Employee Absenteeism Project

Madhurima Biswas

29 November 2018

Contents

I. Introduction:

1.1. Problem Statement

Absenteeism is a serious workplace problem and an expensive occurrence for both employers and employees. Absenteeism of employees from work leads to back logs, piling of work and thus work delay. Human capital plays an important role in collection, transportation and delivery at XYZ Courier Company. The company is passing through genuine issue of Absenteeism. The aim of the project is to reduce the number of absenteeism amongst employees of company XYZ by bringing necessary changes accordingly. We would like to predict the losses that will occur every month if the same trend of absenteeism continues in the year 2011.

1.2. Data

The task is to build predictive regression models, which will predict the employees' absenteeism, based on the various factors given in the data of the XYZ Courier Company.

Given below is a sample of the data set that we are using to predict the employee absenteeism:

Table 1.1: Absenteeism at work sample data (XYZ Company)

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
11	26	7	3	1	289	36	13	33	239,554
36	0	7	3	1	118	13	18	50	239,554
3	23	7	4	1	179	51	18	38	239,554
7	7	7	5	1	279	5	14	39	239,554
11	23	7	5	1	289	36	13	33	239,554

Hit target	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Weight	Height	Body mass index	Absenteeism time in hours
97	0	1	2	1	0	1	90	172	30	4
97	1	1	1	1	0	0	98	178	31	0
97	0	1	0	1	0	0	89	170	31	2
97	0	1	2	1	1	0	68	168	24	4
97	0	1	2	1	0	1	90	172	30	2

Therefore, in the table below, we have the following 20 independent variables, using which we have to predict the employee Absenteeism:

Table 1.2: Predictor Variables

No.	Independent 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

We have 11 numerical and 9 categorical independent variables in this data.

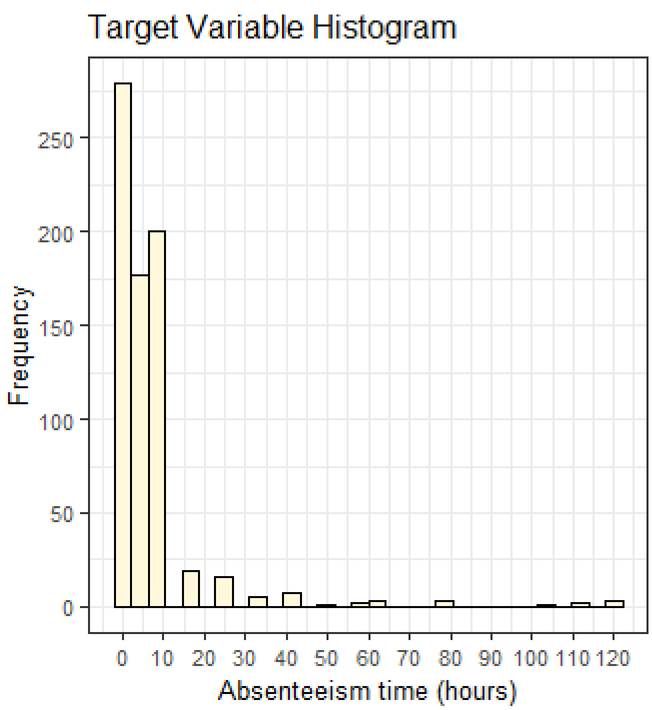
II. Methodology

2.1. Exploratory Data Analysis:

The objective first is to study each feature available in the data and try to assess some patterns and understand the dimensions & properties of the data by exploring it visually. It helps us in understanding the nature of data in terms of distribution of the individual variables/features, finding missing values, relationship with other variables and many other things.

2.1.1. Univariate Analysis:

A. Dependent Target Variable: "Absenteeism time in hours"



Since our target variable is continuous, we can visualize it by plotting its histogram.

Fig.2.1

Observation:

- Most of the employees have taken leave of short durations, but in the long run, it adds up to the absenteeism problem. It is a right (positively) skewed variable and can be transformed to treat its skewness using logarithmic, cube root or square root function.
- Removing outliers might help reduce the skewness in data as well.

