Employee Absenteeism

Author-

Pooja Yadwani

Contents

Chapter 1 3

Introduction 3

1.1 Problem Statement 3

1.2 Data 3

Chapter 2 5

Methodology 5

2.1 Exploratory Data Analysis (EDA) 5

2.1.1 Target Variable - Absenteeism time in hours 6

2.1.2 Uniqueness in Variable 7

2.1.3 Missing Value Analysis 8

2.1.4 Outlier Analysis 9

2.2 Feature Selection 11

2.3 Feature Scaling 12

Chapter 3 13

Modeling 13

3.1 Decision Tree 13

3.2 Random Forest 13

3.3 Liner Regression 13

3.4 Gradient boosting 13

Chapter 4 15

Conclusion 15

4.1 Model Evaluation 15

4.2 Model Selection 15

Appendix A: Extra Figures 16

Appendix B: R Code 18

References 31

Chapter 1

Introduction

Absenteeism is the habitual non-presence of an employee at his or her job. Habitual non-presence extends beyond what is expected as a normal amount of time away for reasons such as scheduled vacation or occasional illness. Possible causes of absenteeism include job dissatisfaction, ongoing personal issues and chronic medical problems. Regardless of the cause, a worker with a pattern of being absent may put his reputation and his employed status at risk. However, some forms of absence from work are legally protected and cannot be grounds for termination.

For example, Angela is dissatisfied with her working environment and job responsibilities. She regularly calls in sick to work for days at a time, often missing five days each month, even though she does not have any actual chronic health problems. This is an example of absenteeism.

BREAKING DOWN Absenteeism

Absenteeism refers to absence from work that extends beyond what would be considered reasonable and normal due to vacation, personal time or occasional illness. Companies expect their employees to miss some work each year due to vacation, illness and personal issues and responsibilities, but missing work becomes a problem for the company when the employee is absent repeatedly and/or unexpectedly, especially if that employee must be paid while absent. While disability leave, performance of jury duty and the observance of religious holidays are all legally protected reasons for an employee to miss work, some employees abuse these laws to take time off that they shouldn't, which incurs unfair costs to the employer.

1.1 Problem Statement

The shortage of company man-power has in recent years accentuated the problem of absence in industry and revived both managerial and scientific interest in the subject. To management these occurrences are a drain to productivity of the concern which can no longer be dealt with by methods which, presumably, were effective when the supply of labor was abundant.

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 it 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?

We aim at building a suitable model in both R and Python to answer the above two questions with a proper report. Let us begin with our analysis.

1.2 Data

Absenteeism_at_work_Project dataset was provided for analysis. Data contains 20 predictor variables and 1 target variable.

Variables	Description
Individual identification (ID)	ID of each employee
Reason for absence (ICD).	Absences attested by the International Code of Diseases (ICD) stratified into 21
Month of absence	-

Day of the week	Monday (2), Tuesday (3), Wednesday (4), Thursday (5), Friday (6)
Seasons	summer (1), autumn (2), winter (3), spring (4)
Transportation expense	-
Distance from Residence to Work	-
Service time	-
Age	-
Work load Average/day	-
Hit target	-
Disciplinary failure	yes=1; no=0
Education	-
Son	number of children
Social drinker	yes=1; no=0
Social smoker	yes=1; no=0
Pet	number of pets
Weight	-
Height	-
Body mass index	-
Absenteeism time in hours	**Target Variable**

Size of Dataset Provided: - 740 rows, 21 Columns

ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target
11	26	7	3	1	289	36	13	33	239554	97
36	0	7	3	1	118	13	18	50	239554	97
3	23	7	4	1	179	51	18	38	239554	97
7	7	7	5	1	279	5	14	39	239554	97
11	23	7	5	1	289	36	13	33	239554	97

Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Weight	Height	Body mass index	Absenteeism time in hours
0	1	2	1	0	1	90	172	30	4
1	1	1	1	0	0	98	178	31	0
0	1	0	1	0	0	89	170	31	2
0	1	2	1	1	0	68	168	24	4
0	1	2	1	0	1	90	172	30	2

Chapter 2

Methodology

Employee Absenteeism is a business scenario in which a company is trying to analyze reasons for employees talking frequent day off from work. For reducing absenteeism rate, we need to identify the issues which employees face or reasons they present to the firm while talking day off. Also, we have some data to train our model which makes our problem as Supervised Classification problem.

- **Exploratory Data Analysis(EDA)-** It includes following steps
  - Looking into the data and analyzing various variables,
  - Visualization,
  - Missing value analysis,
  - Outlier Analysis
  - Correlation analysis,
  - ANOVA
  - Feature Scaling
  - Feature Sampling.

- **Basic Modeling-** Trying different models over preprocessed data
  - Decision Tree
  - Random forest,
  - Linear regression,
  - Gradient Boosting

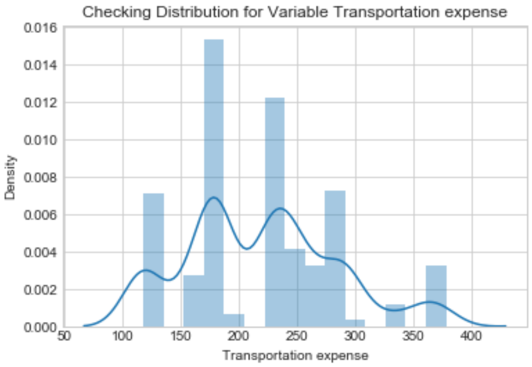
- **Model Evaluation & Optimization-** Evaluating model performances and then selecting the best model fit for our data, optimizing hyper parameters tuning and cost effectiveness of model. This step is optional. We may or may not involve it. It is basically done to avoid a scenario where the selected approach works very well with training data but fails to support out test data in similar way.

