The company has shared its dataset and requested to have an answer on the following areas:

1. What changes company should bring to reduce the number of absenteeism?
2. How much losses every month can we project in 2011 if same trend of absenteeism continues?

## 1.2 Data

There are 21 variables in our data in which 20 are independent variables and 1 (Absenteeism time in hours) is dependent variable. Since the type of target variable is continuous, this is a regression problem.

Variable Information:

1. Individual identification (ID)

2. Reason for absence (ICD).

- Absences attested by the International Code of Diseases (ICD) stratified into 21 categories (I to XXI) as follows:

I certain infectious and parasitic diseases

II Neoplasms

III Diseases of the blood and blood-forming organs and certain disorders involving the immune mechanism

IV Endocrine, nutritional and metabolic

diseases V Mental and behavioural disorders

VI Diseases of the nervous system

VII Diseases of the eye and adnexa

VIII Diseases of the ear and mastoid process

IX Diseases of the circulatory system

X Diseases of the respiratory system

XI Diseases of the digestive system

XII Diseases of the skin and subcutaneous tissue

XIII Diseases of the musculoskeletal system and connective tissue

XIV Diseases of the genitourinary system

XV Pregnancy, childbirth and the puerperium

XVI Certain conditions originating in the perinatal period

XVII Congenital malformations, deformations and chromosomal abnormalities

XVIII Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified

XIX Injury, poisoning and certain other consequences of external causes

XX External causes of morbidity and mortality

XXI Factors influencing health status and contact with health services.

And 7 categories without (CID) patient follow-up (22), medical consultation (23), blood donation (24), laboratory examination (25), unjustified absence (26), physiotherapy (27), dental consultation (28).

3. Month of absence

4. Day of the week (Monday (2), Tuesday (3), Wednesday (4), Thursday (5), Friday (6))

5. Seasons (summer (1), autumn (2), winter (3), spring (4))

6. Transportation expense

7. Distance from Residence to Work (KMs)

8. Service time

9. Age

10. Work load Average/day

11. Hit target

12. Disciplinary failure (yes=1; no=0)

13. Education (high school (1), graduate (2), postgraduate (3), master and doctor (4))

14. Son (number of children)

15. Social drinker (yes=1; no=0)

16. Social smoker (yes=1; no=0)

17. Pet (number of pet)

18. Weight

19. Height

20. Body mass index

21. Absenteeism time in hours (target)

Below is the sample data from the dataset,

| ▲ | ID | Reason.for.absence | Month.of.absence | Day.of.the.week | Seasons | Transportation.expense | Distance.from.Residence.to.Work | Service.time | Age |
|---|----|--------------------|------------------|-----------------|---------|------------------------|---------------------------------|--------------|-----|
| 1 | 11 |                    | 26 Jul           | Tue             | summer  | 289                    | 36                              | 13           | 33  |
| 2 | 36 |                    | 0 Jul            | Tue             | summer  | 118                    | 13                              | 18           | 50  |
| 3 | 3  |                    | 23 Jul           | Wed             | summer  | 179                    | 51                              | 18           | 38  |
| 4 | 7  |                    | 7 Jul            | Thu             | summer  | 279                    | 5                               | 14           | 39  |
| 5 | 11 |                    | 23 Jul           | Thu             | summer  | 289                    | 36                              | 13           | 33  |

| Work.load.Average.day | Hit.target | Disciplinary.failure | Education   | Son | Social.drinker | Social.smoker | Pet  | Weight | Height | Body.mass.index |
|-----------------------|------------|----------------------|-------------|-----|----------------|---------------|------|--------|--------|-----------------|
| 239554                | 97         | no                   | high school | two | yes            | no            | one  | 90     | 172    | 30              |
| 239554                | 97         | yes                  | high school | one | yes            | no            | zero | 98     | 178    | 31              |
| 239554                | 97         | no                   | high school | NA  | yes            | no            | zero | 89     | 170    | 31              |
| 239554                | 97         | no                   | high school | two | yes            | yes           | zero | 68     | 168    | 24              |
| 239554                | 97         | no                   | high school | two | yes            | no            | one  | 90     | 172    | 30              |

| Absenteeism.time.in.hours |
|---------------------------|
| 4                         |
| 0                         |
| 2                         |
| 4                         |
| 2                         |

**Fig 1.1 – First five rows of data**

As you can see in the table below, we have the following 21 variables, using which we have to correctly predict the below questions.

1. What changes company should bring to reduce the number of absenteeism (Absenteeism time in hours)?
2. How much losses every month can we project in 2011 if same trend of absenteeism continues?

**Fig 1.2 – Predictor Variables**

| Sr. No | Variables                       |
|--------|---------------------------------|
| 1      | ID                              |
| 2      | Reason for absence              |
| 3      | Month of absence                |
| 4      | Day of the week                 |
| 5      | Seasons                         |
| 6      | Transportation expense          |
| 7      | Distance from Residence to Work |
| 8      | Service time                    |
| 9      | Age                             |
| 10     | Work load Average/day           |
| 11     | Hit target                      |
| 12     | Disciplinary failure            |
| 13     | Education                       |
| 14     | Son                             |
| 15     | Social drinker                  |
| 16     | Social smoker                   |
| 17     | Pet                             |
| 18     | Weight                          |
| 19     | Height                          |
| 20     | Body mass index                 |

## Chapter 2: Methodology

### 2.1 Pre – Processing

Any predictive modelling requires that we look at the data before we start modelling. 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. To start this process, we will first try and look at all the probability distributions of the variables. Most analysis like regression, require the data to be normally distributed. We can visualize that in a glance by looking at the probability distributions or probability density functions of the variable.

### 2.2 Missing Value Analysis

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.

Missing value Percentage of top 3 Variables.

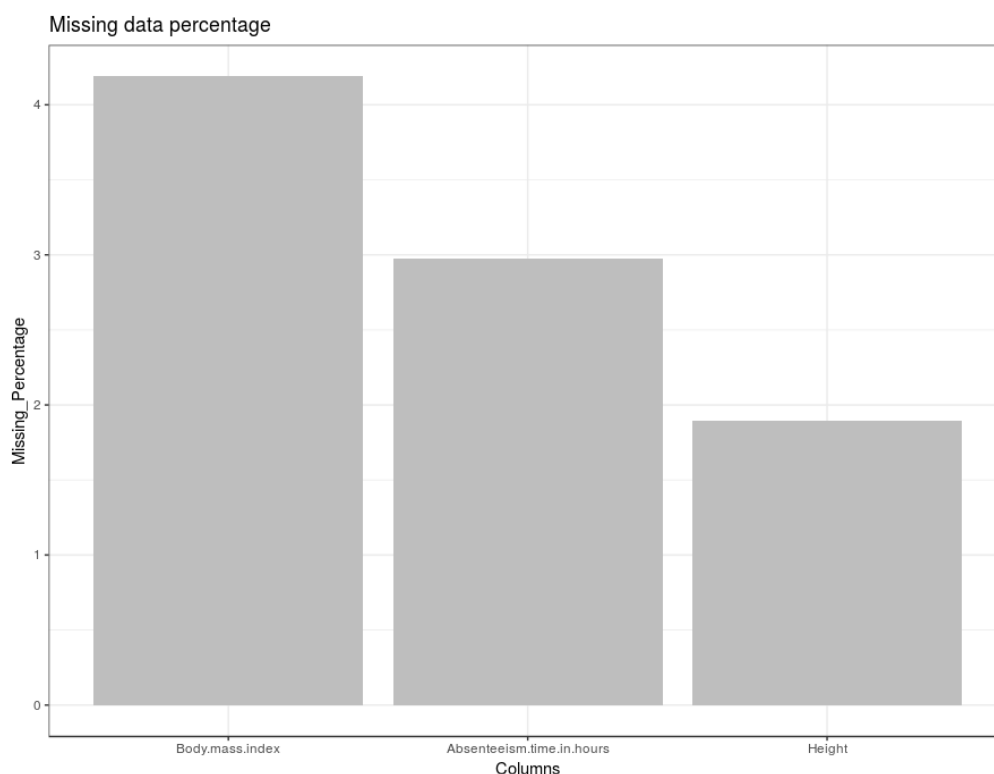
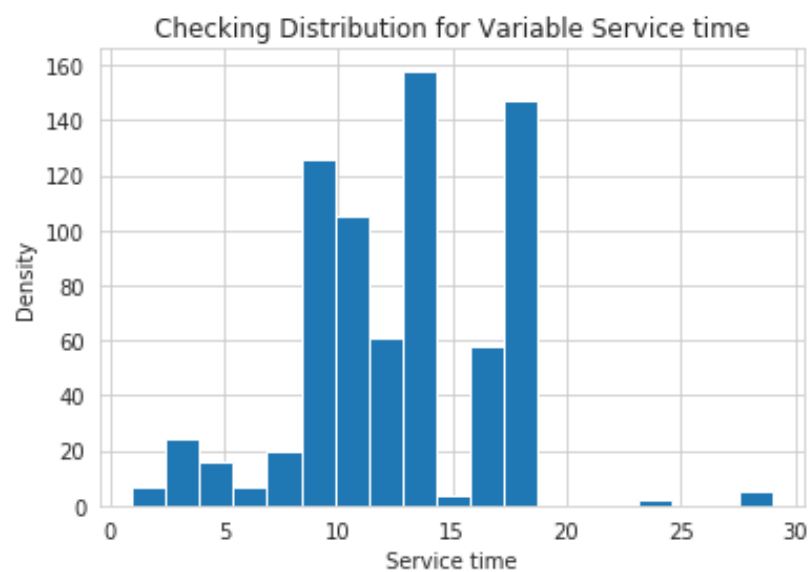
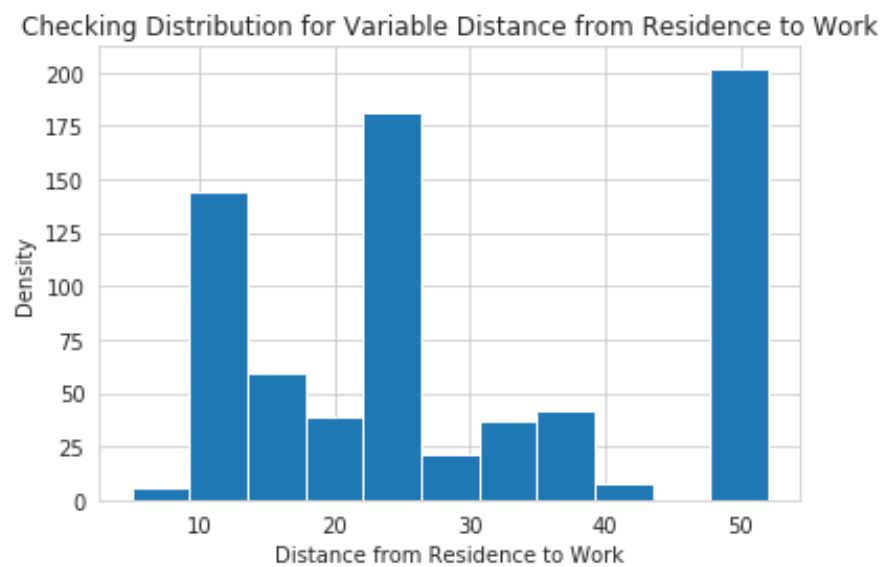
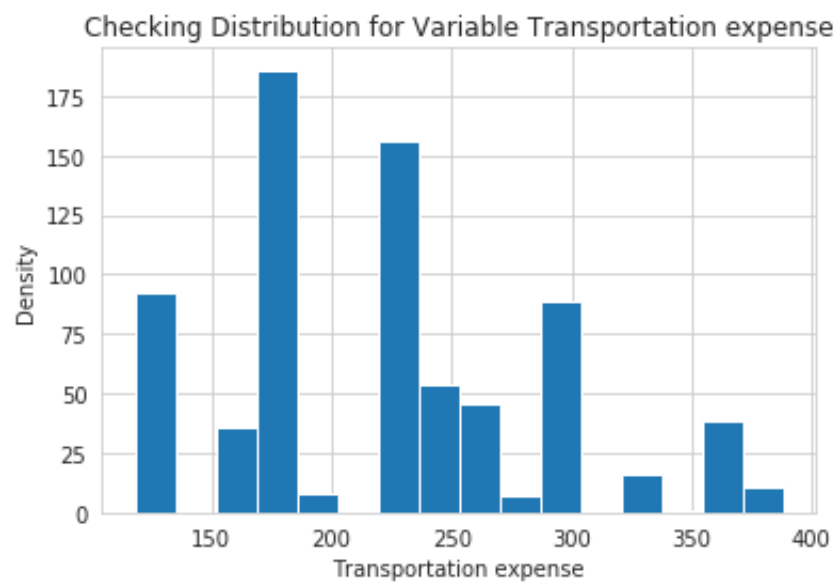
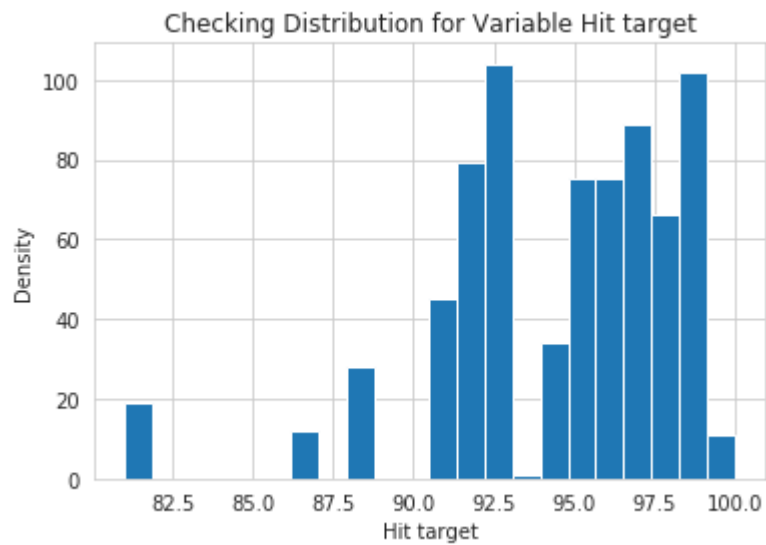
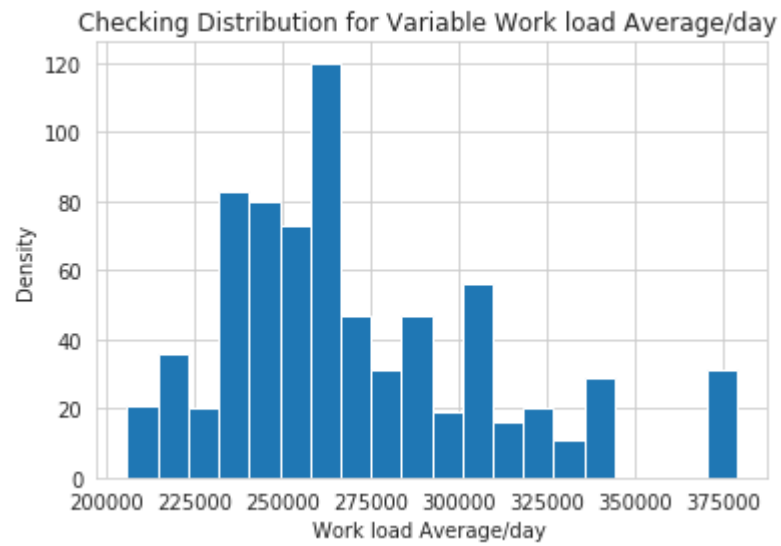
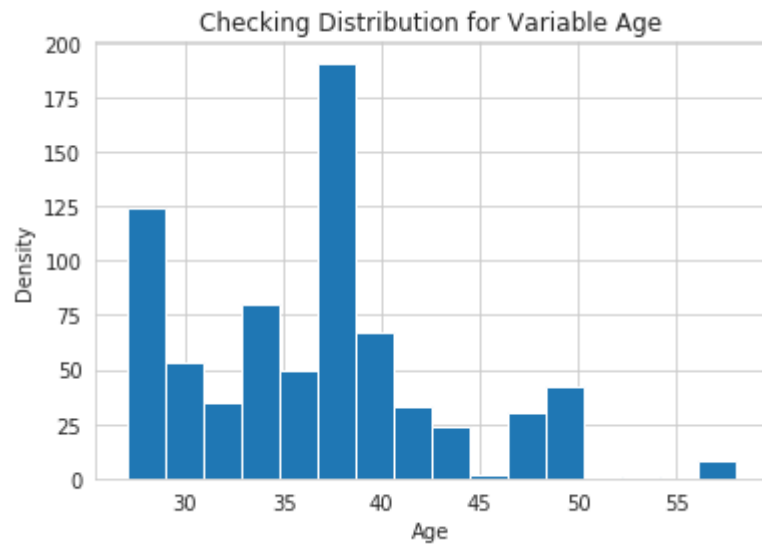


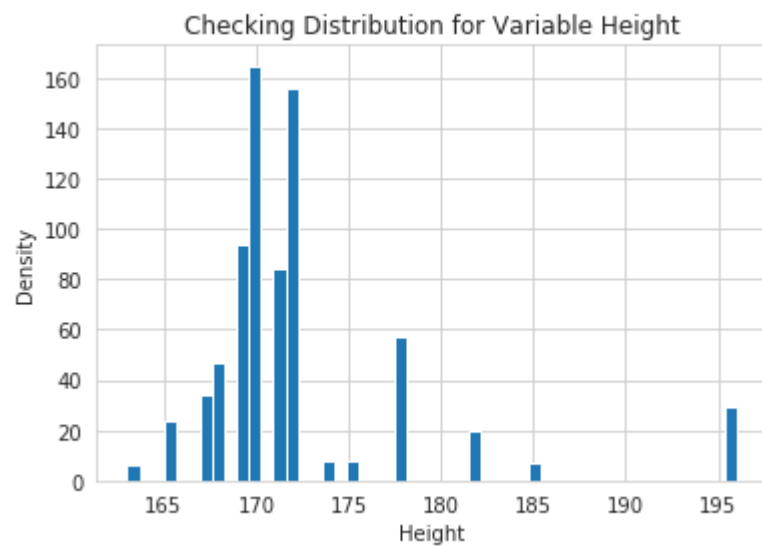
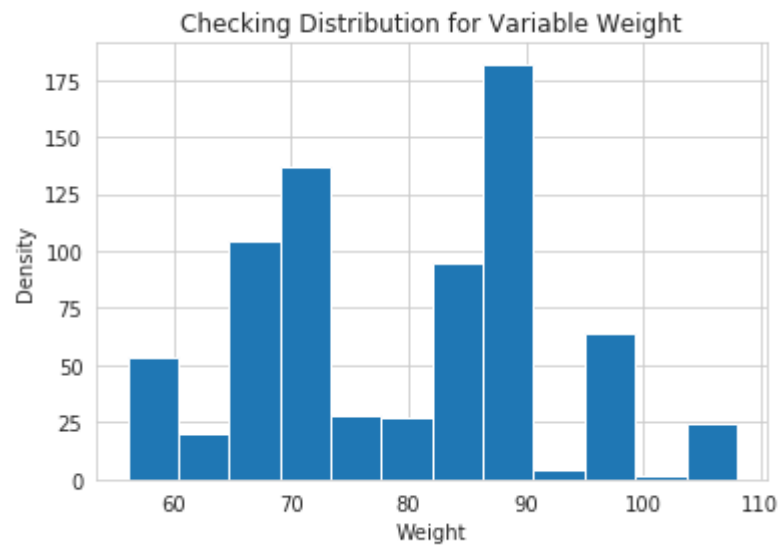
Fig 2.1 – Missing value Percentage

Below figure shows the Probability Distributions of Variables.