B. Independent Numeric Variables:

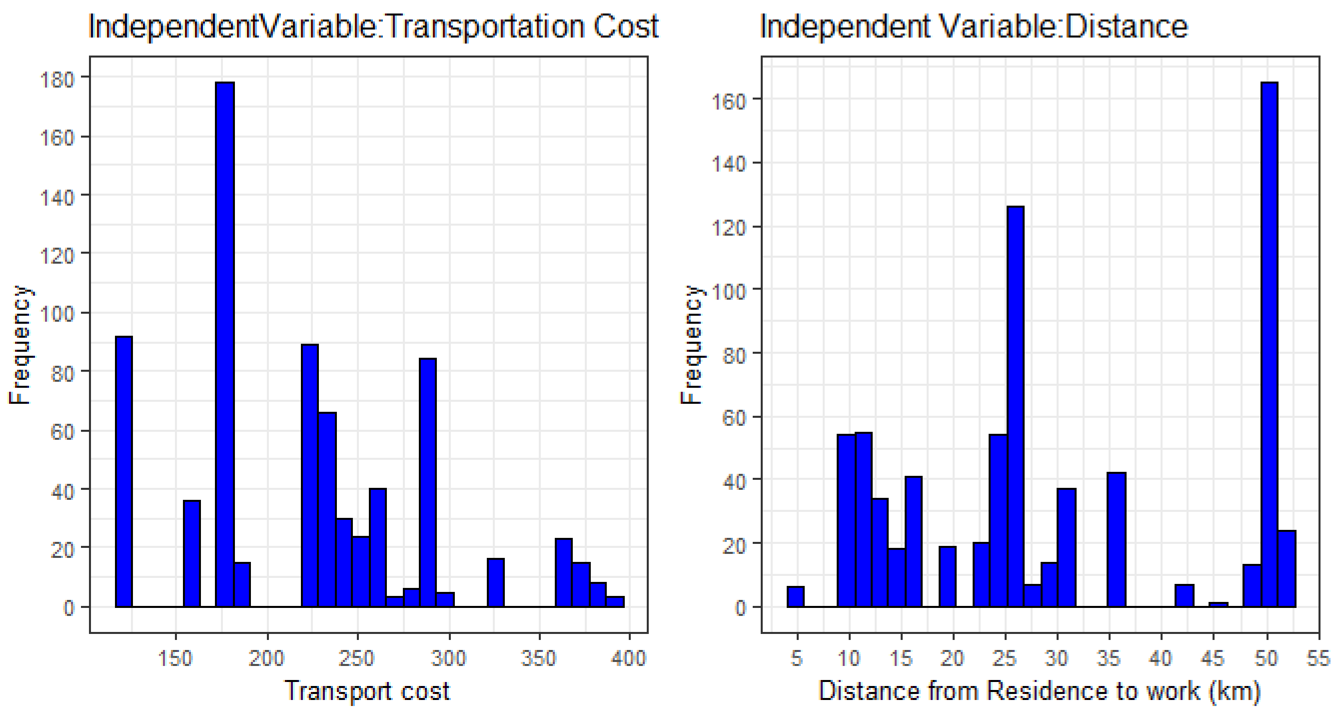


Fig.2.2

Observation:

- The mean transportation expense of the employees is 221.03 (rs.) and mean distance from residence to work is 29.67 km.
- There is no clear-cut pattern in “transportation Expense” & “Distance from residence to work” variable.