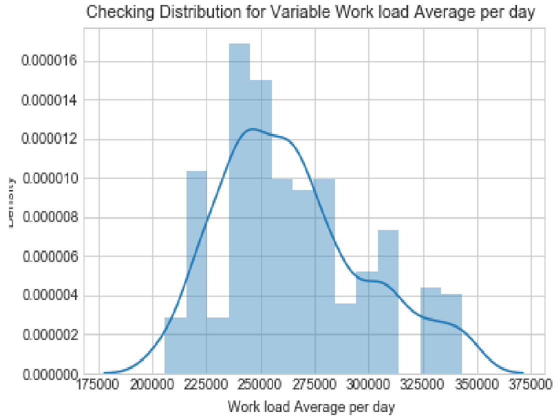
- **Implementation model on Final test data and saving the results**

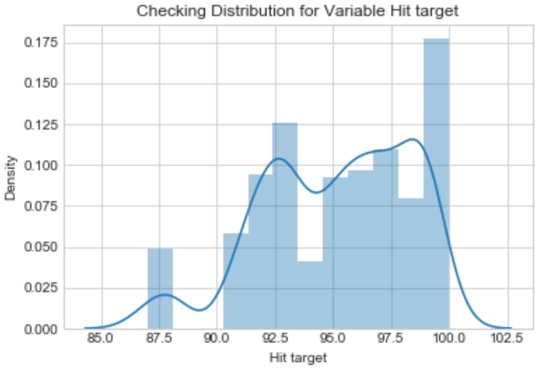
2.1 Exploratory Data Analysis (EDA)

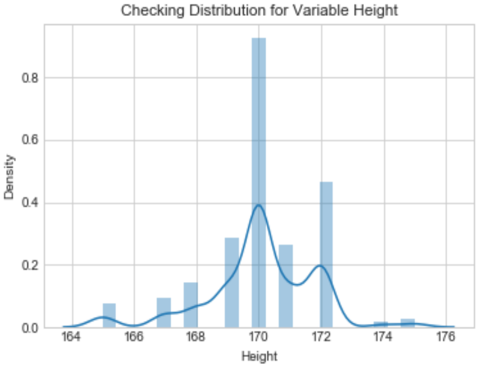
Exploratory Data Analysis refers to the critical process of performing initial investigations on data to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations. It is a good practice to understand the data first and try to gather as many insights from it. EDA is all about making sense of data in hand, before getting them dirty with it.

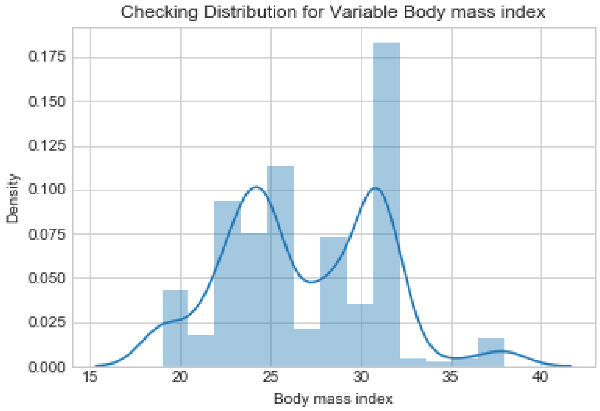
To start this process, we will first try and look at all the probability distributions of the variables. Most analysis like regression, require the data to be normally distributed. We can visualize that in a glance by looking at the probability distributions of the variable.

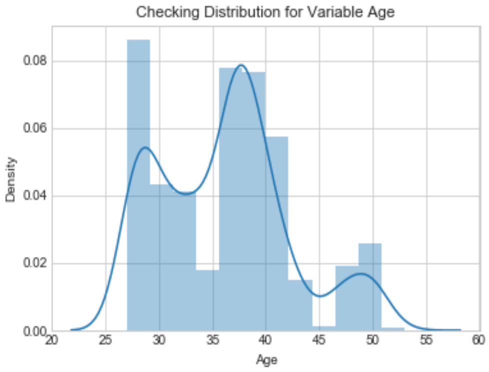






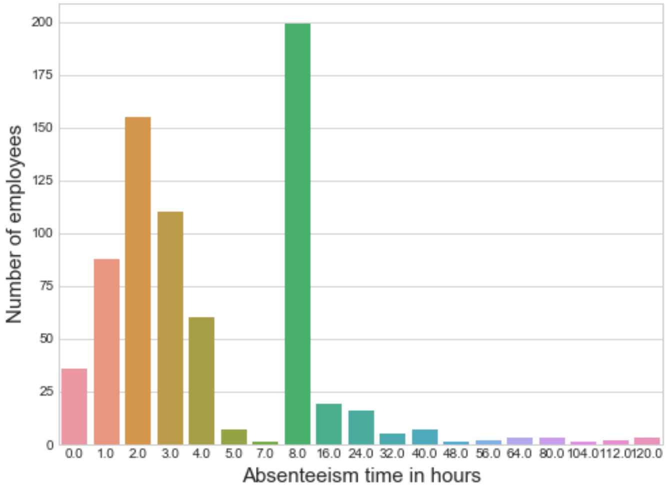






2.1.1 Target Variable - Absenteeism time in hours

Our target variable is continuous which includes 19 unique values that employees have availed during their absence.



2.1.2 Uniqueness in Variable

In the given data set there are 21 variables and data types of all variables are either float64 or int64. There are 740 observations and 21 columns in our data set. Missing value is also present in our data. We have concluded that there are 10 continuous variables and 11 categorical variables in nature.