We can clearly observe from these probability distributions that most of the variables are skewed, for example, Height, Age, Transportation expense and Work load Average/day. The skew in these distributions can be most likely explained by the presence of outliers and extreme values in the data.

## 2.3 Distribution of Categorical Variables

Bar graphs are used to visualize the distribution of categorical variables.

Employees who are social drinkers have more absent hours than those who do not drink.

Employees having zero, one or two children have more absent hours.

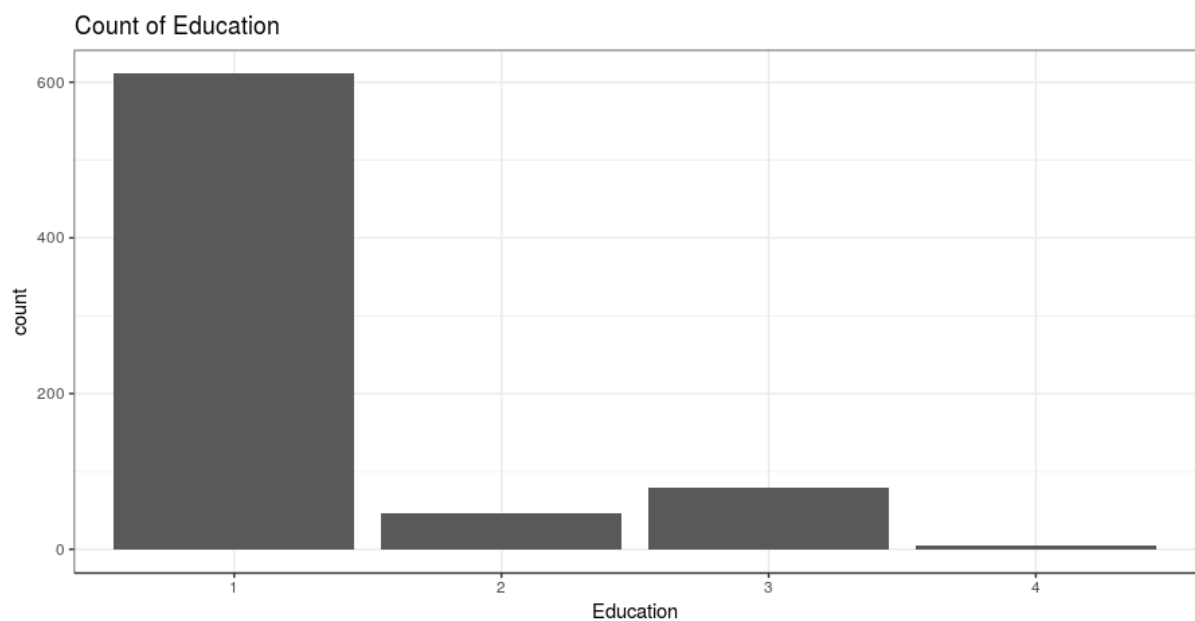
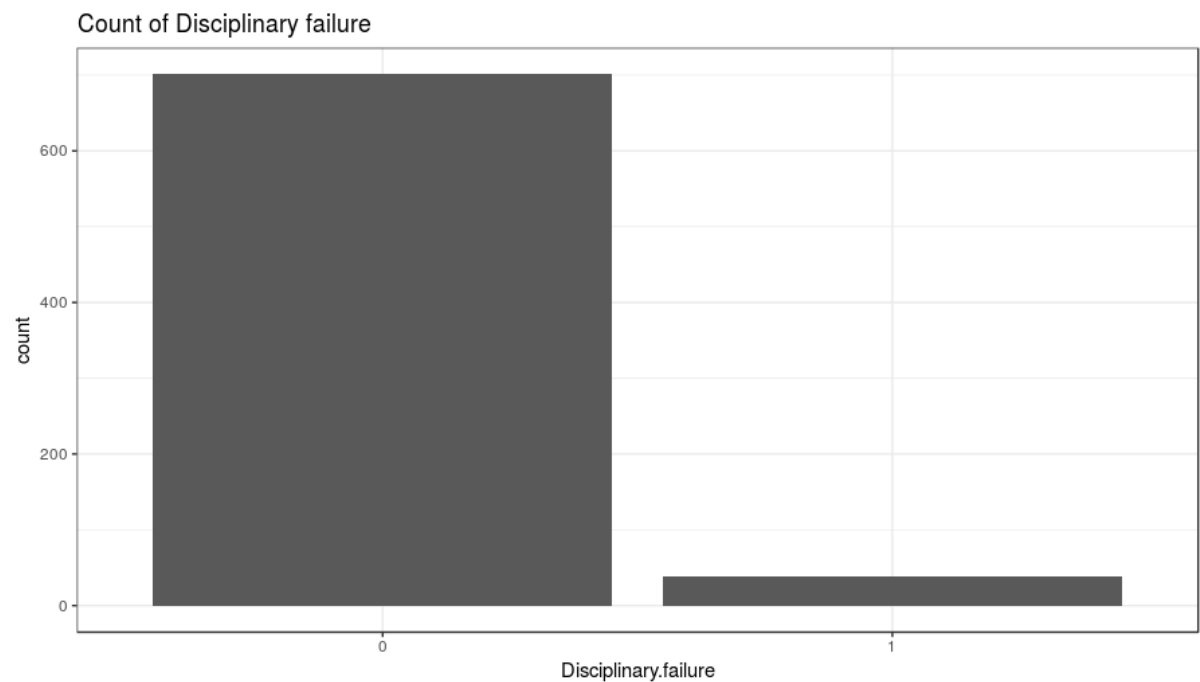
Employees with ID number 3 and 28 are absent the most.

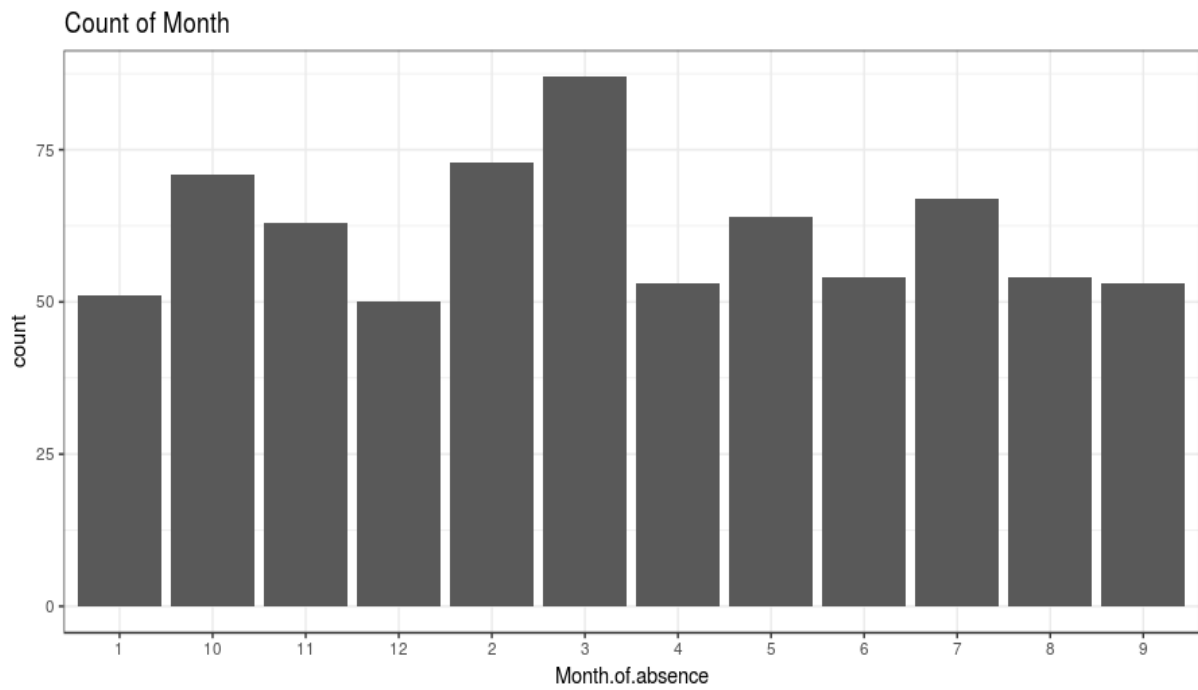
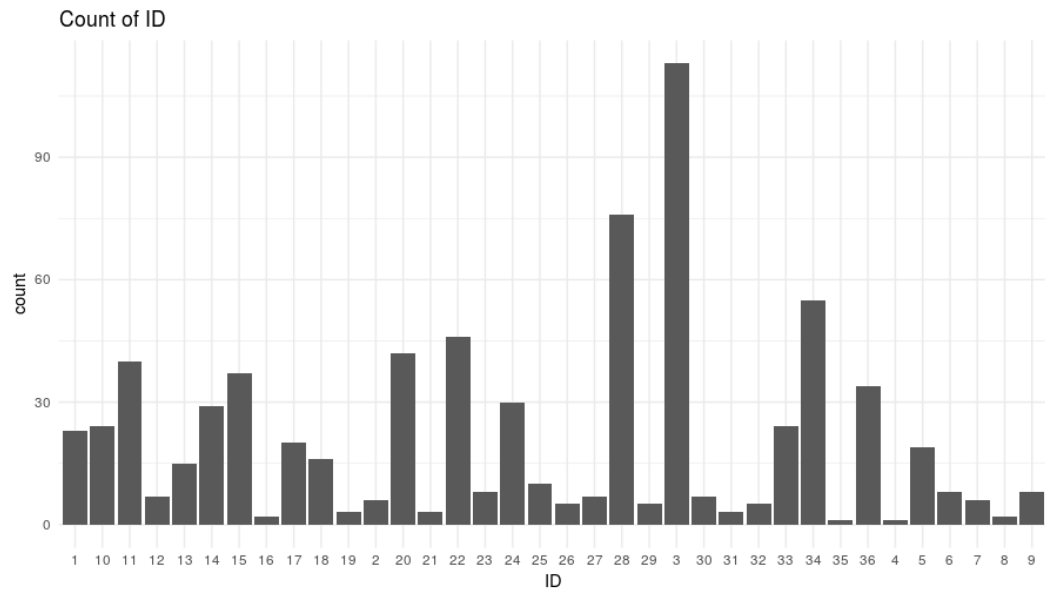
Employees are absent the most on Mondays and the least on Thursdays.

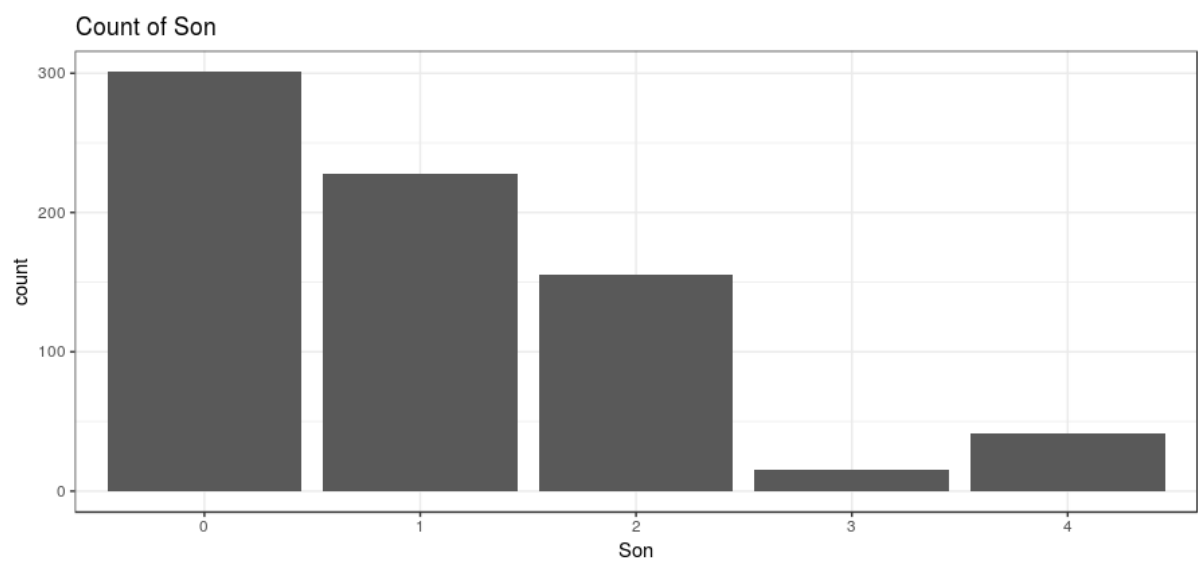
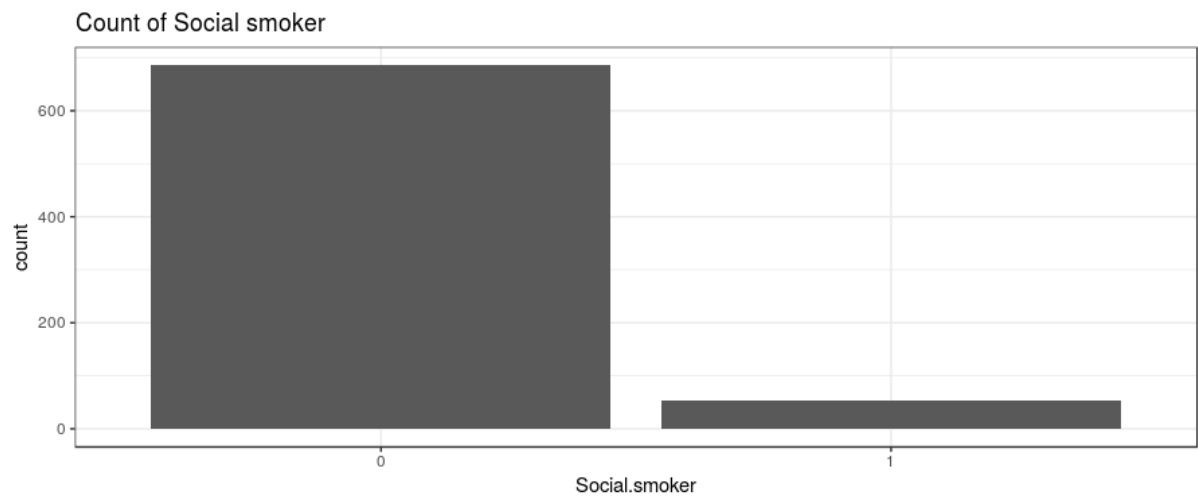
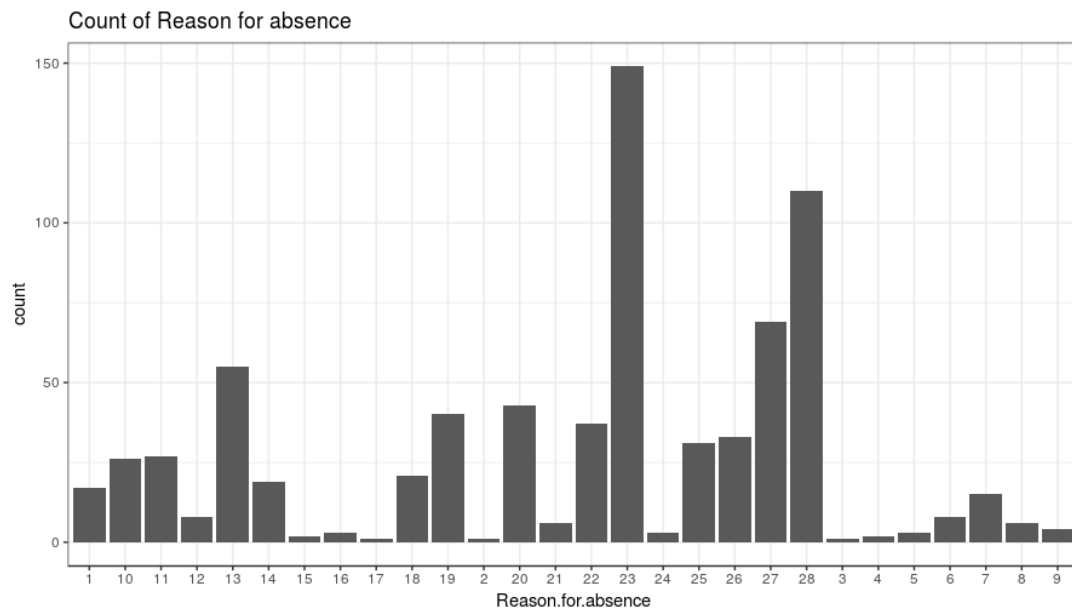
Reason 23 and 28 are the reasons employee give the most for being absent.

Employees who have completed only high school education are absent more than others.

Employees are absent the most in the month of March.