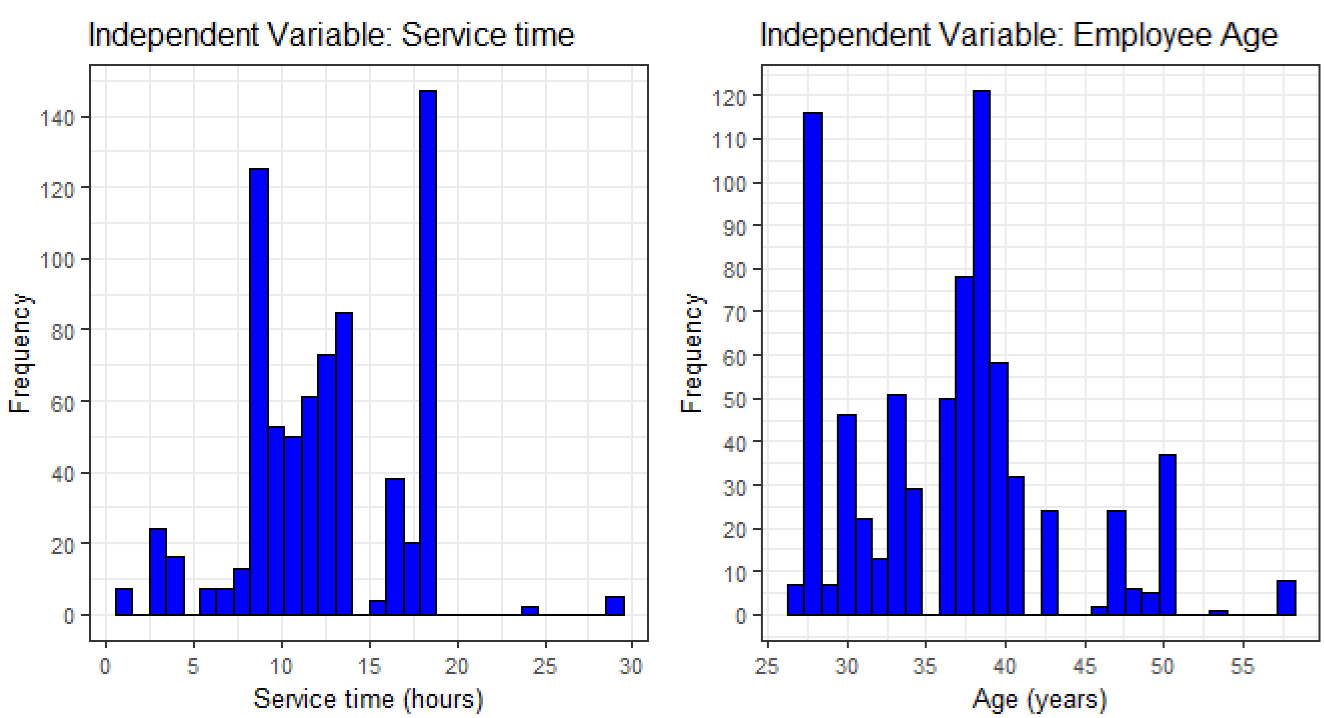


Fig.2.3

Observation:

- Most of the absentees have a "Service time" around the median hour of 13 hours and an average of 12.56 hours, i.e. employees with high or less service time have not taken much leaves comparatively.
- Most of the absentees are below the "Age" of 50, i.e. around the median of 37 years. However, this could be because they hire less old employees working in this company as there is need for active members who can collect, transport & delivery courier and therefore, we have to look for more relations.