Variable	Unique Counts
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**

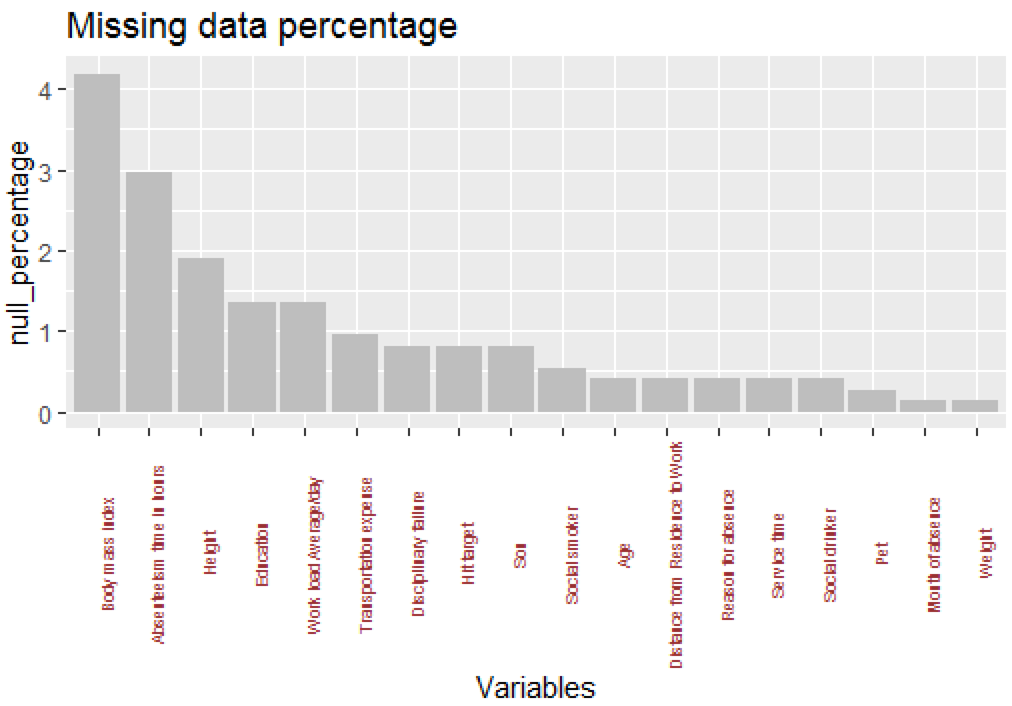
2.1.3 Missing Value Analysis

In statistics, *missing data*, or *missing values*, occur when no *data value* is stored for the variable in an observation. *Missing data* are a common occurrence and can have a significant effect on the conclusions that can be drawn from the *data*. If a column has more than 30% of data as missing value either we ignore the entire column, or we ignore those observations. In the given data the maximum percentage of missing value is 4.189% for body mass index column. So, we will compute missing value for all the columns.

Variable	Null Values
ID	0
Reason for absence	3
Month of absence	1
Day of the week	0
Seasons	0
Transportation expense	7
Distance from Residence to Work	3
Service time	3
Age	3
Work load Average/day	10
Hit target	6
Disciplinary failure	6
Education	10
Son	6
Social drinker	3
Social smoker	4
Pet	2
Weight	1
Height	14
Body mass index	31
Absenteeism time in hours	**22**

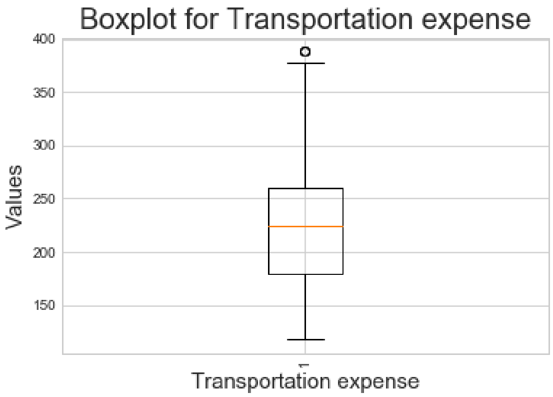
In this project we have used mode imputation to impute missing value.
Technically KNN Imputation is the best method to deal with missing values. However, we need "fancyimpute" library to use it. Since I am having a 32-bit OS I could not install tensorflow (prerequisite for fanyimpute). Hence, I have used "mode" for imputation.

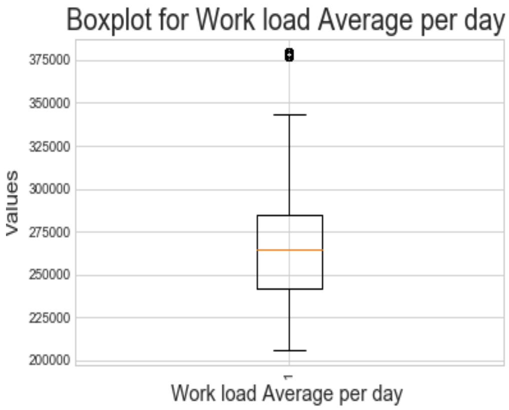
Below is a plot showing missing values for all the variables in dataset:

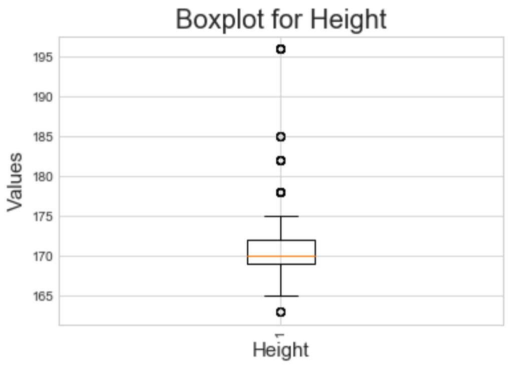


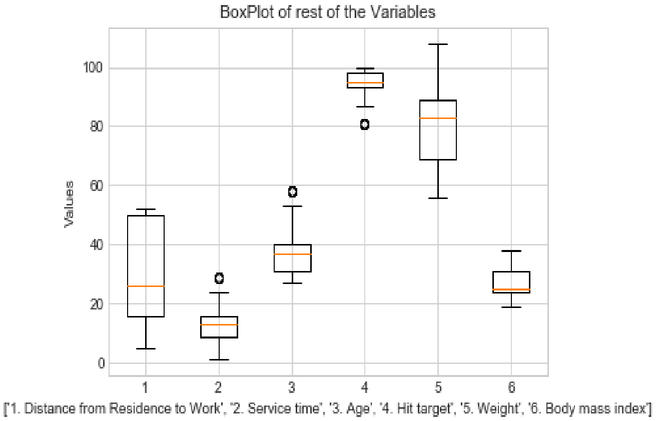
2.1.4 Outlier Analysis

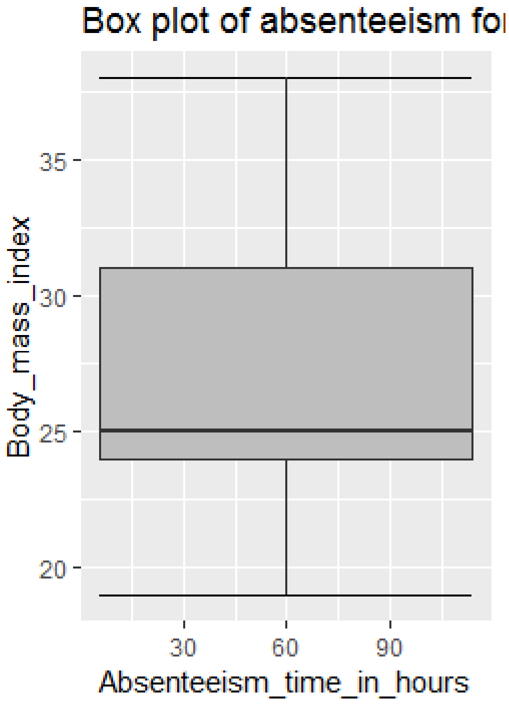
We can clearly observe from these probability distributions that most of the variables are skewed. The skew in these distributions can be most likely explained by the presence of outliers and extreme values in the data. One of the other steps of pre-processing apart from checking for normality is the presence of outliers. In this case we use a classic approach of removing outliers. We visualize the outliers using boxplots.

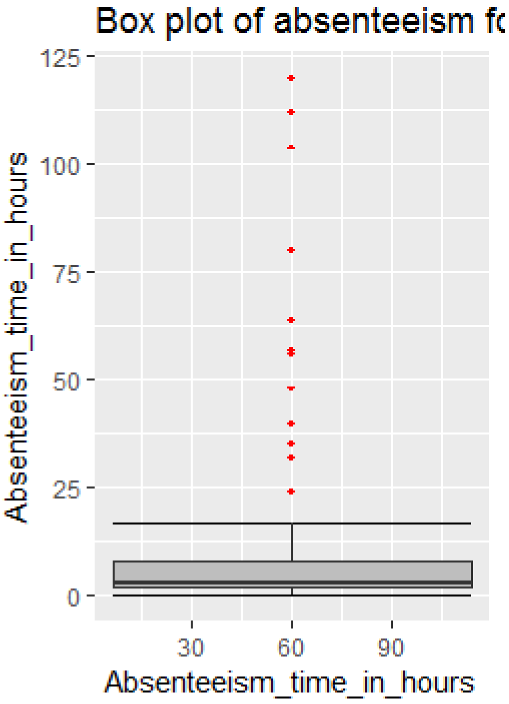








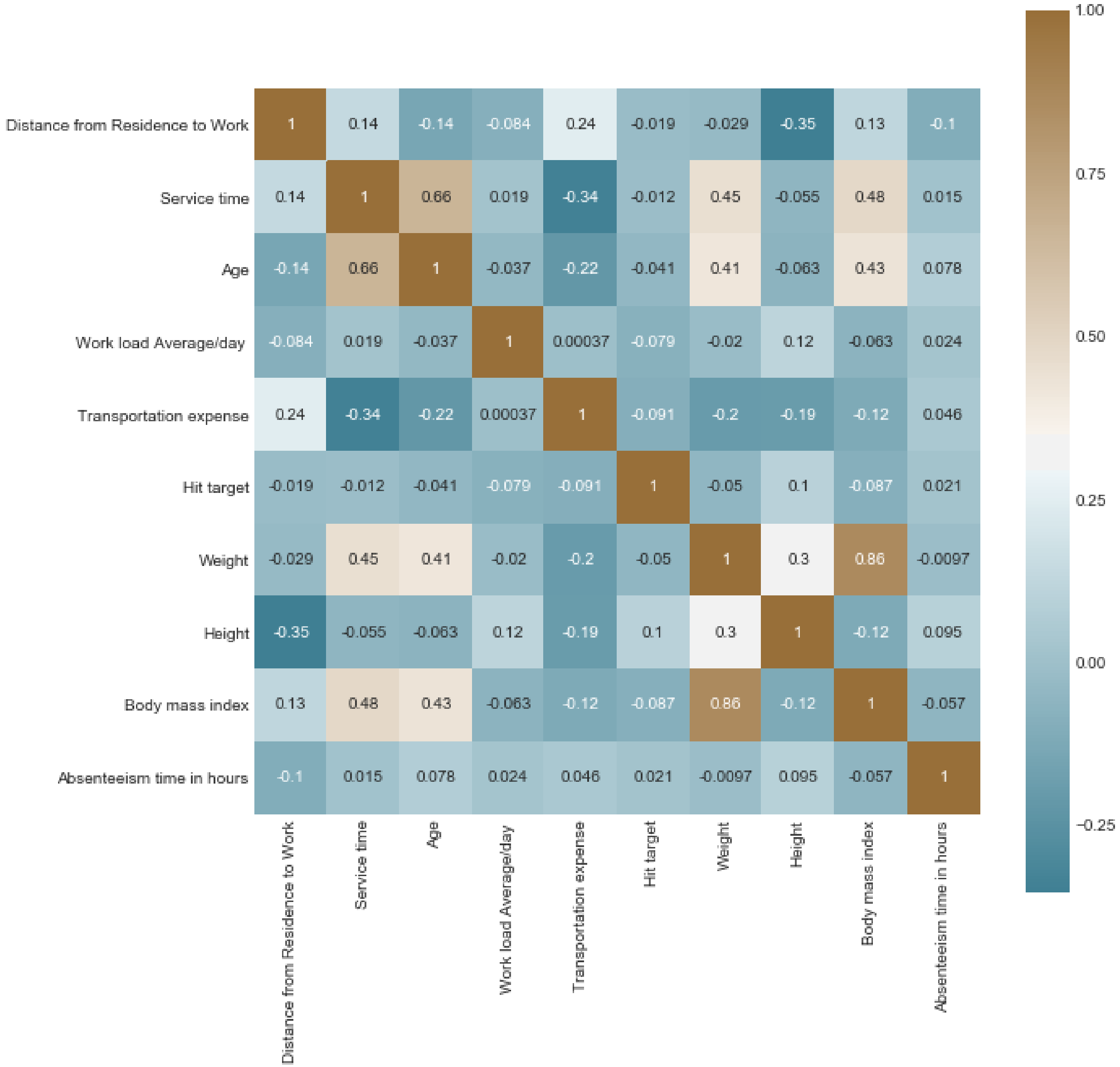




From the boxplot almost all the variables **except "Distance from residence to work", "Weight" and "Body mass index"** consists of outliers. We have converted the outliers (data beyond minimum and maximum values) as NA i.e. missing values and fill them by **mode** imputation.