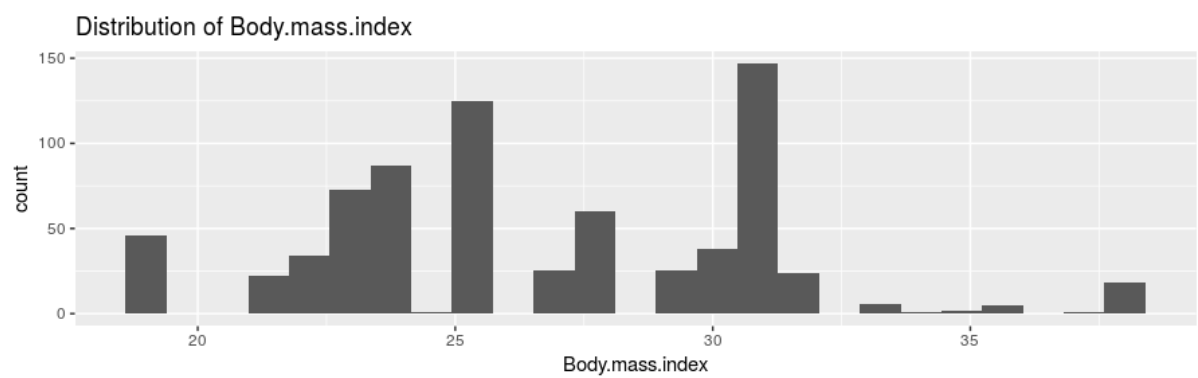
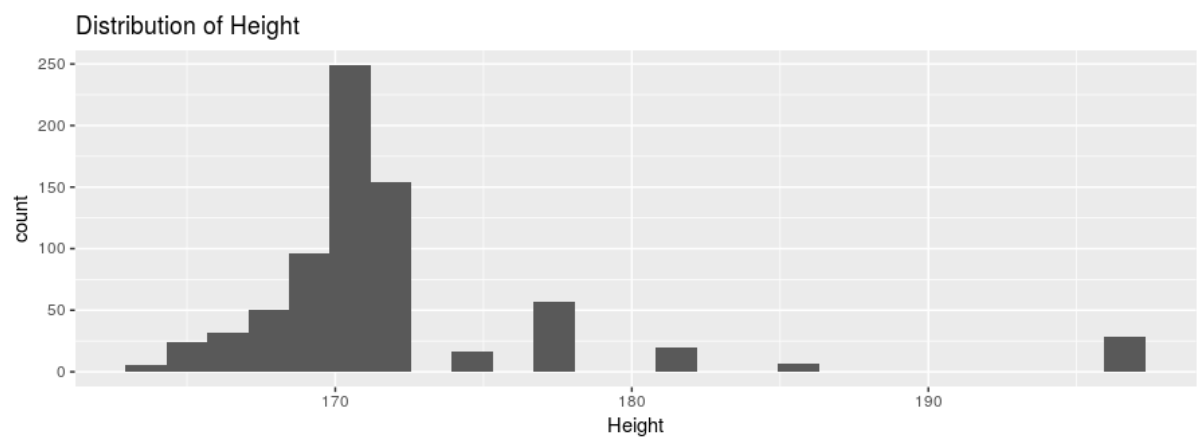
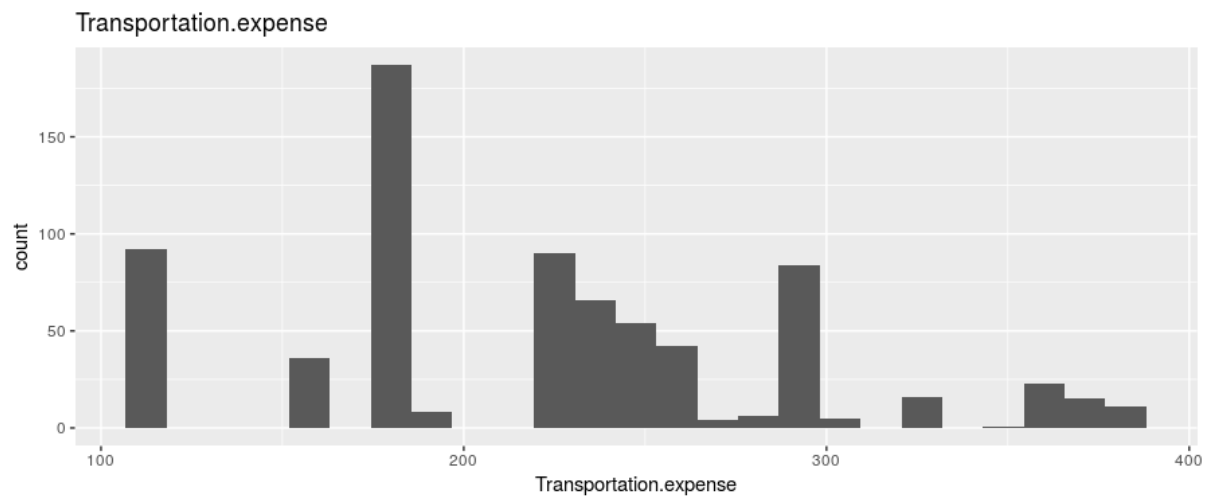


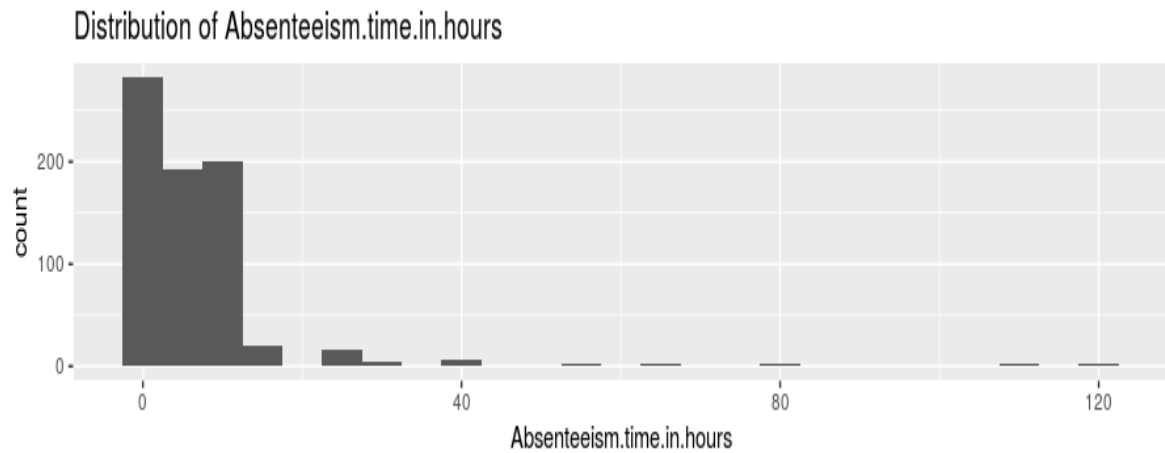




**Fig 2.2 – Distribution of Categorical variables using Bar graph**

## 2.4 Distribution of Continuous Variables



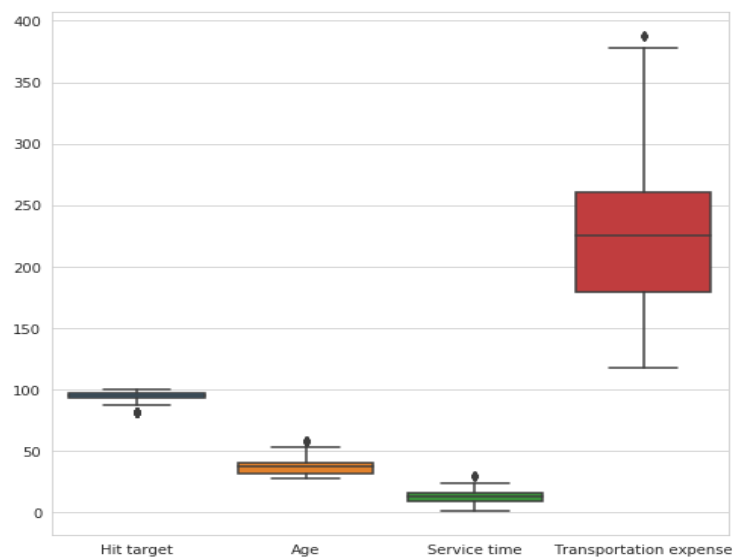


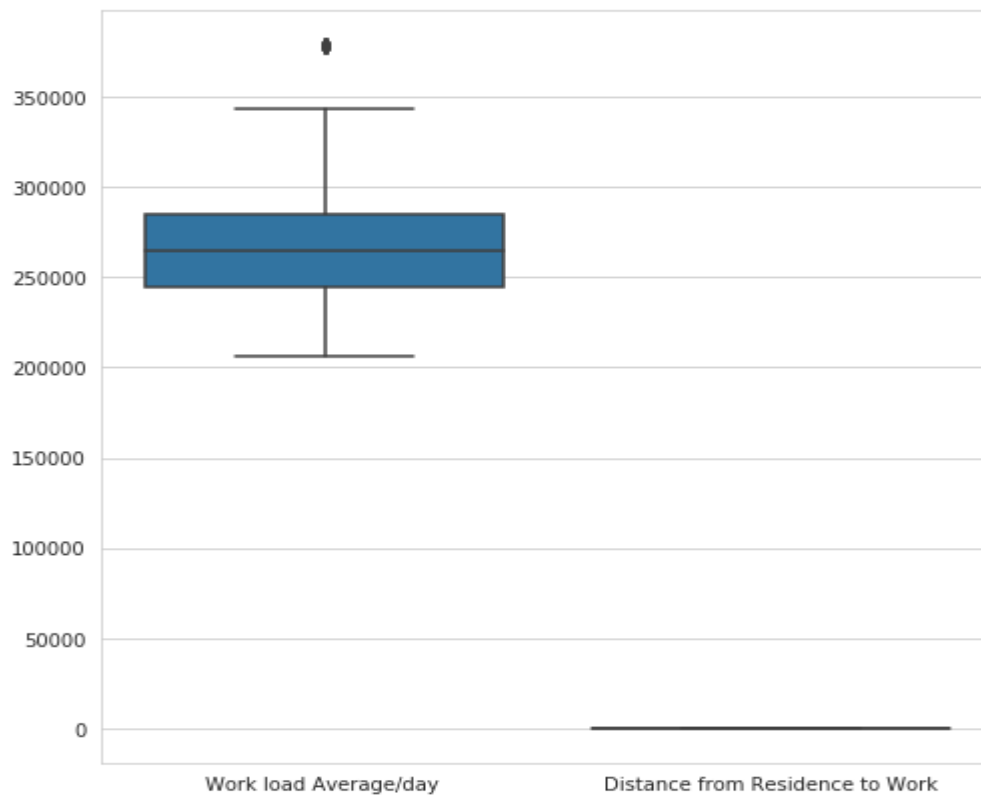
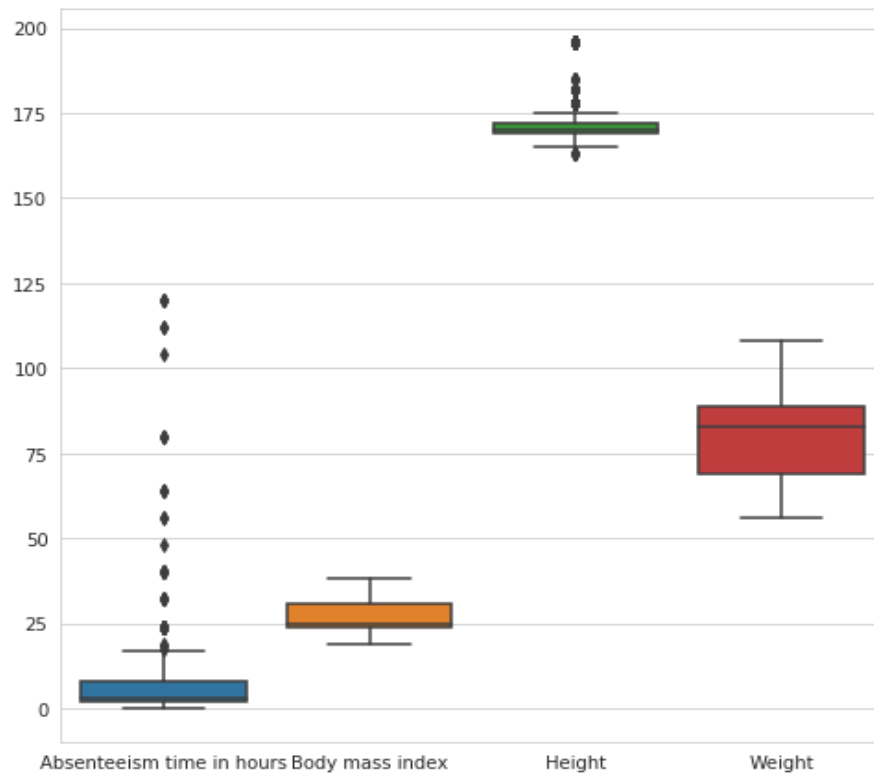
**Fig 2.3 – Distribution of Continuous variables using histograms**

## 2.5 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.

Variables excluding Distance from residence to work, Weight and Body mass index, contain outliers.



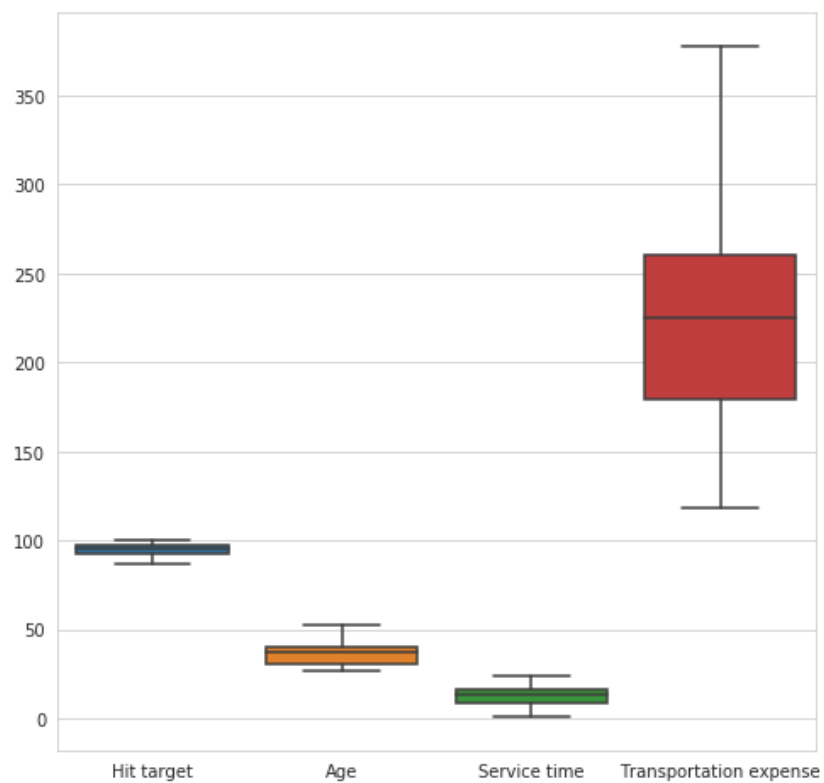
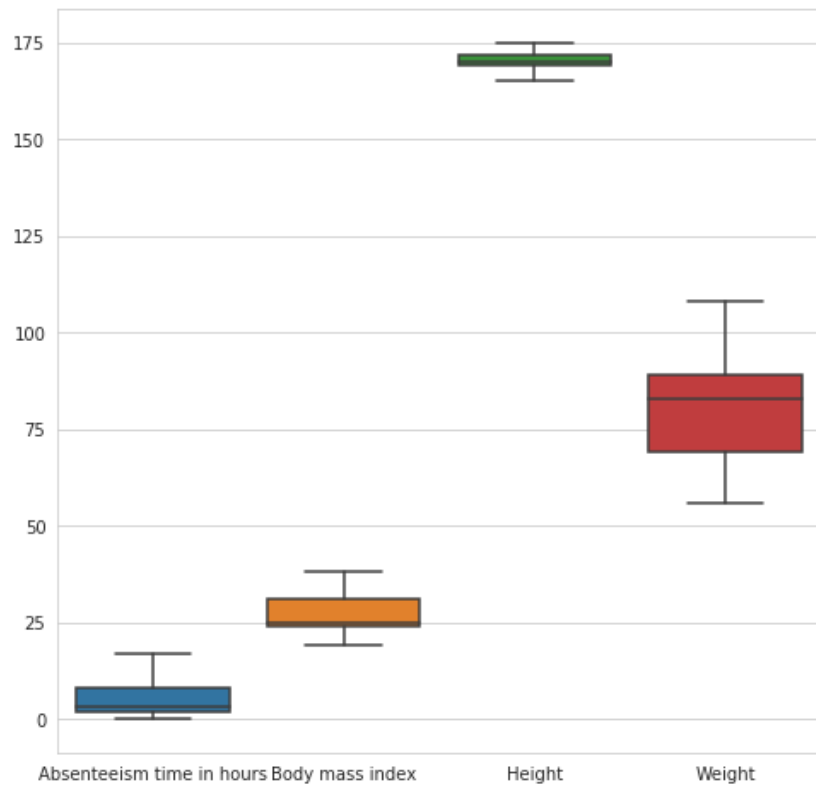


**Fig 2.4 – Boxplots of continuous variables with outliers**

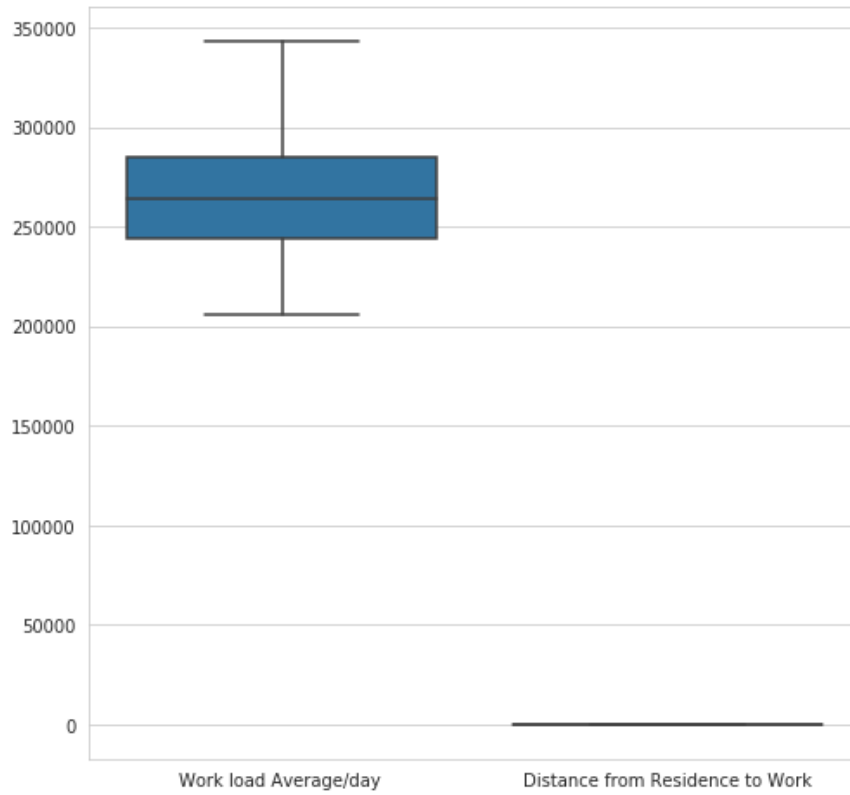
### Imputing outlier values:

Missing values obtained from boxplots are first converted to have NA values. Then these missing values are imputed using KNN imputation method.

Below figure shows the boxplots of variables after removing outliers.







**Fig 2.5 – Boxplots of continuous variables without outliers**

## 2.6 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.

From correlation analysis we have found that Weight and Body Mass Index has high correlation ( $>0.7$ ), so we have excluded the Body Mass Index column.