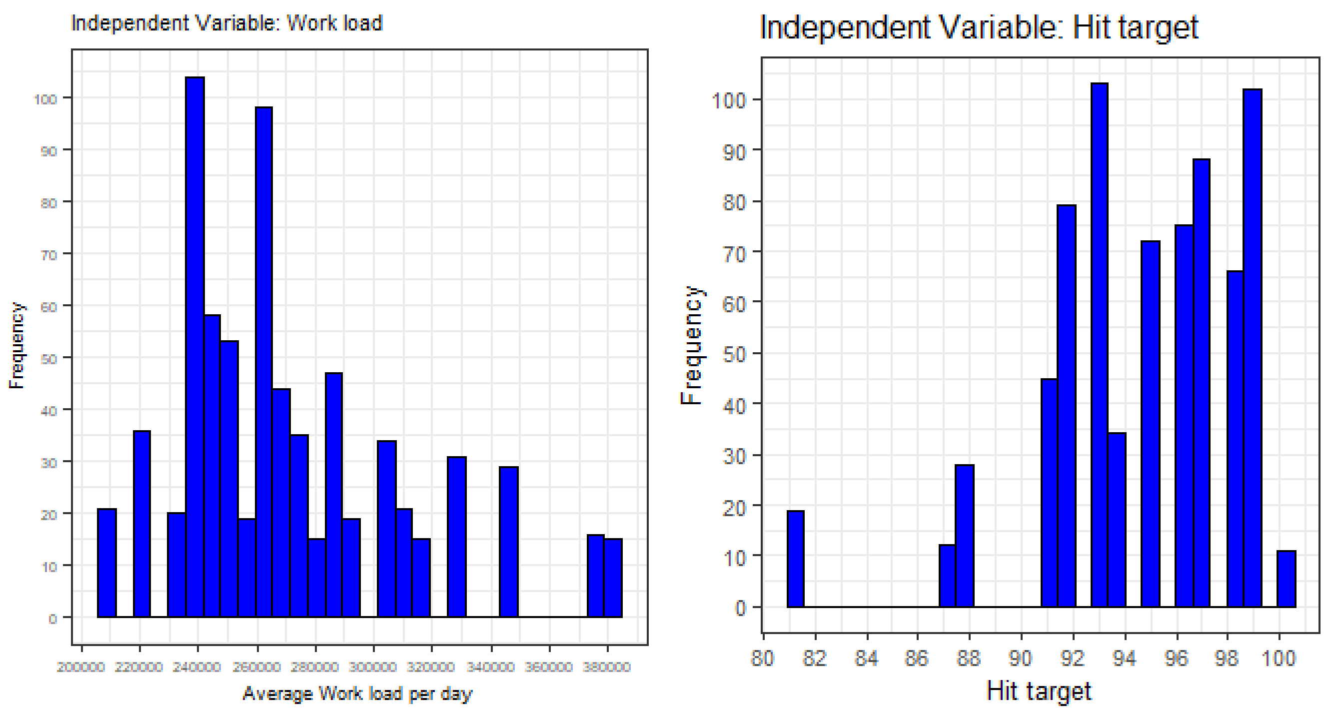


Fig.2.4

Observation:

- Employees with moderately low "Average work load per day" were absent having an average of 271188.8.
- It can be clearly seen that employees with high "Hit Target" were more absent from work comparatively. Hence, the employer can look into minimizing their employee's targets.

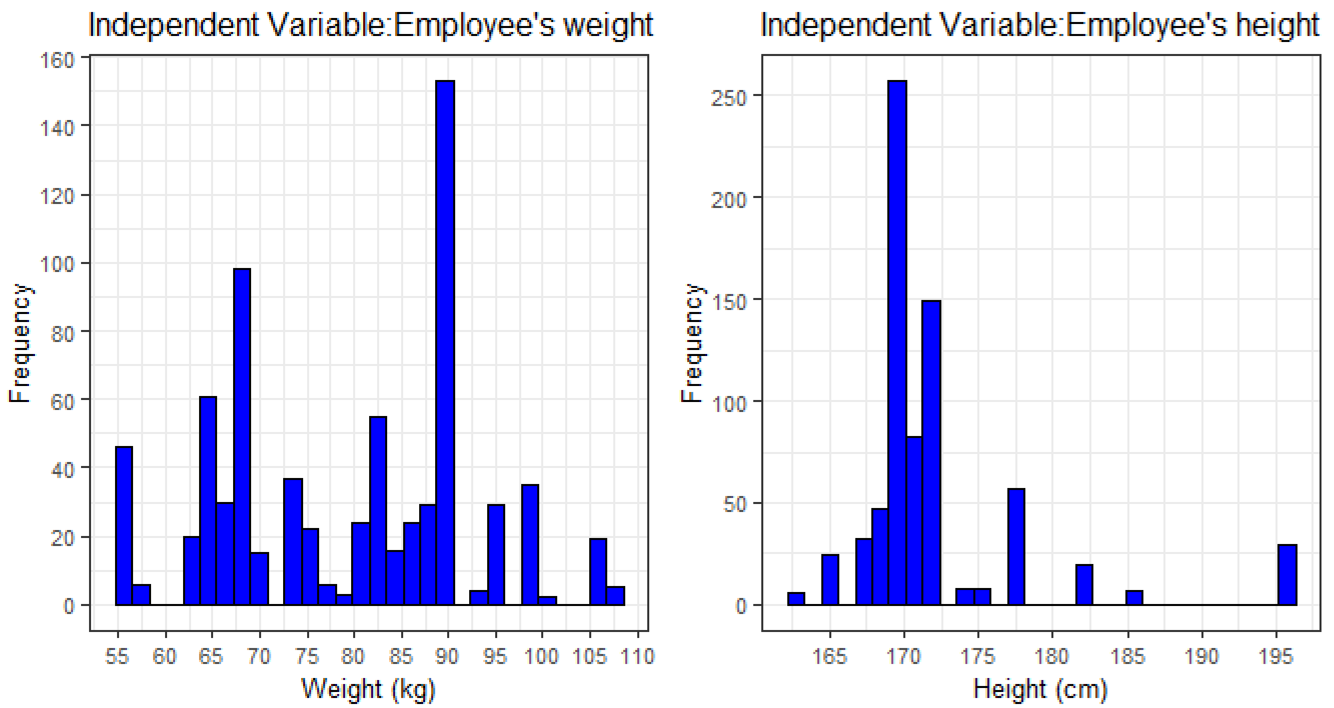


Fig.2.5

Observation:

- There is no clear-cut pattern in “Employee’s weight”.
- There is right skewness towards low heighted employees and this can be reduced by outlier analysis.