2.2 Feature Selection

Before performing any type of modeling, we need to assess the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of class prediction. Selecting subset of relevant columns for the model construction is known as **Feature Selection**. We cannot use all the features because some features may be carrying the same information or irrelevant information which can increase overhead. To reduce overhead, we adopt feature selection technique to extract meaningful features out of data. This in turn helps us to avoid the problem of multi collinearity. In this project we have selected **Correlation Analysis** for numerical variable and **ANOVA** (Analysis of variance) for categorical variable.



	Distance from Residence to Work	Service time	Age	Work load Average/day	Transportation expense	Hit target	Weight	Height	Body mass index	Absenteeism time in hours
Distance from Residence to Work	1	0.14	-0.14	-0.084	0.24	-0.019	-0.029	-0.35	0.13	-0.1
Service time	0.14	1	0.66	0.019	-0.34	-0.012	0.45	-0.055	0.48	0.015
Age	-0.14	0.66	1	-0.037	-0.22	-0.041	0.41	-0.063	0.43	0.078
Work load Average/day	-0.084	0.019	-0.037	1	0.00037	-0.079	-0.02	0.12	-0.063	0.024
Transportation expense	0.24	-0.34	-0.22	0.00037	1	-0.091	-0.2	-0.19	-0.12	0.046
Hit target	-0.019	-0.012	-0.041	-0.079	-0.091	1	-0.05	0.1	-0.087	0.021
Weight	-0.029	0.45	0.41	-0.02	-0.2	-0.05	1	0.3	0.86	-0.0097
Height	-0.35	-0.055	-0.063	0.12	-0.19	0.1	0.3	1	-0.12	0.095
Body mass index	0.13	0.48	0.43	-0.063	-0.12	-0.087	0.86	-0.12	1	-0.057
Absenteeism time in hours	-0.1	0.015	0.078	0.024	0.046	0.021	-0.0097	0.095	-0.057	1

From correlation analysis we have found that **Weight** and **Body mass index** has high correlation (>0.7), so we have excluded the **Weight** column.

2.3 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 preprocessing step. Since the range of values of raw data varies widely, in some machine learning algorithms, objective functions will not work properly without normalization. For example, 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 approximately proportionately to the final distance. Since our data is not uniformly distributed we will use **Normalization** as Feature Scaling Method.

Chapter 3

Modeling

After a thorough preprocessing we will use some regression models on our processed data to predict the target variable. Following are the models which we have built –

3.1 Decision Tree

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. Each branch connects nodes with "and" and multiple branches are connected by "or". It can be used for classification and regression. It is a supervised machine learning algorithm. Accept continuous and categorical variables as independent variables. Extremely easy to understand by the business users. Split of decision tree is seen in the below tree. The RMSE value and R^2 value for our project in R and Python are –

Decision Tree	R	PYTHON
RMSE Train	4.44	0.058
RMSE Test	2.37	0.061
R^2 Test	0.48	0.922

3.2 Random Forest

Random Forest is an ensemble technique that consists of many decision trees. The idea behind Random Forest is to build n number of trees to have more accuracy in dataset. It is called random forest as we are building n no. of trees randomly. In other words, to build the decision trees it selects randomly n no of variables and n no of observations to build each decision tree. It means to build each decision tree on random forest we are not going to use the same data. The RMSE value and R^2 value for our project in R and Python are –

Random Forest	R	PYTHON
RMSE Train	4.52	0.0027
RMSE Test	1.43	0.0018
R^2 Test	0.85	0.99

3.3 Liner Regression

Linear Regression is one of the statistical methods of prediction. It is applicable only on continuous data. To build any model we have some assumptions to put on data and model. Here are the assumptions to the linear regression model.

Linear Regression	R	PYTHON
RMSE Train	4.17	6.89E-16
RMSE Test	2.81	6.47E-16
R^2 Test	0.26	1

3.4 Gradient boosting

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. It builds the model in a stage-wise fashion like other boosting methods do, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

Gradient boosting	R	PYTHON
RMSE Train	4.53	0.0003
RMSE Test	1.8	0.001
R^2 Test	0.71	0.99

Chapter 4

Conclusion

In this chapter we are going to evaluate our models, select the best model for our dataset and try to get answers of the asked questions.

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.