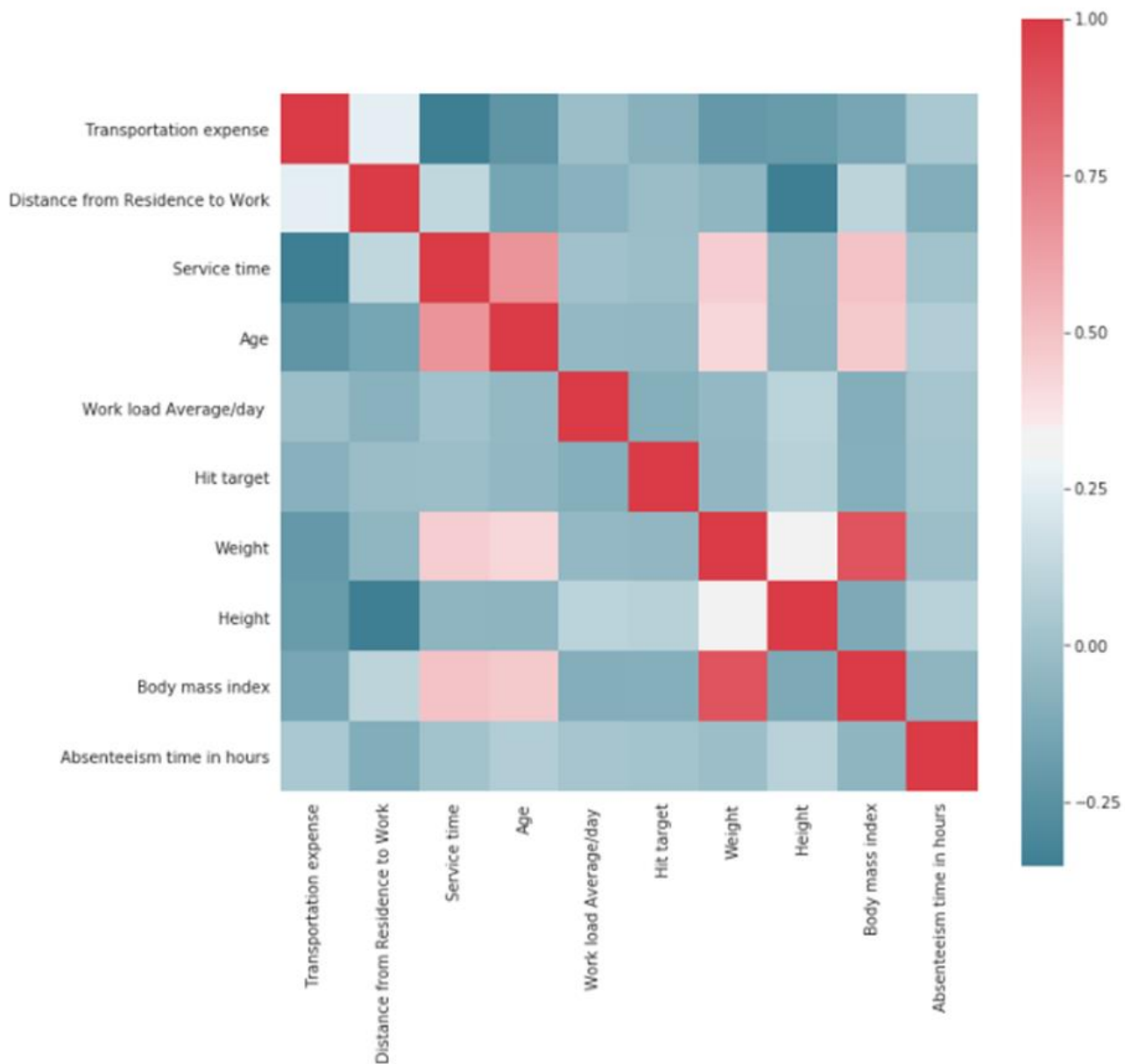


Fig 2.6 – Correlation plot of Continuous variables

## 2.7 Feature Scaling

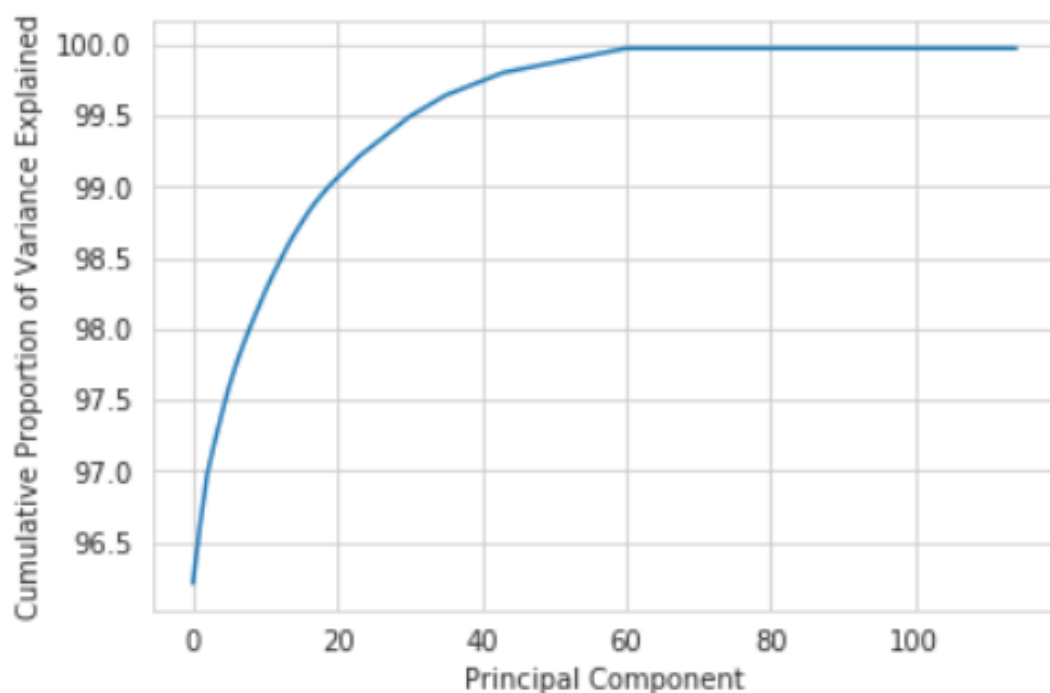
Feature scaling is a method used to standardize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data pre-processing step.

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 normalized so that each feature contributes proportionately to final distance. Since our data is not uniformly distributed, we will use Normalization as Feature Scaling Method.

## 2.8 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 116 columns and 740 observations. This high number of columns leads to bad accuracy.



**Fig 2.7 – Cumulative Graph of Principal Components**

After applying PCA algorithm and observing the above Cumulative graph, it can be observed that almost 95% of the data can be explained by 45 variables out of 116. 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 Linear Regression, Decision Tree and Random Forest. The error metric chosen for the given problem statement is Root Mean Square Error (RMSE).

### 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 continuous value). The general motive of using Decision Tree is to create a training model which can use to predict class or value of target variables by learning decision rules inferred from prior data (training data).

The RMSE values and  $R^2$  values for the given project in R and Python are:

| DECISION TREE | RMSE    | $R^2$  |
|---------------|---------|--------|
| R             | 0.4427  | 0.9787 |
| PYTHON        | 0.07939 | 0.9999 |

### 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. Assembling 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.

| RANDOM FOREST | RMSE   | $R^2$  |
|---------------|--------|--------|
| R             | 0.4361 | 0.9828 |
| PYTHON        | 0.0004 | 0.9999 |

### 3.4 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.

| LINEAR REGRESSION | RMSE   | R^2    |
|-------------------|--------|--------|
| R                 | 0.0030 | 0.9999 |
| PYTHON            | 0.0013 | 0.9999 |

## Chapter 4: Conclusion

### 4.1 Model Evaluation

In the previous chapter we have seen the Root Mean Square Error (RMSE) 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. Whereas 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.

### 4.2 Model Selection

From the observation of all RMSE Value and R-Squared Value we have concluded that **Linear Regression Model** has minimum value of RMSE, and its R-Squared Value is also maximum.

### 4.3 Solutions of Problem Statement

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

Solution:

- a. It can be observed that employees having education only till high school tend to be absent more than others. So, the company can either hire employees who have at least graduated from college or inform those employees who have completed only their high school education to reduce the number of hours they are absent.

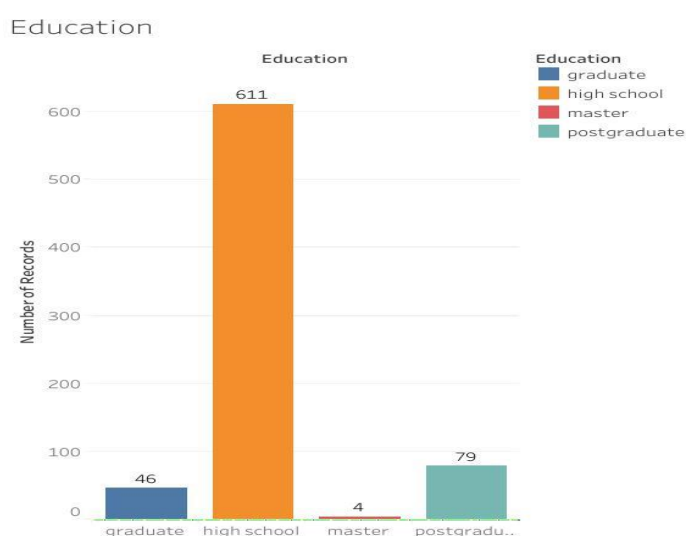
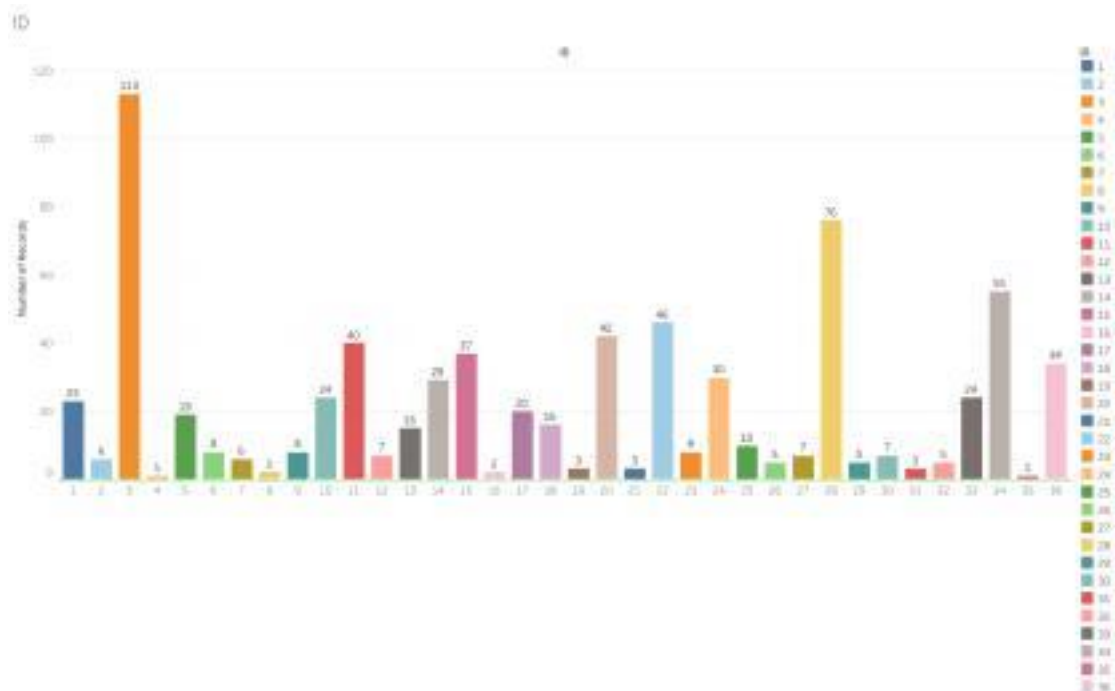


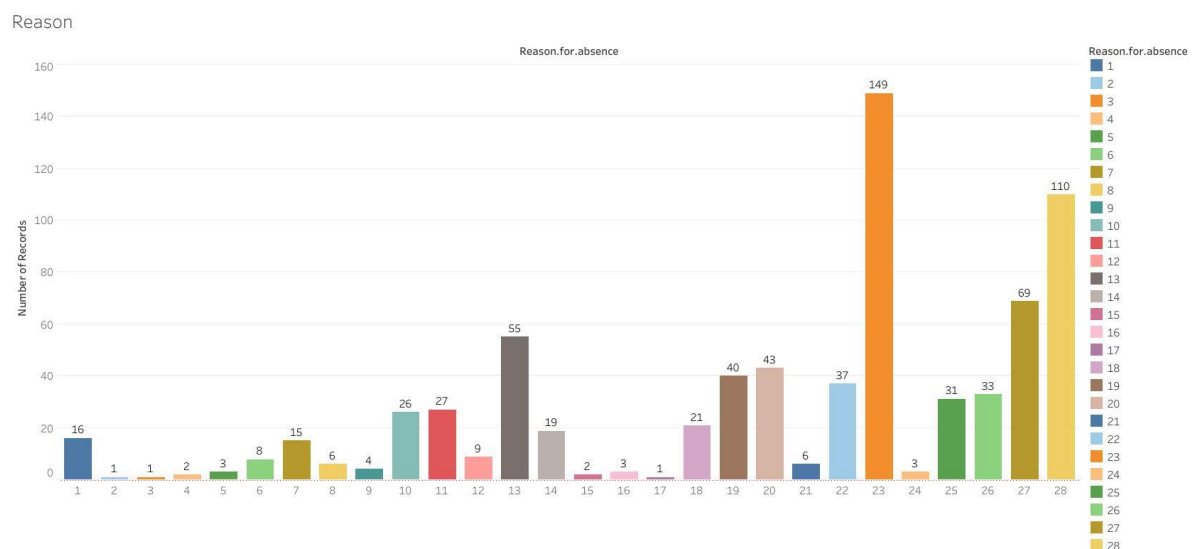
Fig 4.1 – Education vs Absent Hours

- b. Employees with ID 3, 28 and 34 are some of the employees who are absent the most. The company may act warn such employees to reduce being absent a lot or if repeated further, it can against them if necessary.



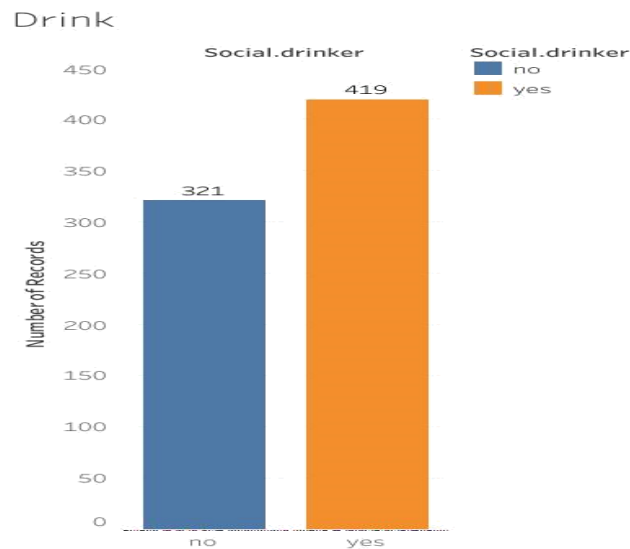
**Fig 4.2 – ID vs Absent Hours**

- c. The reasons most used by employees to be absent are reason 13, 20, 23 and 28. These reasons include Medical consultation, Dental appointments, morbidity, mortality and diseases of musculoskeletal system and connective tissue. The company XYZ can help in informing employees on how to keep themselves healthier by having monthly campus consultations.



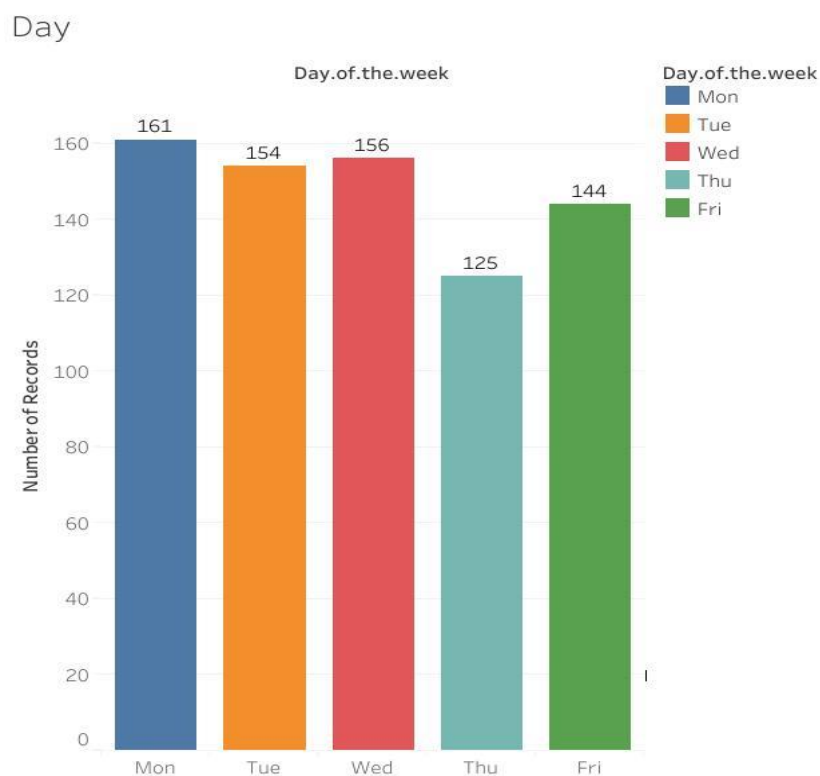
**Fig 4.3 – Reason of Absence vs Absent Hours**

- d. People who tend to be social drinkers tend to be more absent than those who don't drink. XYZ can keep a track of those people and inform those employees to reduce the intake of alcohol during working days.



**Fig 4.4 – Social Drinker vs Absent Hours**

- e. Employees are absent the most on Mondays with absent hours equal to 1426 and Tuesdays with absent hours equal to 1322.4. XYZ can inform employees to not take as many absent hours on such days.



**Fig 4.5 – Day of the Week vs Absent Hours**



f. Employees are mostly absent during Spring Season.

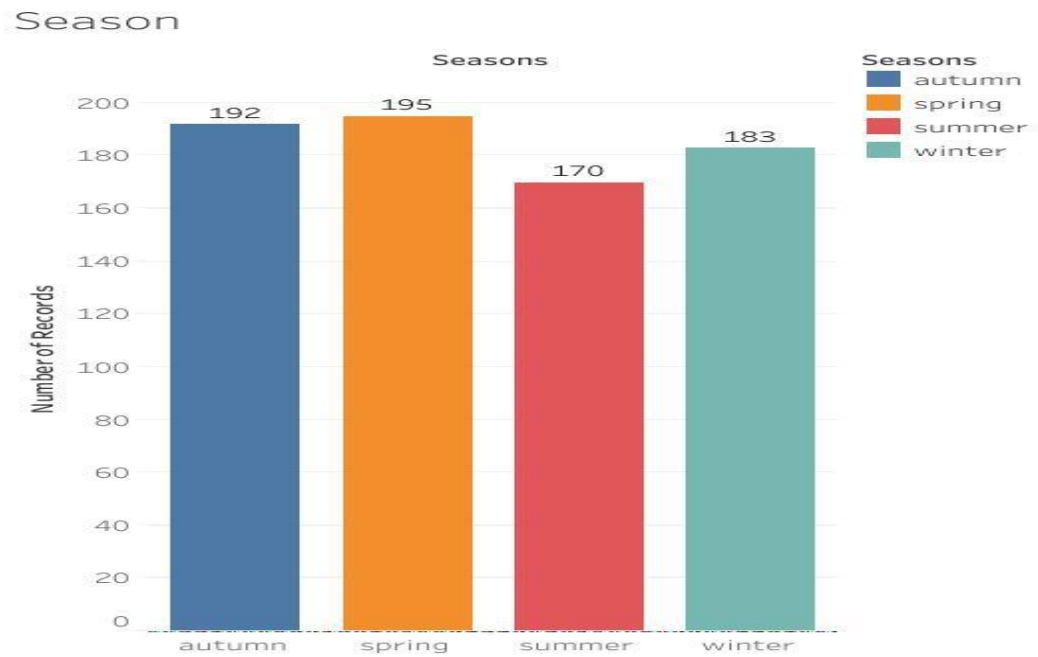


Fig 4.6 – Season vs Absent Hours

g. Employees having a maximum of two children or no child at all are absent the most.