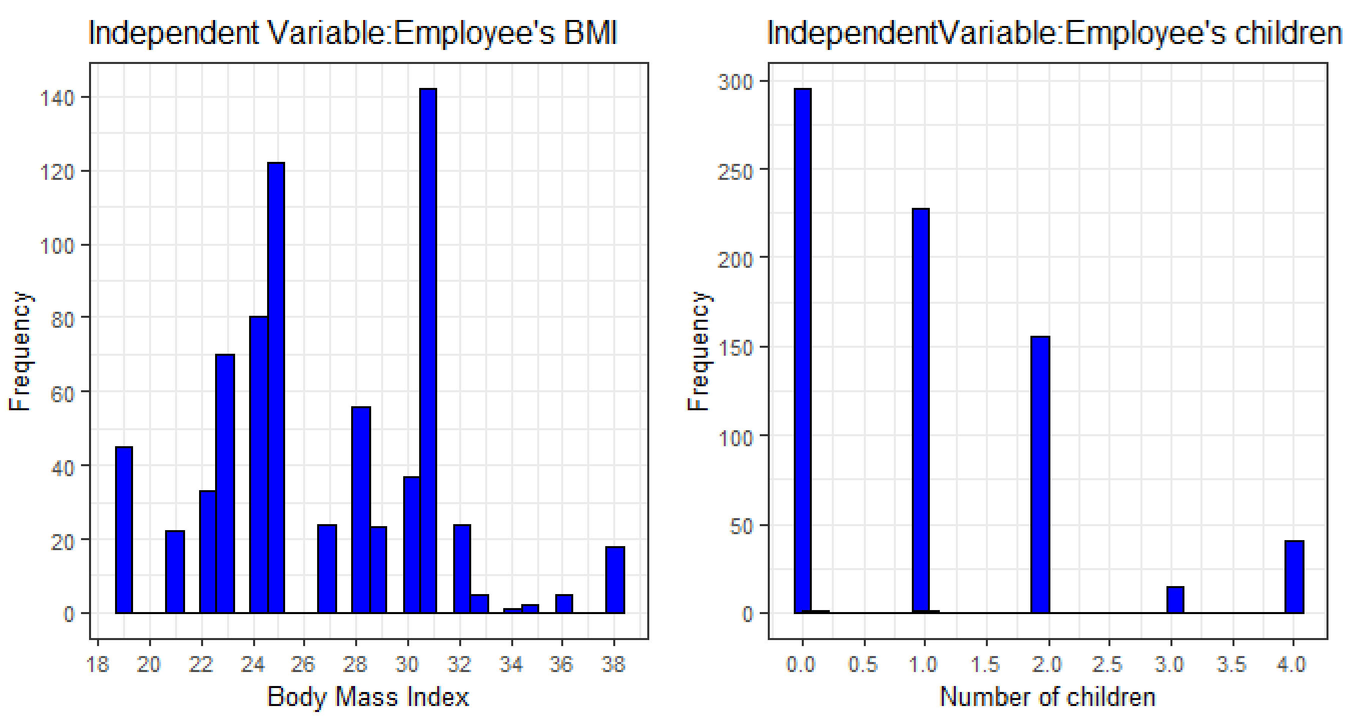


fig.2.6

Observation:

- Normal BMI is considered between 18.5 to 25. The average "BMI" for absentees is 26.7, which comes under the category of 'slightly overweight'. The XYZ Company can therefore create some fitness awareness or routines for their employees.
- Most of the absentees have no "Son" or just one in number. The median of this variable is 1.