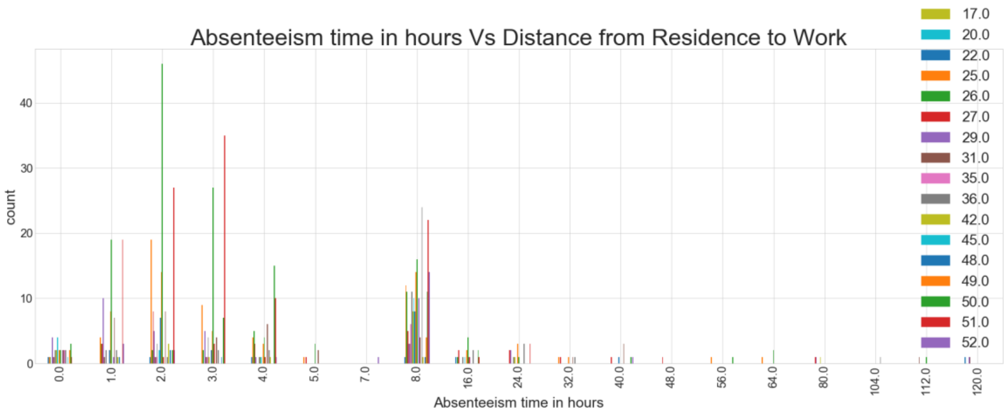
4.2 Model Selection

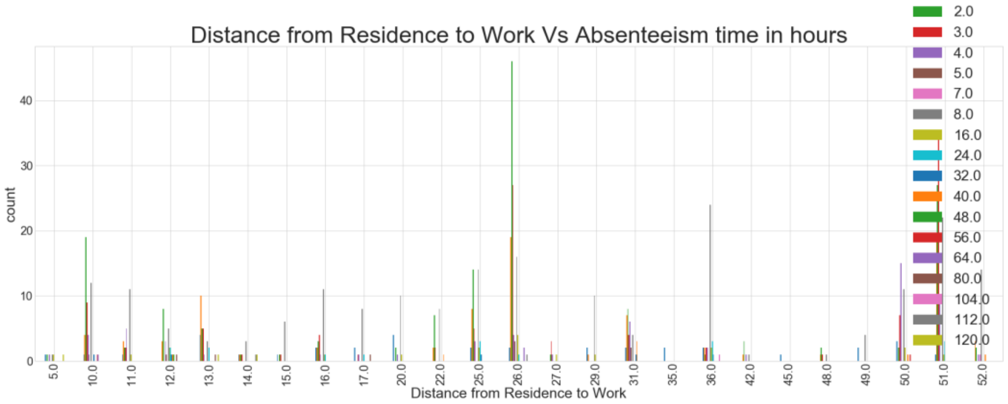
From the observation of all **RMSE Value** and **R-Squared** Value we have concluded that **Linear Regression Model** has minimum value of RMSE and its **R-Squared** Value is also maximum (i.e. 1). The RMSE value of Testing data and Training does not differs a lot this implies that it is not the case of overfitting.

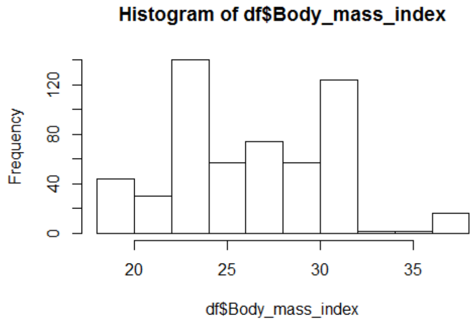
Below is the final matrix showing final results for all the models used for this analysis-

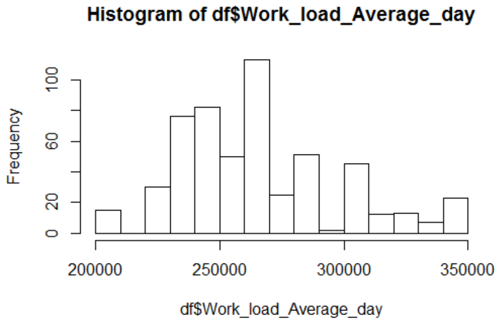
Model	Parameter	R	PYTHON
Decision Tree	**RMSE Train**	4.44	0.058
	RMSE Test	2.37	0.061
	R^2 Test	0.48	0.922
Random Forest	**RMSE Train**	4.52	0.0027
	RMSE Test	1.43	0.0018
	R^2 Test	0.85	0.99
Linear Regression	**RMSE Train**	4.17	6.89E-16
	RMSE Test	2.81	6.47E-16
	R^2 Test	0.26	1
Gradient boosting	**RMSE Train**	4.53	0.0003
	RMSE Test	1.8	0.001
	R^2 Test	0.71	0.99