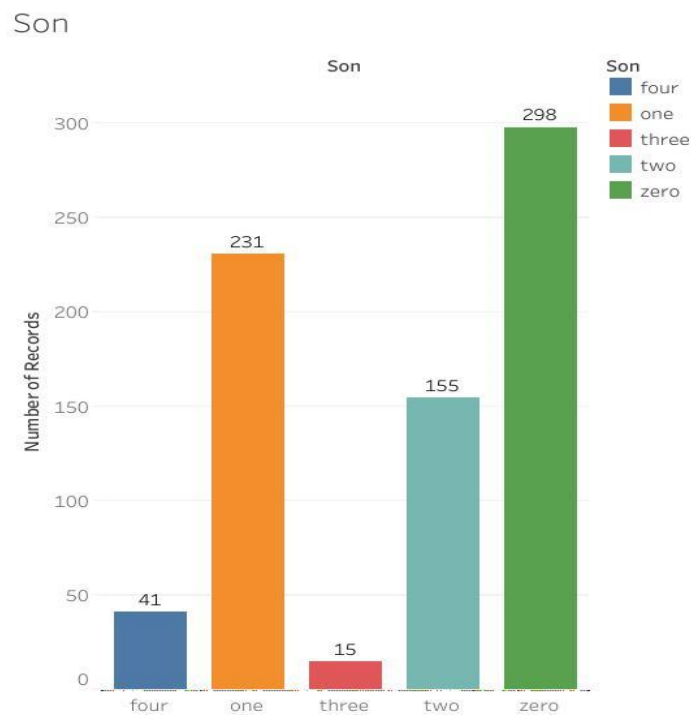


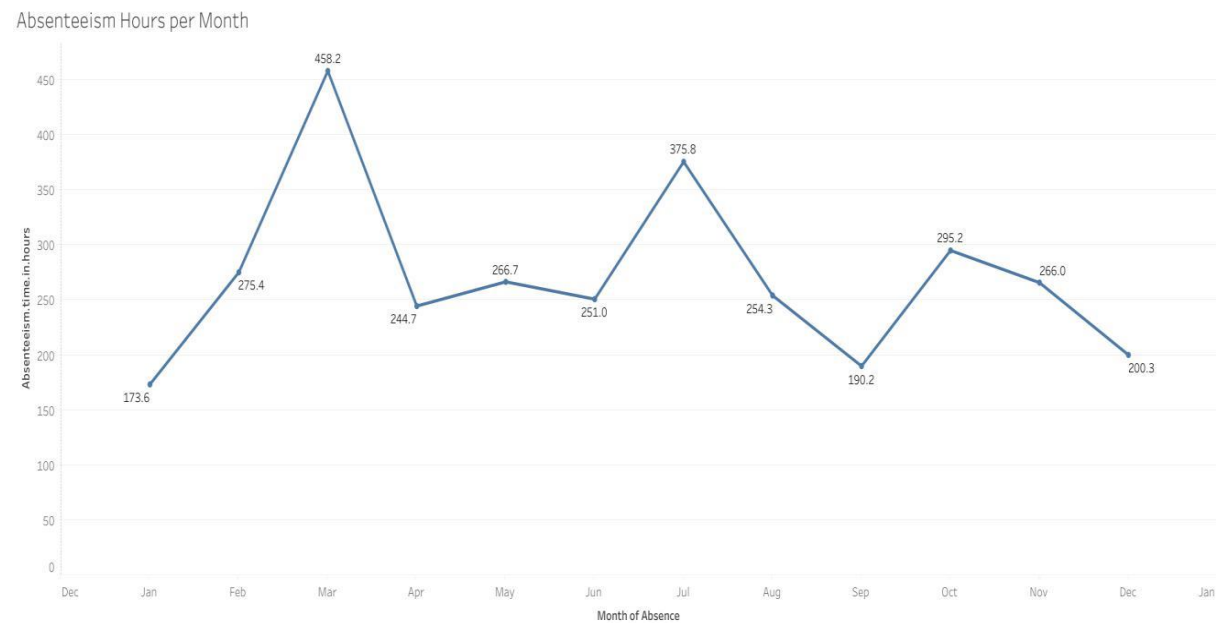
Fig: 4.7 – Sons vs Absent Hours

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 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 4.8 – 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: Appendix

### 5.1 Figures

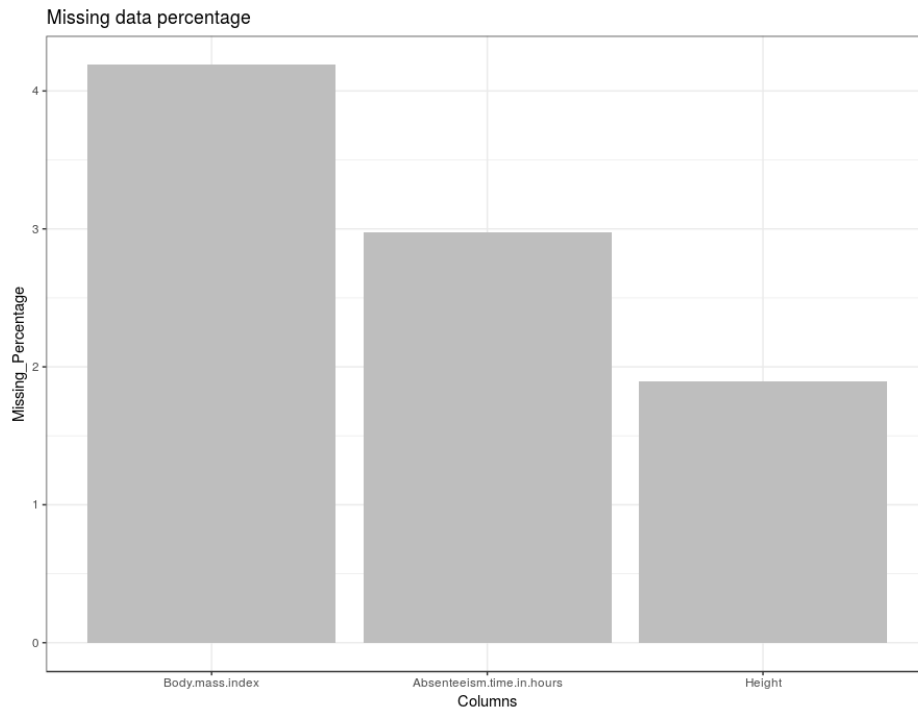
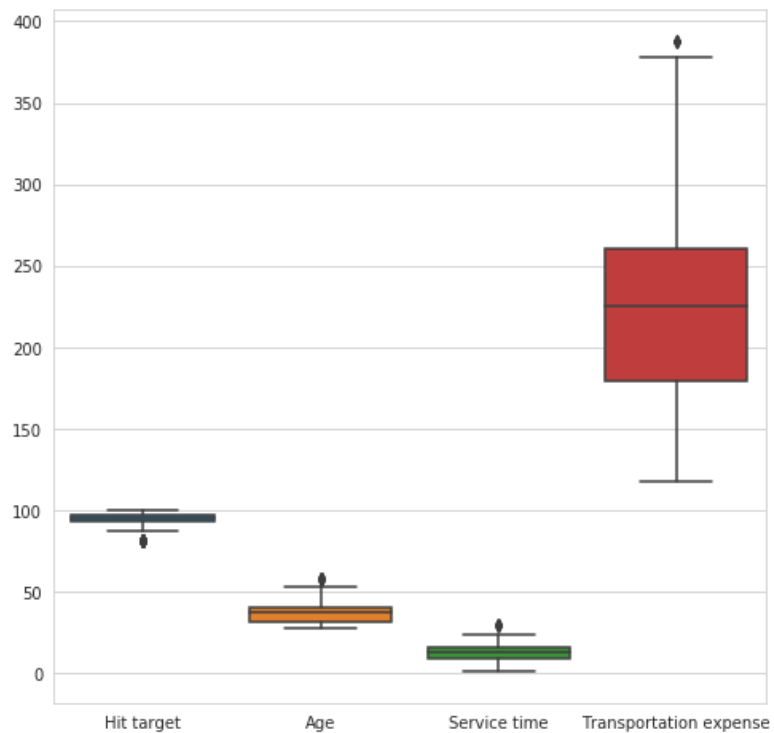
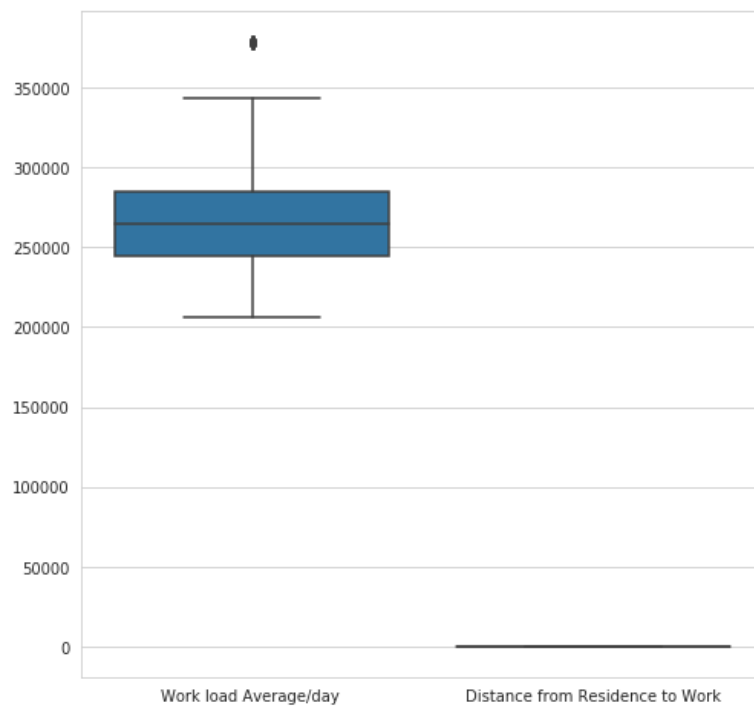
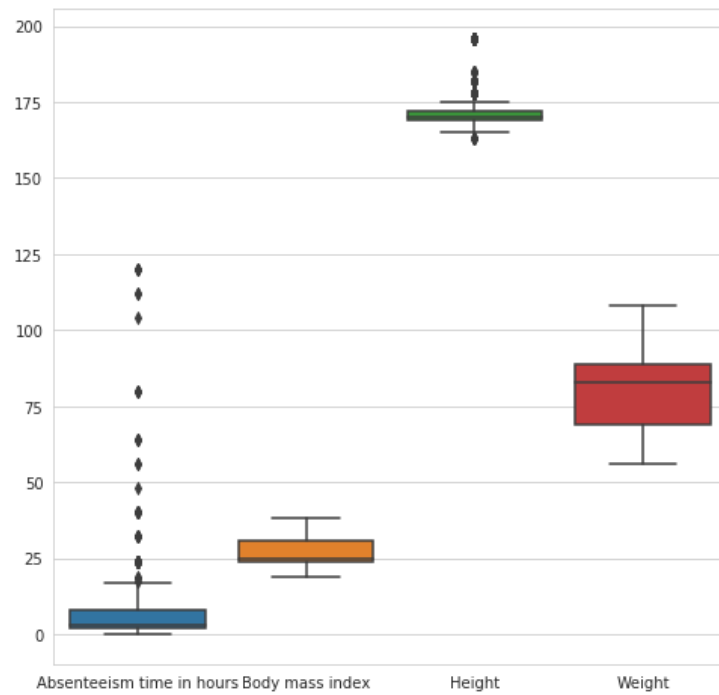
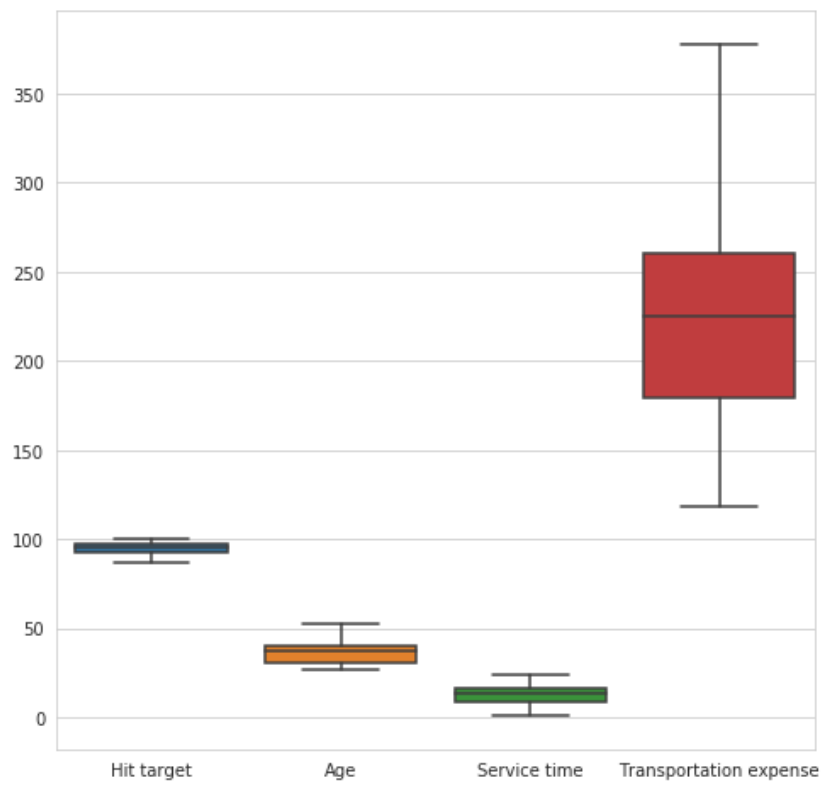
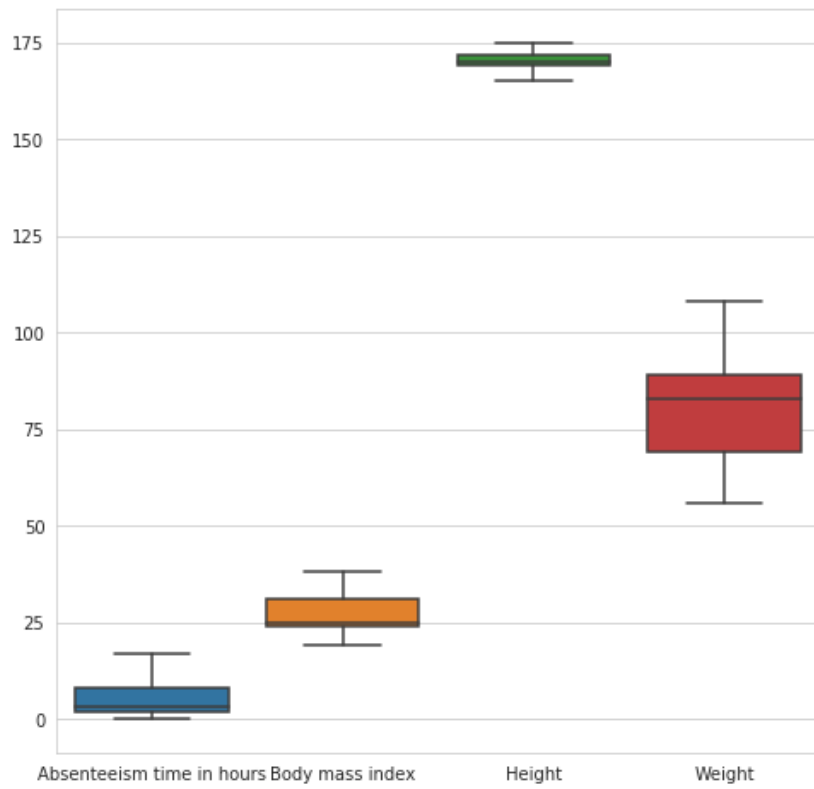