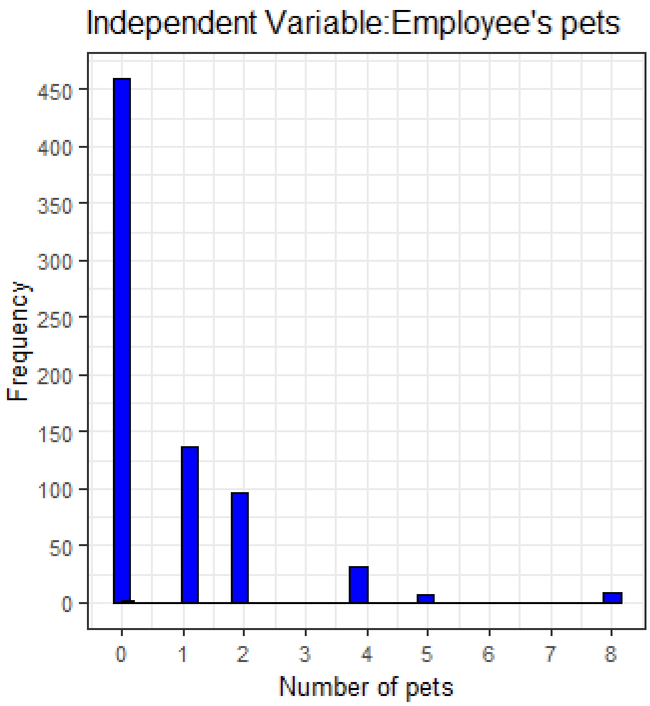


fig.2.7

Observation:

- Most of the employees who were absent have no pet, median & mode being '0'. However, we cannot say that responsibility of a pet at home could be necessarily a generic reason of absence in this company.

C. Independent Categorical Variables:

Bar graphs for the categorical variables in the data as follows:

Fig.2.8

Observation:

- The bar graph for "ID" tells us about which employees were absent the most. Thus, recognizing the most frequent absentees and helping the company heads to micromanage them.
- The mode of "Reason of Absence" variable is '23: Medical Consultation' and 2nd highest mode is '28: Dental Consultation'. So, if the company can arrange regular health checkups near or in their company premises, the losses due absenteeism can be reduced.
- In the "Reason of Absence" variable, there is a category '0', which is an anomaly in data as it is not mentioned in either ICD reasons or without ICD ones.