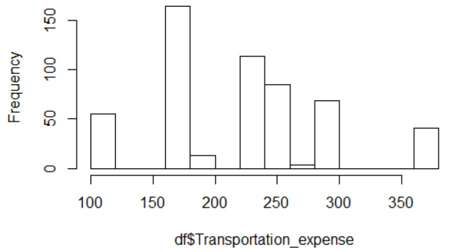
Appendix A: Extra Figures



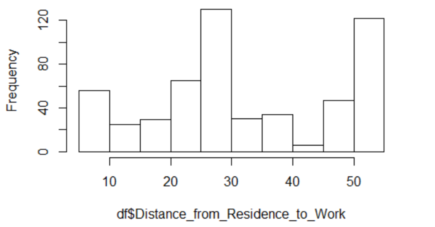




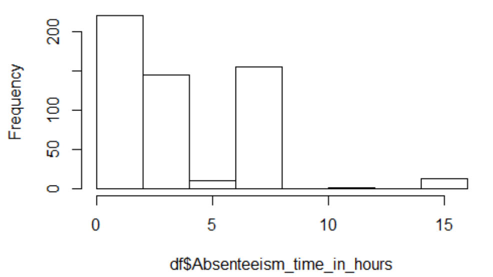




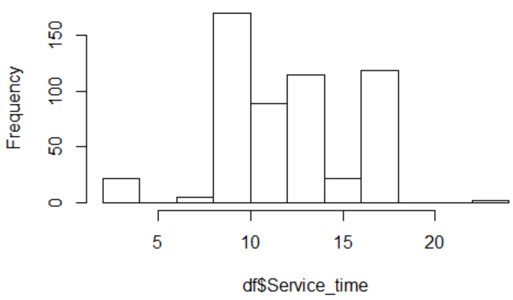
Histogram of df$Transportation_expense



Histogram of df$Distance_from_Residence_to_Work

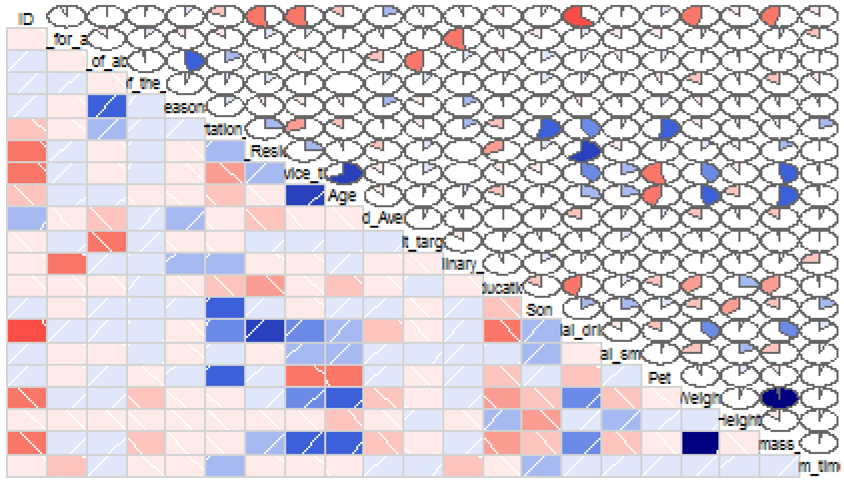


Histogram of df$Absenteeism_time_in_hours



Histogram of df$Service_time

CORRELATION PLOT



Appendix B: R Code

```
rm(list = ls())

setwd("C:/Users/Click/Desktop/EmployeeAbsenteeism_project")

getwd()

# #loading Libraries

x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "e1071",

      "DataCombine", "pROC", "doSNOW", "class", "readxl","ROSE","dplyr", "plyr",
"reshape","xlsx","pbapply", "unbalanced", "dummies", "MASS" , "gbm" ,"Information", "rpart")


# #install.packages if not

#lapply(x, install.packages)


# #load libraries

lapply(x, require, character.only = TRUE)

rm(x)


#Input Data Source

df = data.frame(read_xls('Absenteeism_at_work_Project.xls', sheet = 1))


#Creating backup of orginal data

data_Original  = df


#####################################################################

#                 EXPLORING DATA                                    #

#####################################################################


#viewing the data

head(df,4)

dim(df)
```

```
#structure of data or data types
str(df)

#Summary of data
summary(df)

#unique value of each count
apply(df, 2,function(x) length(table(x)))

#Replacing the dot b/w collumn name to underscore for easy to use
names(df) <- gsub('\\.','_',names(df))

# From the above EDA and problem statement categorising data in 2 category "continuous" and
"catagorical"
cont_vars = c('Distance_from_Residence_to_Work', 'Service_time', 'Age',
              'Work_load_Average_day', 'Transportation_expense',
              'Hit_target', 'Weight', 'Height',
              'Body_mass_index', 'Absenteeism_time_in_hours')

cata_vars = c('ID','Reason_for_absence','Month_of_absence','Day_of_the_week',
              'Seasons','Disciplinary_failure', 'Education', 'Social_drinker',
              'Social_smoker', 'Son', 'Pet')


################################################################################
#         Checking Missing data                                                #
################################################################################
apply(df, 2, function(x) {sum(is.na(x))}) # in R, 1 = Row & 2 = Col

#Creating dataframe with missing values present in each variable
```

```
null_val = data.frame(apply(df,2,function(x){sum(is.na(x))}))
null_val$Columns = row.names(null_val)
names(null_val)[1] =  "null_percentage"

#Calculating percentage missing value
null_val$null_percentage = (null_val$null_percentage/nrow(df)) * 100

# Sorting null_val in Descending order
null_val = null_val[order(-null_val$null_percentage),]
row.names(null_val) = NULL

# Reordering columns
null_val = null_val[,c(2,1)]

# Saving output result into csv file
write.csv(null_val, "MissingVal_perc_R.csv", row.names = F)



#################################################################################
#            Visualizing the data                                               #
#################################################################################

#library(ggplot2)
#Missing data percentage

#library(ggplot2)
#Missing data percentage
ggplot(data = null_val[1:18,], aes(x=reorder(Columns, -null_percentage),y = null_percentage))+
  geom_bar(stat = "identity",fill = "grey")+xlab("Variables")+
  ggtitle("Missing data percentage") + theme(axis.text.x = element_text( color="#993333",
                                        size=6, angle=90))

#################################################################
#              Data Imputation                                                #
#################################################################
#Here we are analysing the best method for imputation by trying to generate a value for existing data in
our data set.
# Actual Value = 23
# Mean = 26.68
# Median = 25
# KNN = 23

#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 Imputation
df = knnImputation(df, k = 3)

# Checking for missing value
sum(is.na(df))



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


## BoxPlots - Distribution and Outlier Check
```

```
# Boxplot for continuous variables
for (i in 1:length(cont_vars))
{
  assign(paste0("gn",i), ggplot(aes_string(y = (cont_vars[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=cont_vars[i],x="Absenteeism_time_in_hours")+
           ggtitle(paste("Box plot of absenteeism for",cont_vars[i])))
}

# ## Plotting plots together
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)

# #Remove outliers using boxplot method
# #loop to remove from all variables
for(i in cont_vars)
{
  print(i)
  val = df[,i][df[,i] %in% boxplot.stats(df[,i])$out]
  #print(length(val))
  df = df[which(!df[,i] %in% val),]
}
```

```
#Replace all outliers with NA and impute
for(i in cont_vars)
{
  val = df[,i][df[,i] %in% boxplot.stats(df[,i])$out]
  #print(length(val))
  df[,i][df[,i] %in% val] = NA
}

# Imputing missing values
df = knnImputation(df,k=3)

##############################################################
#                     Feature Selection                      #
##############################################################

#Here we will use corrgram to find corelation

##Correlation plot
#library('corrgram')

corrgram(df,
         order = F,  #we don't want to reorder
         upper.panel=panel.pie,
         lower.panel=panel.shade,
         text.panel=panel.txt,
         main = 'CORRELATION PLOT')


#We can see that the highly corr related vars in plot are marked in dark blue.
```

```
#Dark blue color means highly positive correlation

##------------------ANOVA validation_set--------------------------##

## ANOVA validation_set for Categprical variable
summary(aov(formula = Absenteeism_time_in_hours~ID,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Reason_for_absence,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Month_of_absence,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Day_of_the_week,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Seasons,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Disciplinary_failure,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Education,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Social_drinker,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Social_smoker,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Son,data = df))
summary(aov(formula = Absenteeism_time_in_hours~Pet,data = df))

## Dimension Reduction
df = subset(df, select = -c(Weight))

##############################################################
#                              Feature Scaling                              #
##############################################################

# Updating the continuous and catagorical variables
cont_vars = c('Distance_from_Residence_to_Work', 'Service_time', 'Age',