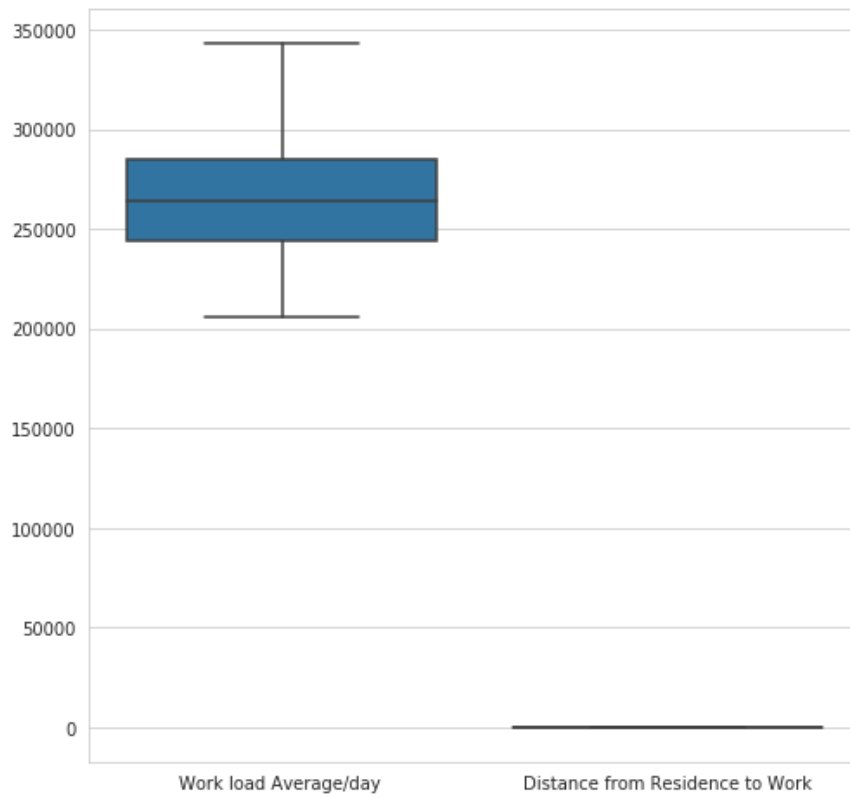
Fig 5.1 – Missing value Percentage



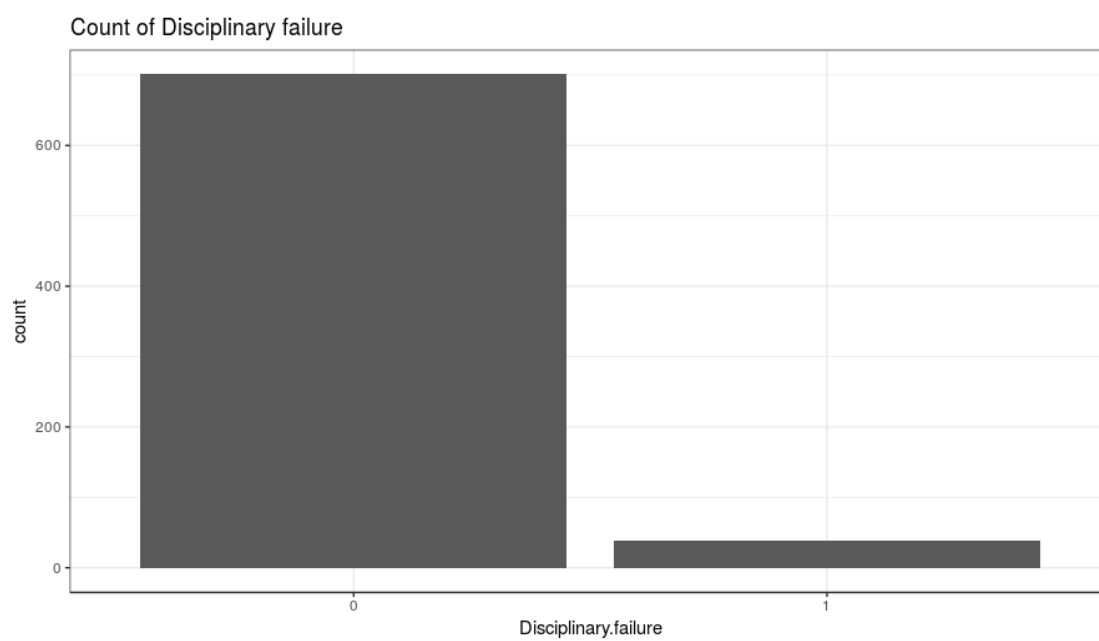


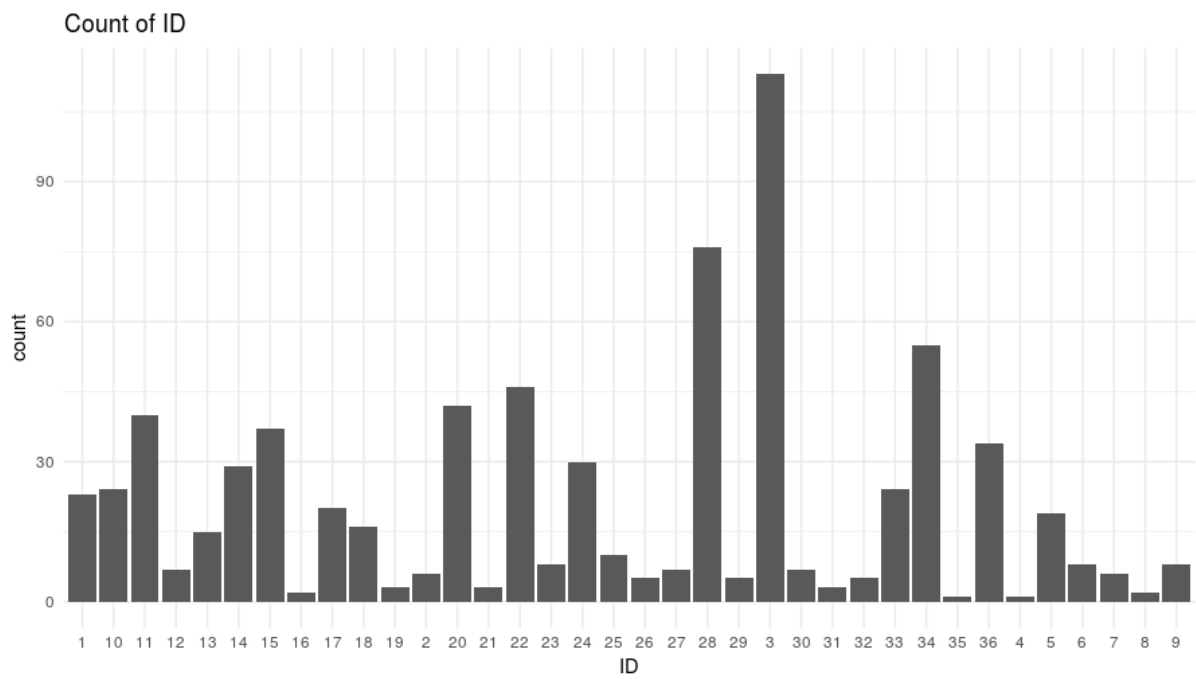
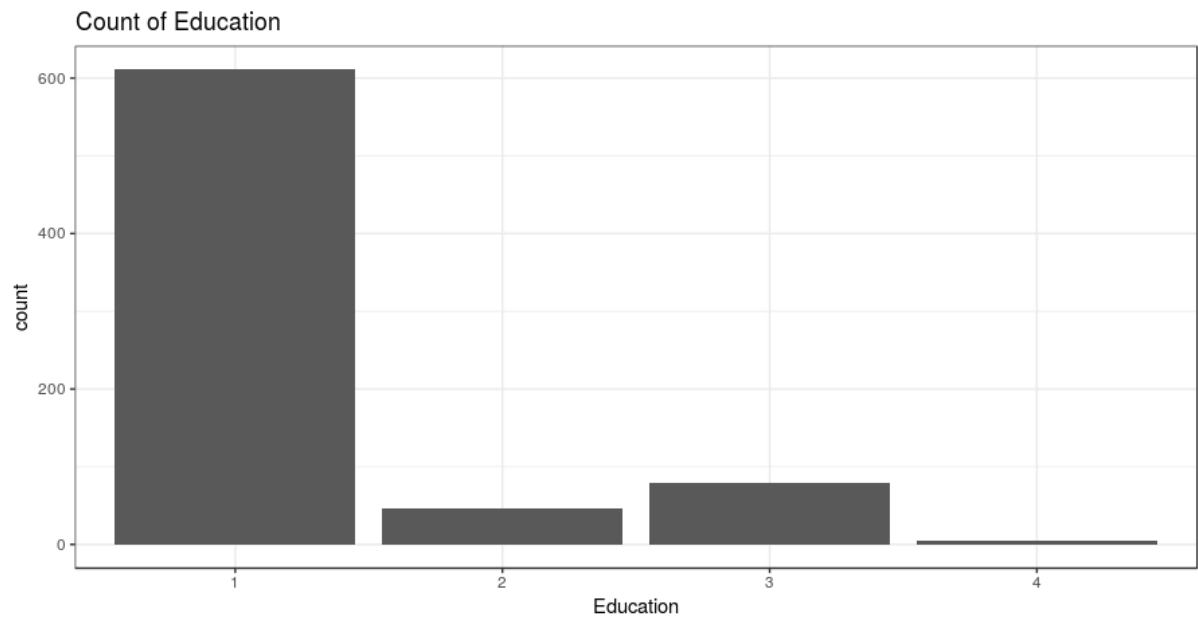
**Fig 5.2 – Boxplots of continuous variables with outliers**

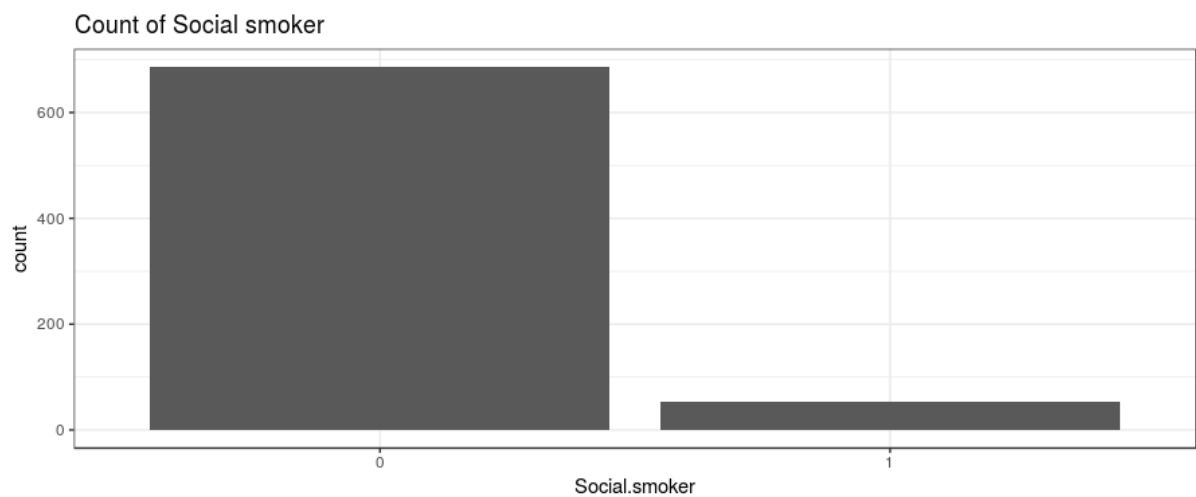
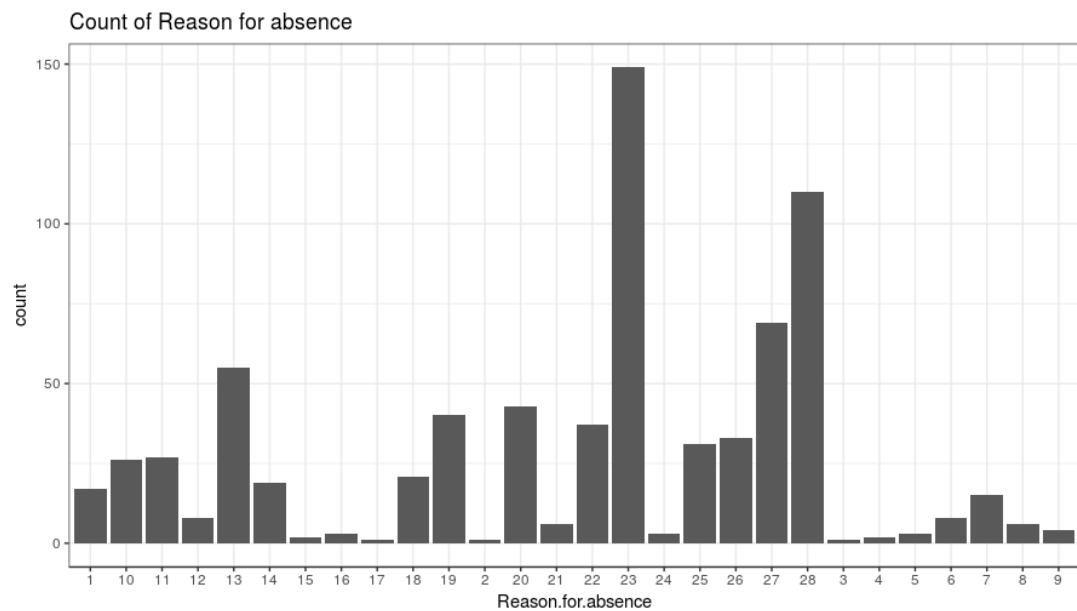
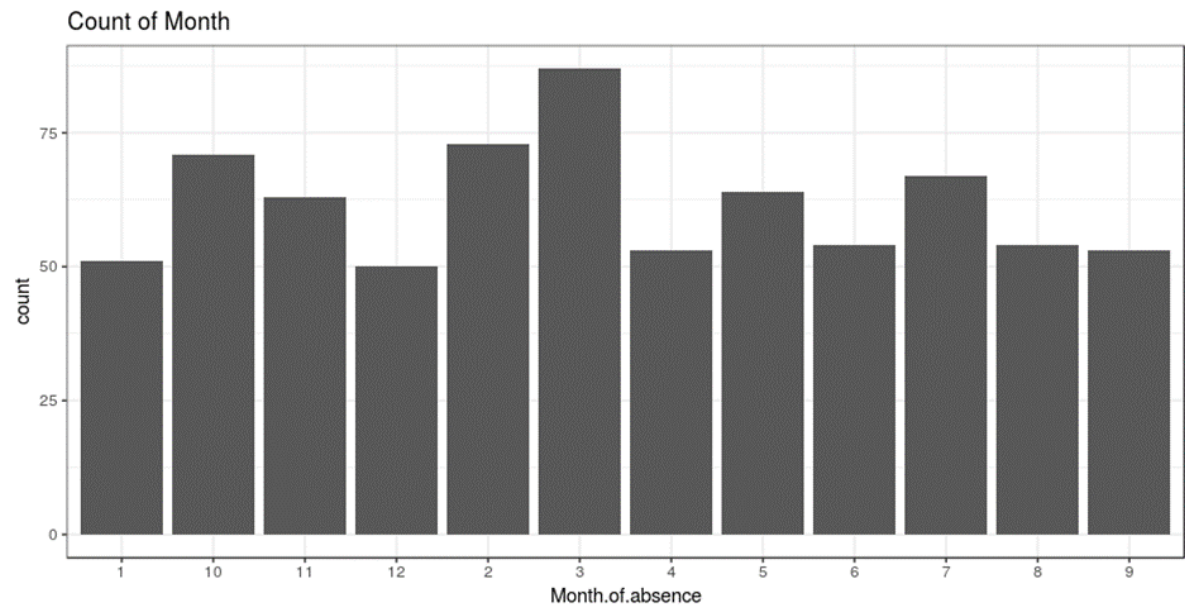




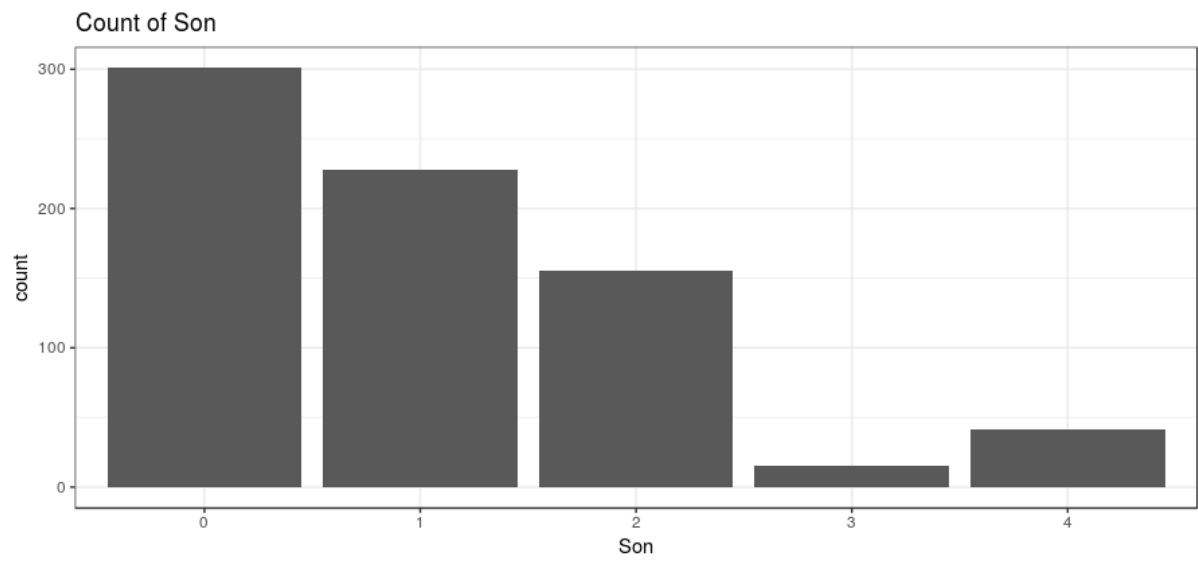
**Fig 5.3 – Boxplots of continuous variables without outliers**



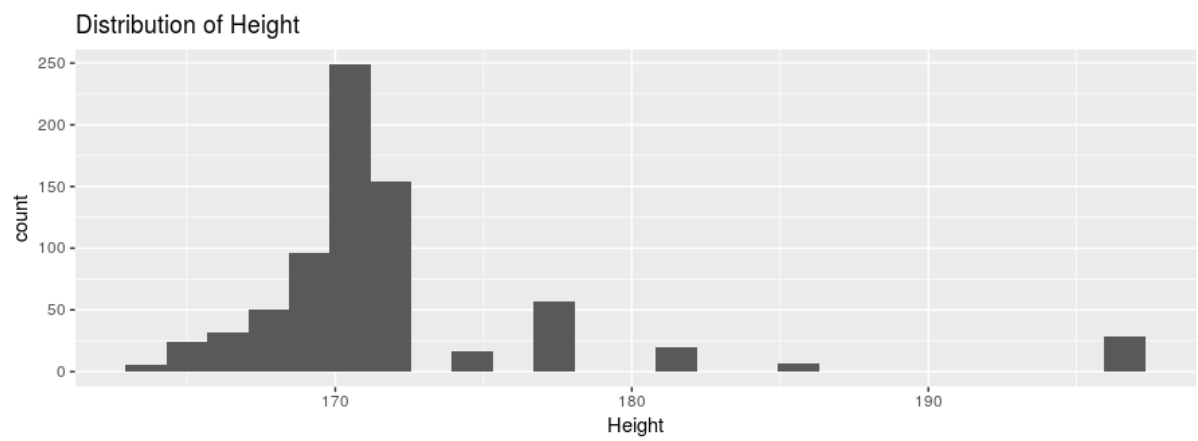
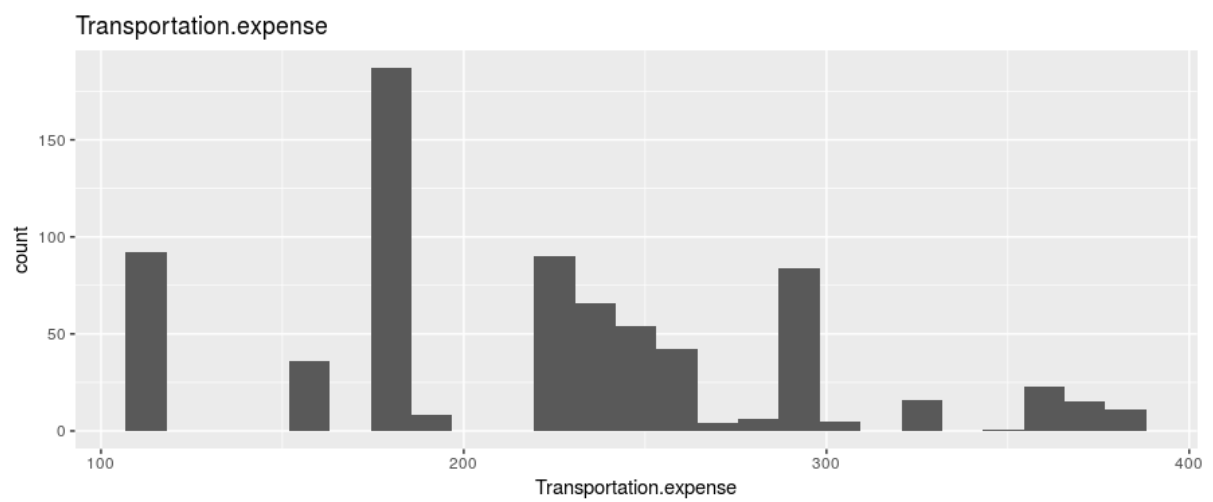


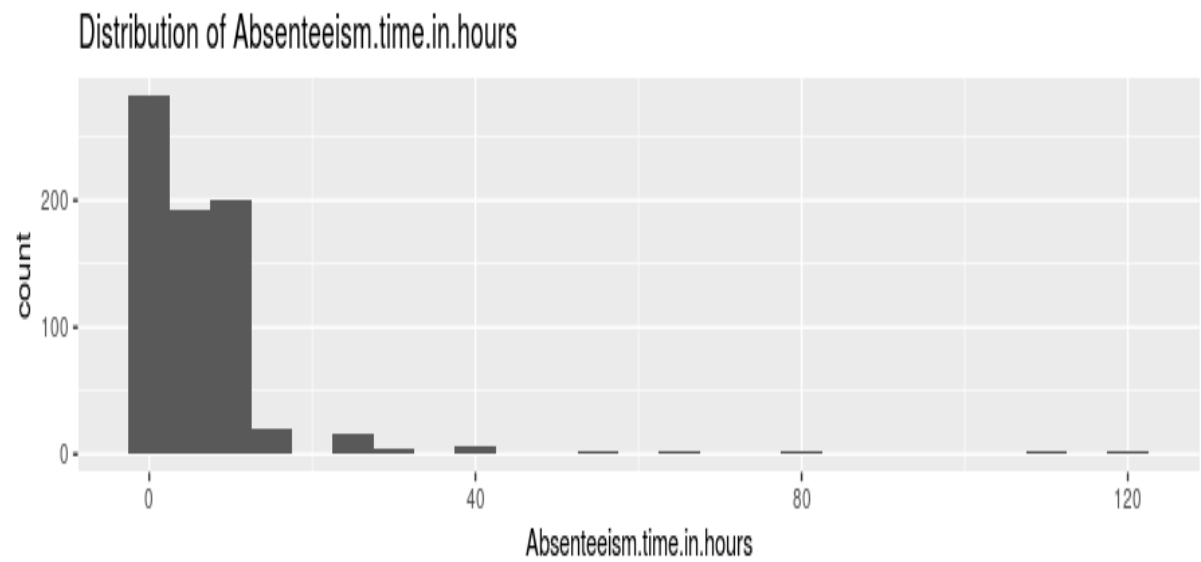
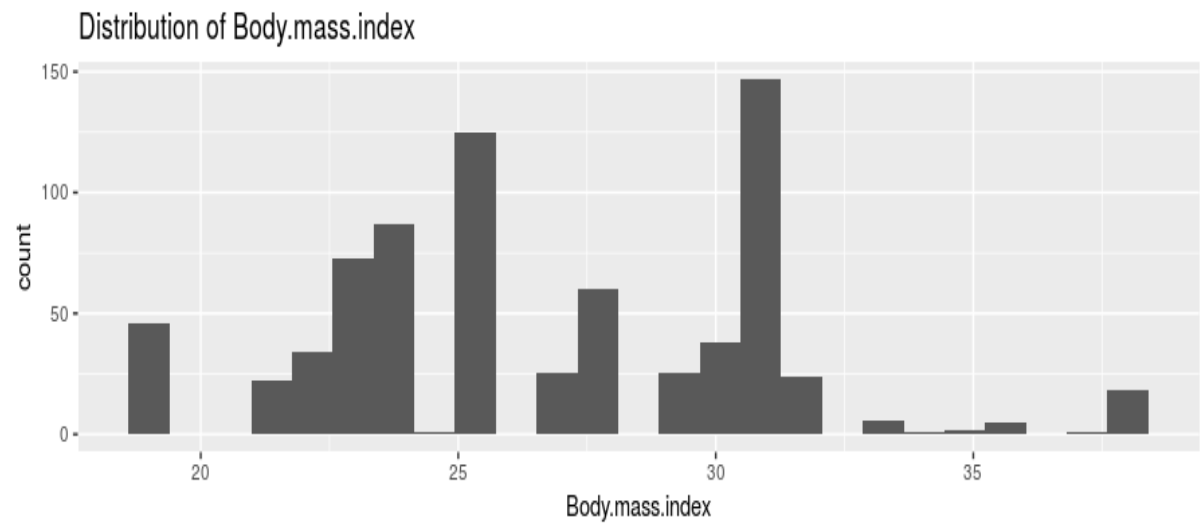






**Fig 5.4 – Distribution of Categorical variables using Bar graph**





**Fig 5.5 – Distribution of Continuous variables using histograms**

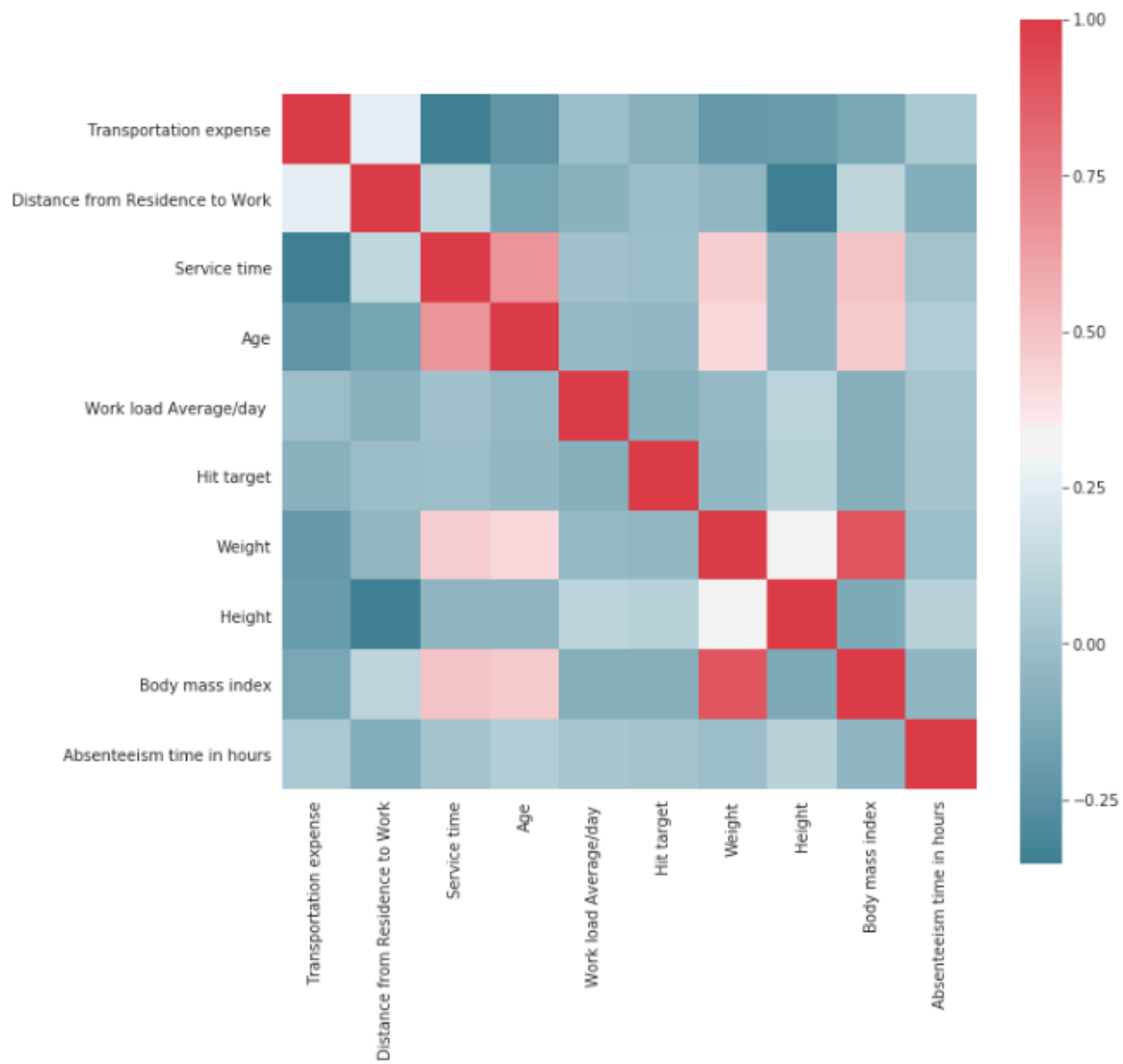


Fig 5.6 – Correlation plot of Continuous variables

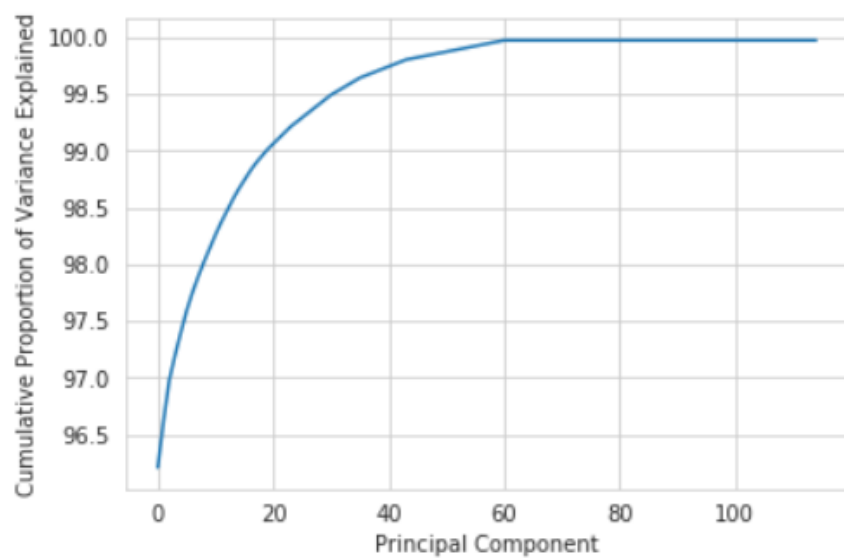


Fig 5.7 – Cumulative Plot of Principal Components

## Chapter 6:

### R code

```
#Cleaning the Environment
rm(list = ls())
#Loading the libraries
library(xlsx)
library(ggplot2)
library(DMwR)
library(gridExtra)
library(MASS)
library(corrgram)
library(rpart)
library(DataCombine)
library(inTrees)
library(caret)
library(gbm)
library(randomForest)
library(mlr)
library(dummies)
library(usdm)
library(rpart.plot)

#libraries to install
#x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "unbalanced", "C50",
"ummies", "e1071", "Information",
#"MASS", "rpart", "gbm", "ROSE", 'sampling', 'DataCombine', 'inTrees')

df = read.xlsx("Absenteeism_at_work_Project.xls", sheetIndex = 1)

# Univariate Analysis and Variable Consolidation
str(df)

df$ID = as.factor(as.character(df$ID))

df$Reason.for.absence[df$Reason.for.absence %in% 0] = 20
df$Reason.for.absence = as.factor(as.character(df$Reason.for.absence))

df$Month.of.absence[df$Month.of.absence %in% 0] = NA
df$Month.of.absence = as.factor(as.character(df$Month.of.absence))

df$Day.of.the.week = as.factor(as.character(df$Day.of.the.week))
df$Seasons = as.factor(as.character(df$Seasons))
df$Disciplinary.failure = as.factor(as.character(df$Disciplinary.failure))
df$Education = as.factor(as.character(df$Education))
df$Social.drinker = as.factor(as.character(df$Social.drinker))
df$Social.smoker = as.factor(as.character(df$Social.smoker))
df$Son = as.factor(as.character(df$Son))
df$Pet = as.factor(as.character(df$Pet))

str(df)

write.csv(df, "Employee.csv", row.names = FALSE)
```

#### ##### Missing Values Analysis #####

```
missing_val = data.frame(apply(df,2,function(x){sum(is.na(x))}))
missing_val$Columns = row.names(missing_val)
names(missing_val)[1] = "Missing_Percentage"
missing_val$Missing_Percentage = (missing_val$Missing_Percentage/nrow(df))*100
missing_val = missing_val[order(-missing_val$Missing_Percentage),]
row.names(missing_val) = NULL
missing_val = missing_val[,c(2,1)]
write.csv(missing_val,"Missing_Percentage.csv",row.names = F)
```

# Missing value percentage representation by bar chart

```
ggplot(data = missing_val[1:3,], aes(x=reorder(Columns, -Missing_Percentage),y =
Missing_Percentage))+
  geom_bar(stat = "identity",fill = "grey")+xlab("Columns")+
  ggtitle("Missing data percentage") + theme_bw()
```

#### ##### Missing value imputation by KNN Method #####

#Actual Value = 29

#mean = 26.68

#median = 25

#knn = 29

#mean method

```
#df$Body.mass.index[is.na(df$Body.mass.index)] = mean(df$Body.mass.index, na.rm = T)
```

# median method

```
#df$Body.mass.index[is.na(df$Body.mass.index)] = median(df$Body.mass.index, na.rm = T)
```

#KNN

```
df = knnImputation(df,5)
```

#Note--- Compare to all the methods, knn method results are more near to the actual values so i decided to go with knn method for further processing--

#Separating continuous and catagorical variables

#for numerical variables

```
numeric_index = sapply(df, is.numeric)
```

```
numeric_data = df[,numeric_index]
```

```
continuous_variables = colnames(numeric_data)
```

```
#continuous_variables = ["Transportation.expense", "Distance.from.Residence.to.Work",
  "Service.time" , "Age" , "Work.load.Average.day." ,
  "Hit.target", "Weight" , "Height", "Body.mass.index",
  "Absenteeism.time.in.hours"
]
```

#for catagorical variables

```
factor_index = sapply(df, is.factor)
```

```
factor_data = df[,factor_index]
```

```

catagorical_variables = colnames(factor_data)

#catagorical_variables = [ "ID", "Reason.for.absence", "Month.of.absence", "Day.of.the.week",
                          "Seasons", "Disciplinary.failure", "Education", "Son",
                          "Social.drinker", "Social.smoker", "Pet"
                          ]

##### Outlier Analysis #####

#Outlier analysis for continuous variables.

for (i in 1:length(continuous_variables)){
  assign(paste0("gn",i), ggplot(aes_string(y = (continuous_variables[i]), x =
"Absenteeism.time.in.hours"), data = subset(df))+
    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=continuous_variables[i],x="Absenteeism.time.in.hours")+
    ggtitle(paste("Box plot of Employee Absenteeism for",continuous_variables[i])))
}

#generating plots

gridExtra::grid.arrange(gn1,gn2,ncol=2)
gridExtra::grid.arrange(gn3,gn4,ncol=2)
gridExtra::grid.arrange(gn5,gn6,ncol=2)
gridExtra::grid.arrange(gn7,gn8,ncol=2)
gridExtra::grid.arrange(gn9,gn10,ncol=2)
#####EDDDDDDDDD#####

#Distribution of factor data using bar plot

bar1 = ggplot(data = df, aes(x = ID)) + geom_bar() +
  ggtitle("Count of ID") + theme_minimal()

bar2 = ggplot(data = df, aes(x = Reason.for.absence)) +
  geom_bar() + ggtitle("Count of Reason for absence") + theme_bw()

bar3 = ggplot(data = df, aes(x = Month.of.absence)) + geom_bar()+
  ggtitle("Count of Month") + theme_bw()

bar4 = ggplot(data = df, aes(x = Disciplinary.failure)) +
  geom_bar() + ggtitle("Count of Disciplinary failure") + theme_bw()

bar5 = ggplot(data = df, aes(x = Education)) + geom_bar()+
  ggtitle("Count of Education")+ theme_bw()

bar6 = ggplot(data = df, aes(x = Son)) + geom_bar()+
  ggtitle("Count of Son") + theme_bw()

bar7 = ggplot(data = df, aes(x = Social.smoker)) + geom_bar() +
  ggtitle("Count of Social smoker") + theme_bw()

```

**#Check the distribution of numerical data using histogram**

```
hist1 = ggplot(data = numeric_data, aes(x =Transportation.expense)) +  
ggtitle("Transportation.expense") + geom_histogram(bins = 25)
```

```
hist2 = ggplot(data = numeric_data, aes(x =Height)) + ggtitle("Distribution of Height") +  
geom_histogram(bins = 25)
```

```
hist3 = ggplot(data = numeric_data, aes(x =Body.mass.index)) + ggtitle("Distribution of  
Body.mass.index") + geom_histogram(bins = 25)
```

```
hist4 = ggplot(data = numeric_data, aes(x =Absenteeism.time.in.hours)) + ggtitle("Distribution of  
Absenteeism.time.in.hours") + geom_histogram(bins = 25)
```

**#Replace all outliers with NA and impute**

```
for(i in continuous_variables){  
  val = df[,i][df[,i] %in% boxplot.stats(df[,i])$out]  
  #print(length(val))  
  df[,i][df[,i] %in% val] = NA  
}
```

**#Imputing missing values with Knn method**

```
df = knnImputation(df,k=5)
```

**##### Feature selection #####**

**#Checking for multicollinearity using VIF**

```
vifcor(numeric_data)
```

**#checking multicollinearity using correlation graph**

```
corrgram(df[,numeric_index], order= F,  
  upper.panel=panel.pie, text.panel = panel.txt, main = "Correlation Plot")
```

**#Dimension Reduction**

```
df_new = subset(df,select = -c(Body.mass.index))
```

```
df_new_cleaned = df_new
```

**##### Feature Scaling #####**

**#Normality Checking**

```
qqnorm(df_new$Absenteeism.time.in.hours)
```

```
hist(df_new$Absenteeism.time.in.hours)
```