            'Work_load_Average_day', 'Transportation_expense',
            'Hit_target', 'Height',
            'Body_mass_index')
```

```
cata_vars = c('ID','Reason_for_absence','Month_of_absence','Day_of_the_week',
              'Seasons','Disciplinary_failure', 'Education', 'Social_drinker',
              'Social_smoker', 'Son', 'Pet')

#Normality check
#Checking Data for Continuous Variables

###############  Histogram   #################
hist(df$Absenteeism_time_in_hours)
hist(df$Distance_from_Residence_to_Work)
hist(df$Transportation_expense)
hist(df$Work_load_Average_day)
hist(df$Body_mass_index)
hist(df$Service_time)

#We have seen that our data is not normally distributed. Hence, we will go for Normalization.
#Normalization
for(i in cont_vars)
{
  print(i)
  df[,i] = (df[,i] - min(df[,i]))/(max(df[,i])-min(df[,i]))
}



#Creating dummy variables for categorical variables
library(mlr)
df1 = dummy.data.frame(df, cata_vars)

#############################################################
#                               Sampling of Data                                                  #
```

```
###################################################################

# #Divide data into trainset and validation_set using stratified sampling method

#install.packages('caret')
#library(caret)
set.seed(101)
split_index = createDataPartition(df$Absenteeism_time_in_hours, p = 0.66, list = FALSE)
trainset = df[split_index,]
validation_set  = df[-split_index,]

#Checking df Set Target Class
table(trainset$Absenteeism_time_in_hours)

#########################################################################################
## Basic approach for ML - Model
 ## We will first get a basic idea of how different models perform on our preprocessed data and then
select the best model and make it more efficient for our Dataset
######################################################################

#---------------------------------------------Decision tree----------------------------------------------#
#Develop Model on training data -> https://www.guru99.com/r-decision-trees.html

fit_DT = rpart(Absenteeism_time_in_hours ~., data = trainset, method = "anova")

#Summary of DT model
summary(fit_DT)

#write rules into disk
write(capture.output(summary(fit_DT)), "Rules.txt")
```

```
#Lets predict for training data
pred_DT_train = predict(fit_DT, trainset[,names(trainset) != "Absenteeism_time_in_hours"])
#rpart.plot(fit_DT, extra = 106)
# For training data
print(postResample(pred = pred_DT_train, obs = trainset[,10]))
#RMSE   Rsquared        MAE
#4.44522583 0.01624301 3.84331513

#-----------------------------------------Linear Regression------------------------------------------#
#Develop Model on training data
fit_LR = lm(Absenteeism_time_in_hours ~ ., data = trainset)
#Lets predict for training data
pred_LR_train = predict(fit_LR, trainset[,names(validation_set) != "Absenteeism_time_in_hours"])
# For training data
print(postResample(pred = pred_LR_train, obs = trainset[,10]))
#RMSE   Rsquared        MAE
#4.17968459 0.03094267 3.87604411

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

#Develop Model on training data
fit_RF = randomForest(Absenteeism_time_in_hours~., data = trainset)

#Lets predict for training data
pred_RF_train = predict(fit_RF, trainset[,names(validation_set) != "Absenteeism_time_in_hours"])

# For training data
print(postResample(pred = pred_RF_train, obs = trainset[,10]))

#     RMSE   Rsquared        MAE
# 4.52959371 0.01351095 3.87059587

#------------------------------------------XGBoost------------------------------------------#

#Develop Model on training data
fit_XGB = gbm(Absenteeism_time_in_hours~., data = trainset, n.trees = 500, interaction.depth = 2)

#Lets predict for training data
pred_XGB_train = predict(fit_XGB, trainset[,names(validation_set) != "Absenteeism_time_in_hours"], n.trees = 500)

# For training data
print(postResample(pred = pred_XGB_train, obs = trainset[,10]))

#      RMSE   Rsquared        MAE
#4.53665834 0.01470712 3.86070501


#-------------------------------------------Decision tree for classification-------------------------------------------------#

#Develop Model on training data
fit_DT = rpart(Absenteeism_time_in_hours ~., data = trainset, method = "anova")

#Lets predict for training data
pred_DT_train = predict(fit_DT, trainset)

# For training data
print(postResample(pred = pred_DT_train, obs = trainset$Absenteeism_time_in_hours))

#     RMSE  Rsquared       MAE
# 2.3747264 0.4801571 1.6626688

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

#Develop Model on training data
fit_LR = lm(Absenteeism_time_in_hours ~ ., data = trainset)

#Lets predict for training data
pred_LR_train = predict(fit_LR, trainset)

# For training data
print(postResample(pred = pred_LR_train, obs = trainset$Absenteeism_time_in_hours))

#RMSE Rsquared      MAE
#2.8151420 0.2694573 2.0619282

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

#Develop Model on training data
fit_RF = randomForest(Absenteeism_time_in_hours~., data = trainset)

#Lets predict for training data
pred_RF_train = predict(fit_RF, trainset)
# For training data
print(postResample(pred = pred_RF_train, obs = trainset$Absenteeism_time_in_hours))

#   RMSE Rsquared      MAE
#1.4310323 0.8522856 1.0158687

#------------------------------------------------XGBoost-----------------------------------------------#
```

```
#Develop Model on training data
fit_XGB = gbm(Absenteeism_time_in_hours~., data = trainset, n.trees = 500, interaction.depth = 2)

#Lets predict for training data
pred_XGB_train = predict(fit_XGB, trainset, n.trees = 500)

# For training data
print(postResample(pred = pred_XGB_train, obs = trainset$Absenteeism_time_in_hours))

#    RMSE Rsquared      MAE
#1.804408 0.706962 1.312741
#############################################################################################
                                              Saving output to file
#############################################################################################
#write.csv(submit,file =
'C:/Users/Click/Desktop/EmployeeAbsenteeism_project/FinalAbsenteeism_R.csv',row.names = F)
rm(list = ls())
```

References

- https://stackoverflow.com/questions/10438752/adding-x-and-y-axis-labels-in-ggplot2
- Gradient boosting in R | DataScience+datascienceplus.com
- How to Rename Columns in the Pandas Python Librarychartio.com
- https://docs.python.org/3.4/library/statistics.html
- https://seaborn.pydata.org/generated/seaborn.heatmap.html
- https://www.researchgate.net/post/what_are_the_best_methods_for_filling_in_missing_values
- https://scikit-learn.org/0.15/auto_examples/imputation.html
- https://scikit-learn.org/0.15/auto_examples/imputation.html
- https://datascience.stackexchange.com/questions/24989/imputing-for-multiple-missing-variables-using-sklearn
- https://datascience.stackexchange.com/questions/24989/imputing-for-multiple-missing-variables-using-sklearn