**#Updating the continuous variables for normalization**

```
continuous_variables = c('Transportation.expense', 'Distance.from.Residence.to.Work',  
'Service.time',  
  'Age', 'Work.load.Average.day.',  
  'Hit.target', 'Height',  
  'Weight')
```

```
catagorical_variables = c('ID','Reason.for.absence','Disciplinary.failure',
                          'Social.drinker', 'Son', 'Pet', 'Month.of.absence', 'Day.of.the.week', 'Seasons',
                          'Education', 'Social.smoker')
```

```
#Normalization
for(i in continuous_variables)
{
  print(i)
  df_new[,i] = (df_new[,i] - min(df_new[,i]))/(max(df_new[,i])-min(df_new[,i]))
}
```

```
#Creating dummy variables for catagorical varibales to maintain the levels.
```

```
df_new = dummy.data.frame(df_new, catagorical_variables)
```

```
#Cleaning the Environment
```

```
rmExcept(keepers = c("df_new","df","df_new_cleaned"))
```

```
##### Model development #####
```

```
#dividing data into test and train using simple random sampling method (because the target
variable is continuous variable)
```

```
set.seed(1)
train_index = sample(1:nrow(df_new),0.8 * nrow(df_new))
train = df_new[train_index,]
test = df_new[-train_index,]
```

```
#-----Decission tree -----#
```

```
#rpart for regression on training data
```

```
model_DT = rpart(Absenteeism.time.in.hours ~ ., data = train, method="anova")
```

```
#Summary of the model
summary(model_DT)
```

```
#summary in the form of tree
rpart.plot(model_DT)
```

```
#predicting for test data
predictions_DT = predict(model_DT,test[,115])
```

```
#creating separate dataframe for analysing actual and predicted observations.
```

```
df_new_dt_pred = data.frame("actual"=test[,115],"predicted" = predictions_DT)
```

```
#Error Metrics using PostResample method
#Calculates Performance Across Resamples
```

```
print(postResample(pred = predictions_DT, obs = test[,115]))
```



```

#Alternate method for error metrics
#regr.eval(test[,115], predictions_DT, stats = c('mae','rmse','mse'))

#Validation

#RMSE   = 2.276450
#Rsquared = 0.440758
#MAE    = 1.694686


#----- Random Forest -----#

#create model using training data
model_RF = randomForest(Absenteeism.time.in.hours ~., data = train, ntree=500)

#predict for test data
predictions_RF = predict(model_RF,test[, -115])

#creating separate dataframe for actual and predicted data
df_new_rf_pred = data.frame("actual" = test[,115], "predicted" = predictions_RF)

#calculate mae, rmse, and rsquared
print(postResample(pred = predictions_RF, obs = test[,115]))

#Validation

#RMSE   = 2.1941162
#Rsquared = 0.4798489
#MAE    = 1.6107736


#----- Linear Regression -----#

#Train the model with training data
model_LR = lm(Absenteeism.time.in.hours ~ ., data = train)

#summary of the model
summary(model_LR)

#predict for test data
predictions_LR = predict(model_LR,test[, -115])

#creating separate dataframe for actual and predicted data
df_new_lr_pred = data.frame("actual" = test[,115], "predicted" = predictions_LR)

#calculate mae, rmse, and rsquared
print(postResample(pred = predictions_LR, obs = test[,115]))

#Validation

#RMSE   = 2.5593810
#Rsquared = 0.3587518
#MAE    = 1.8604850

```

**#Compared to all the models the performance is not optimizing due to dimension of the data,  
#this can be achived using the principal component analysis**

**##### Dimension Reduction by PCA #####**

**#principal component analysis  
prin\_comp = prcomp(train)**

**#sdev for each of the principal components**

**pr\_stdev = prin\_comp\$sdev**

**#calculate variance  
pr\_var = pr\_stdev^2**

**#Proportion of variance  
pro\_var = pr\_var/sum(pr\_var)**

**#Screen plot  
plot(cumsum(pro\_var), xlab = "Principal Component",  
      ylab = "Cumulative Proportion of Variance Explained",  
      type = "l")**

**#Add a training set with principal of components  
train.data = data.frame(Absenteeism.time.in.hours = train\$Absenteeism.time.in.hours,  
prin\_comp\$x)**

**#From the above plot selecting 45 components since it explains almost 95+ % data variance  
train.data = train.data[,1:45]**

**#Transform test data into PCA  
test.data = predict(prin\_comp, newdata = test)  
test.data = as.data.frame(test.data)**

**#select the first 45 components  
test.data = test.data[,1:45]**

**##### Model development after principal component analysis #####**

**#-----Decission tree -----#**

**#rpart for regression on training data**

**model\_DTP = rpart(Absenteeism.time.in.hours ~ ., data = train.data, method="anova")**

**#Summary of the model  
summary(model\_DTP)**

**#summary in the form of tree  
rpart.plot(model\_DTP)**

**#predicting for test data  
predictions\_DTP = predict(model\_DTP,test.data)**

**#creating separate dataframe for analysing actual and predicted observations.**

```

df_new_dtp_pred = data.frame("actual"=test[,115],"predicted" = predictions_DTP
#Error Metrics using PostResample method
#Calculates Performance Across Resamples

print(postResample(pred = predictions_DTP, obs = test[,115]))

#Alternate method for error metrics
#regr.eval(test[,115], predictions_DTP, stats = c('mae','rmse','mse'))

#Validation

#RMSE   = 0.4427499
#Rsquared = 0.9787753
#MAE    = 0.3012380

#----- Random Forest -----#

#create model using training data
model_RFP = randomForest(Absenteeism.time.in.hours ~., data = train.data, ntree=500)

#predict for test data
predictions_RFP = predict(model_RFP,test.data)

#creating separate dataframe for actual and predicted data
df_new_rfp_pred = data.frame("actual" = test[,115], "predicted" = predictions_RFP)

#calculate mae, rmse, and rsquared
print(postResample(pred = predictions_RFP, obs = test[,115]))

#Validation
#RMSE   = 0.4361794
#Rsquared = 0.9828053
#MAE    = 0.2505895

#----- Linear Regression -----#
#Train the model with training data
model_LRP = lm(Absenteeism.time.in.hours ~ ., data = train.data)

#summary of the model
summary(model_LRP)

#predict for test data
predictions_LRP = predict(model_LRP,test.data)

#creating separate dataframe for actual and predicted data
df_new_lrp_pred = data.frame("actual" = test[,115], "predicted" = predictions_LRP)

#calculate mae, rmse, and rsquared
print(postResample(pred = predictions_LRP, obs = test[,115]))

#Validation
#RMSE   = 0.003016381
#Rsquared = 0.999999023
#MAE    = 0.002247516

```

## Python code

```
#Importing libraries
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from fancyimpute import KNN
from scipy.stats import chi2_contingency
from random import randrange, uniform
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
import seaborn as sns
% matplotlib inline

#Reading File
df = pd.read_excel("../input/Absenteeism_at_work_Project.xls")

#checking file
df.head()

##### Exploratory Data Analysis #####

#Rearranging datatypes of the variables

df['ID'] = df['ID'].astype('category')

df['Reason for absence'] = df['Reason for absence'].replace(0,20)
df['Reason for absence'] = df['Reason for absence'].astype('category')

df['Month of absence'] = df['Month of absence'].replace(0,np.nan)
df['Month of absence'] = df['Month of absence'].astype('category')

df['Day of the week'] = df['Day of the week'].astype('category')
df['Seasons'] = df['Seasons'].astype('category')
df['Disciplinary failure'] = df['Disciplinary failure'].astype('category')
df['Education'] = df['Education'].astype('category')
df['Son'] = df['Son'].astype('category')
df['Social drinker'] = df['Social drinker'].astype('category')
df['Social smoker'] = df['Social smoker'].astype('category')
df['Pet'] = df['Pet'].astype('category')

#separating continous and categrocal variables
continuous_variables = ["Transportation expense", "Distance from Residence to Work",
                        "Service time", "Age", "Work load Average/day",
                        "Hit target", "Weight", "Height", "Body mass index",
                        "Absenteeism time in hours"
                        ]

categorical_variables = [ "ID", "Reason for absence", "Month of absence", "Day of the week",
                        "Seasons", "Disciplinary failure", "Education", "Son",
                        "Social drinker", "Social smoker", "Pet"
                        ]
```

#### ##### Missing Value Analysis #####

#creating separate dataframe with missing values

```
missing_val = pd.DataFrame(df.isnull().sum())
missing_val = missing_val.reset_index()
missing_val = missing_val.rename(columns = {'index' : 'Variables', 0: 'missing_perc'})
missing_val
missing_val['missing_perc'] = (missing_val['missing_perc']/len(df))*100
missing_val = missing_val.sort_values('missing_perc', ascending=False).reset_index(drop = True)
missing_val.to_csv("missing_val.csv", index=False)
```

#Actual Value = 29

#Mean = 26

#Median = 25

#KNN = 27

```
#print(df['Body mass index'].iloc[9])
```

```
#df['Body mass index'].iloc[9] = np.nan
```

#Mean

```
#df['Body mass index'] = df['Body mass index'].fillna(df['Body mass index'].median())
```

#Median

```
#df['Body mass index'] = df['Body mass index'].fillna(df['Body mass index'].median())
```

#KNN

```
df = pd.DataFrame(KNN(k = 5).fit_transform(df), columns = df.columns)
```

#Checking null value count

```
df.isnull().sum()
```

#Rounding the values of categorical variables

```
for i in categorical_variables:
```

```
    df.loc[:,i] = df.loc[:,i].round()
```

```
    df.loc[:,i] = df.loc[:,i].astype('category')
```

#### ##### Visualization of Distributed data by graphs #####

```
sns.set_style("whitegrid")
```

```
sns.factorplot(data=df, x='Reason for absence', kind= 'count',size=3,aspect=2)
```

```
sns.factorplot(data=df, x='Seasons', kind= 'count',size=3,aspect=2)
```

```
sns.factorplot(data=df, x='Education', kind= 'count',size=3,aspect=2)
```

```
sns.factorplot(data=df, x='Disciplinary failure', kind= 'count',size=3,aspect=2)
```

#### ##### Outlier Analysis #####

#Checking Outliers in data using boxplot

```
sns.boxplot(data=df[['Hit target','Age','Service time','Transportation expense',]])
```

```
fig=plt.gcf()
```

```
fig.set_size_inches(8,8)
```

```

#Checking Outliers in data using boxplot
sns.boxplot(data=df[['Absenteeism time in hours','Body mass index','Height','Weight',]])
fig=plt.gcf()
fig.set_size_inches(8,8)

sns.boxplot(data=df[['Work load Average/day ','Distance from Residence to Work',]])
fig=plt.gcf()
fig.set_size_inches(8,8)

#Detecting outliers using boxplot and replacing with NA
for i in continuous_variables:
    q75, q25 = np.percentile(df[i],[75,25])

    # Calculating Interquartile range
    iqr = q75 - q25

    #calculating upper and lower fences
    minimum = q25 - (iqr*1.5)
    maximum = q75 + (iqr*1.5)

    #Replace all the outliers with NA
    df.loc[df[i]<minimum,i] = np.nan
    df.loc[df[i]>maximum,i] = np.nan

#Check for missing values
df.isnull().sum()

#Impute missing values with knn
df = pd.DataFrame(KNN(k=3).fit_transform(df), columns = df.columns)

#Check for missing values after applying KNN
df.isnull().sum()

#checking once again for outliers in the data after applying KNN
sns.boxplot(data=df[['Absenteeism time in hours','Body mass index','Height','Weight',]])
fig=plt.gcf()
fig.set_size_inches(8,8)

#checking once again for outliers in the data after applying KNN
sns.boxplot(data=df[['Hit target','Age','Service time','Transportation expense',]])
fig=plt.gcf()
fig.set_size_inches(8,8)

#checking once again for outliers in the data after applying KNN
sns.boxplot(data=df[['Work load Average/day ','Distance from Residence to Work',]])
fig=plt.gcf()
fig.set_size_inches(8,8)

```

**##### Future Selection #####**

**##Correlation analysis**

**#Correlation plot**

**df\_cor = df.loc[:,continuous\_variables]**

**#Check for multicollinearity using corelation graph**

**#Set the width and hieght of the plot**

**f, ax = plt.subplots(figsize=(10,10))**

**#Generate correlation matrix**

**cor\_mat = df\_cor.corr()**

**#Plot using seaborn library**

**sns.heatmap(cor\_mat, mask=np.zeros\_like(cor\_mat, dtype=np.bool),**

**cmap=sns.diverging\_palette(220, 10, as\_cmap=True),**

**square=True, ax=ax)**

**plt.plot()**

**#Getting copy of the data**

**df\_old = df.copy()**

**#Variable reduction**

**df\_new = df.drop(['Body mass index'], axis = 1)**

**#Updating columns in Continous\_variable**

**continuous\_variables.remove('Body mass index')**

**continuous\_variables.remove('Absenteeism time in hours')**

**##### Future Scaling #####**

**#Make a copy of cleaned data**

**#df\_cleaned\_data = df\_new.copy()**

**df\_new = df\_cleaned\_data.copy()**

**df\_cleaned\_data.shape**

**#Normality Check**

**for i in continuous\_variables:**

**plt.hist(df\_new[i],bins='auto')**

**plt.title("Checking Distribution for Variable "+str(i))**

**plt.ylabel("Density")**

**plt.xlabel(i)**

**plt.show()**

**#Normalization of continous variables**

**for i in continuous\_variables:**

**df\_new[i] = (df\_new[i] - min(df\_new[i]))/(max(df\_new[i]) - min(df\_new[i]))**

**#Dummy Variable creation for categorical variables**

**df\_new = pd.get\_dummies(data = df\_new,columns=categorical\_variables)**

# ##### Machine Learning Models #####

**#Splitting data into train and test data**

**from sklearn.model\_selection import train\_test\_split**

**X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(df\_new.iloc[:,df\_new.columns != 'Absenteeism  
time in hours' ],df\_new.iloc[:, 8],test\_size = 0.20, random\_state = 1)**

**#-----Decision Tree-----#**

**#Importing libraries for Decision tree regressor**

**from sklearn.tree import DecisionTreeRegressor**

**#create a model decision tree using DecisionTreeRegressor**

**model\_DT = DecisionTreeRegressor(random\_state = 1).fit(X\_train,Y\_train)**

**#Predict for the test data**

**predictions\_DT = model\_DT.predict(X\_test)**

**#Create separate dataframe for actual and predicted data**

**df\_new\_dt\_pred = pd.DataFrame({'actual':Y\_test,'predicted':predictions\_DT})**

**print(df\_new\_dt\_pred.head())**

**#Function to create to RMSE**

**def RMSE(y\_actual,y\_predicted):**

**rmse = np.sqrt(mean\_squared\_error(y\_actual,y\_predicted))**

**return rmse**

**#Calculate RMSE and R-Squared Value**

**print("RMSE: "+str(RMSE(Y\_test, predictions\_DT)))**

**print("R-Squared Value: "+str(r2\_score(Y\_test, predictions\_DT)))**

**\*\*Decision Tree**

**RMSE: 3.7141966443496677**

**R-Squared Value: -0.13882343999361768\*\***

**#-----Random Forest-----#**

**#Importing libraries for Random Forest**

**from sklearn.ensemble import RandomForestRegressor**

**#create a model Random forest using RandomForestRegressor**

**model\_RF = RandomForestRegressor(n\_estimators = 500, random\_state = 1).fit(X\_train,Y\_train)**

**#predict for test data**

**predictions\_RF = model\_RF.predict(X\_test)**

**#create a dataframe for actual and predicted data**

**df\_new\_rf\_pred = pd.DataFrame({'actual':Y\_test,'predicted':predictions\_RF})**

**print(df\_new\_rf\_pred.head())**

**#calculate RMSE and RSquared values**

**print("RMSE: "+str(RMSE(Y\_test, predictions\_RF)))**

**print("R-Squared Value: "+str(r2\_score(Y\_test, predictions\_RF)))**

**\*\*Random Forest**

**RMSE: 2.725268748784219**

**R-Squared Value: 0.386880282274243\*\***



```

#-----Linear Regression-----#
#Import libraries for Linear Regression
from sklearn.linear_model import LinearRegression

#Create model Linear Regression using LinearRegression
model_LR = LinearRegression().fit(X_train,Y_train)

#Predict for the test cases
predictions_LR = model_LR.predict(X_test)

#Create a separate dataframe for the actual and predicted data
df_new_lr_pred = pd.DataFrame({'actual':Y_test,'predicted':predictions_LR})

print(df_new_lr_pred.head())

#Calculate RMSE and RSquared values
print("RMSE: "+str(RMSE(Y_test, predictions_LR)))
print("R-Squared Value: "+str(r2_score(Y_test, predictions_LR)))

**Linear Regression
RMSE: 16390064550.910776
R-Squared Value: -2.2176241320666194e+19**

##### Dimension Reduction using PCA #####
#Get a target variable
target_variable = df_new['Absenteeism time in hours']
df_new.shape

#import library for PCA
from sklearn.decomposition import PCA

#Converting data into numpy array
X = df_new.values

pca = PCA(n_components = 115)
pca.fit(X)

#Proportion of variance
var = pca.explained_variance_ratio_

#Calculate Screen plot
var1 = np.cumsum(np.round(pca.explained_variance_ratio_,decimals=4)*100)

#Draw the plot
plt.plot(var1)
plt.xlabel("Principal Component")
plt.ylabel("Cumulative Proportion of Variance Explained")
plt.show()

```

```

#Selecting 45 Components since it explains almost 95+ % data variance
pca = PCA(n_components=45)

#Fitting the selected components to the data
pca.fit(X)

#Splitting data into train and test
X_train, X_test, Y_train, Y_test = train_test_split(X, target_variable, test_size=0.2, random_state =
1)

##### Model Development After PCA #####

#-----Decision Tree-----#
#Importing libraries for Decision tree regressor
from sklearn.tree import DecisionTreeRegressor

#create a model decision tree using DecisionTreeRegressor
model_DTP = DecisionTreeRegressor(random_state = 1).fit(X_train, Y_train)

#Predict for the test data
predictions_DTP = model_DTP.predict(X_test)

#Create separate dataframe for actual and predicted data
df_new_dtp_pred = pd.DataFrame({'actual':Y_test, 'predicted':predictions_DTP})

print(df_new_dtp_pred.head())

#Function to create to RMSE
def RMSE(y_actual, y_predicted):
    rmse = np.sqrt(mean_squared_error(y_actual, y_predicted))
    return rmse

#Calculate RMSE and R-Squared Value
print("RMSE: "+str(RMSE(Y_test, predictions_DTP)))
print("R-Squared Value: "+str(r2_score(Y_test, predictions_DTP)))

**Decision Tree
RMSE: 0.07939996345382828
R-Squared Value: 0.9994795641369799**

#-----Random Forest-----#
#Importing libraries for Random Forest
from sklearn.ensemble import RandomForestRegressor

#create a model Random forest using RandomForestRegressor
model_RFP = RandomForestRegressor(n_estimators = 500, random_state = 1).fit(X_train, Y_train)

#predict for test data
predictions_RFP = model_RFP.predict(X_test)

#create a dataframe for actual and predicted data
df_new_rfp_pred = pd.DataFrame({'actual':Y_test, 'predicted':predictions_RFP})
print(df_new_rfp_pred.head())

#calculate RMSE and RSquared values

```

```
print("RMSE: "+str(RMSE(Y_test, predictions_RFP)))
print("R-Squared Value: "+str(r2_score(Y_test, predictions_RFP)))
```

**\*\*Random Forest**

**RMSE: 0.05554332987415368**

**R-Squared Value: 0.99974532258328\*\***

**#-----Linear Regression-----#**

**#Import libraries for Linear Regression**

**from sklearn.linear\_model import LinearRegression**

**#Create model Linear Regression using LinearRegression**

**model\_LRP = LinearRegression().fit(X\_train,Y\_train)**

**#Predict for the test cases**

**predictions\_LRP = model\_LRP.predict(X\_test)**

**#Create a separate dataframe for the actual and predicted data**

**df\_new\_lrp\_pred = pd.DataFrame({'actual':Y\_test,'predicted':predictions\_LRP})**

**print(df\_new\_lrp\_pred.head())**

**#Calculate RMSE and RSquared values**

**print("RMSE: "+str(RMSE(Y\_test, predictions\_LRP)))**

**print("R-Squared Value: "+str(r2\_score(Y\_test, predictions\_LRP)))**

**\*\*Linear Regression**

**RMSE: 0.0004365935184874104**

**R-Squared Value: 0.9999999842644771\*\***