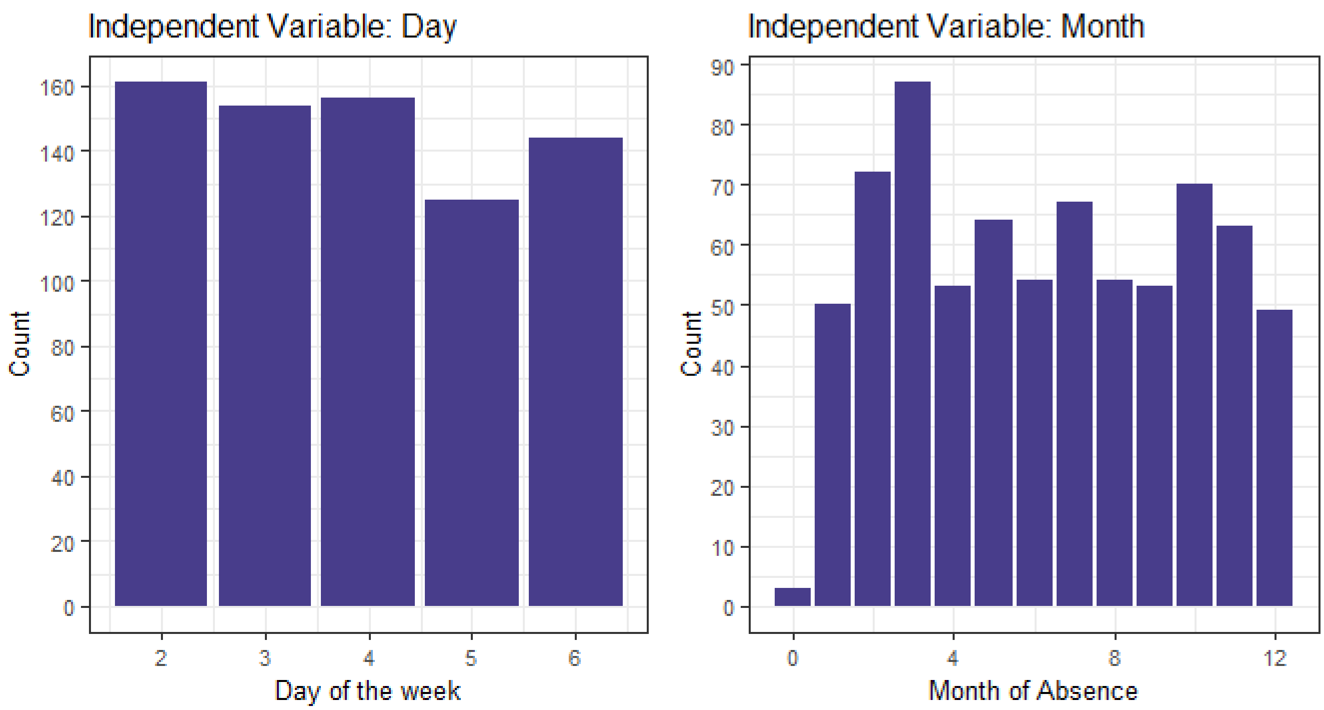


Fig.2.9

Observation:

- There is no clear-cut pattern in both "Day of the week" & "Month of Absence" variables.
- There is anomaly in the "Month of Absence" variable as '0'Th month is not a valid month.

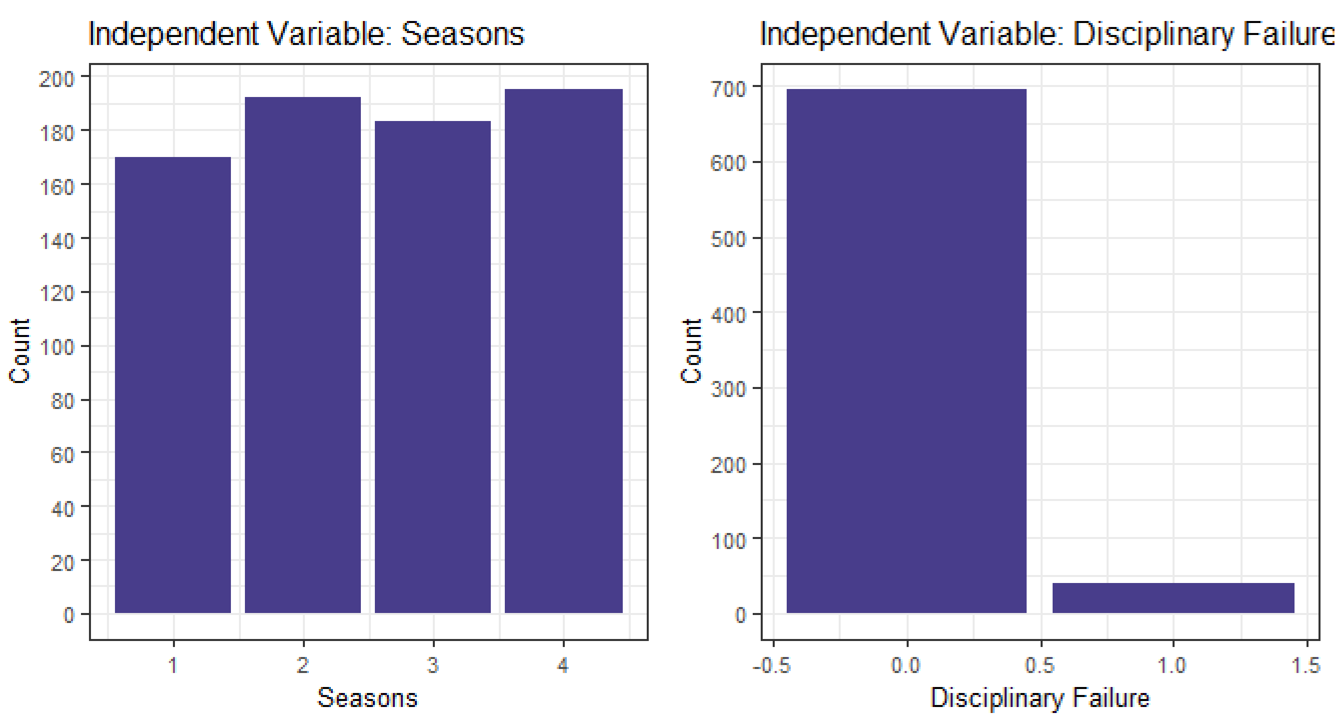


Fig.2.10

Observation:

- There is no clear-cut pattern in “Seasons” variable.
- There are very few absentees with disciplinary failure. This, in general, is a positive thing for the company, but this may or not be related to absenteeism issue. Let’s see further.

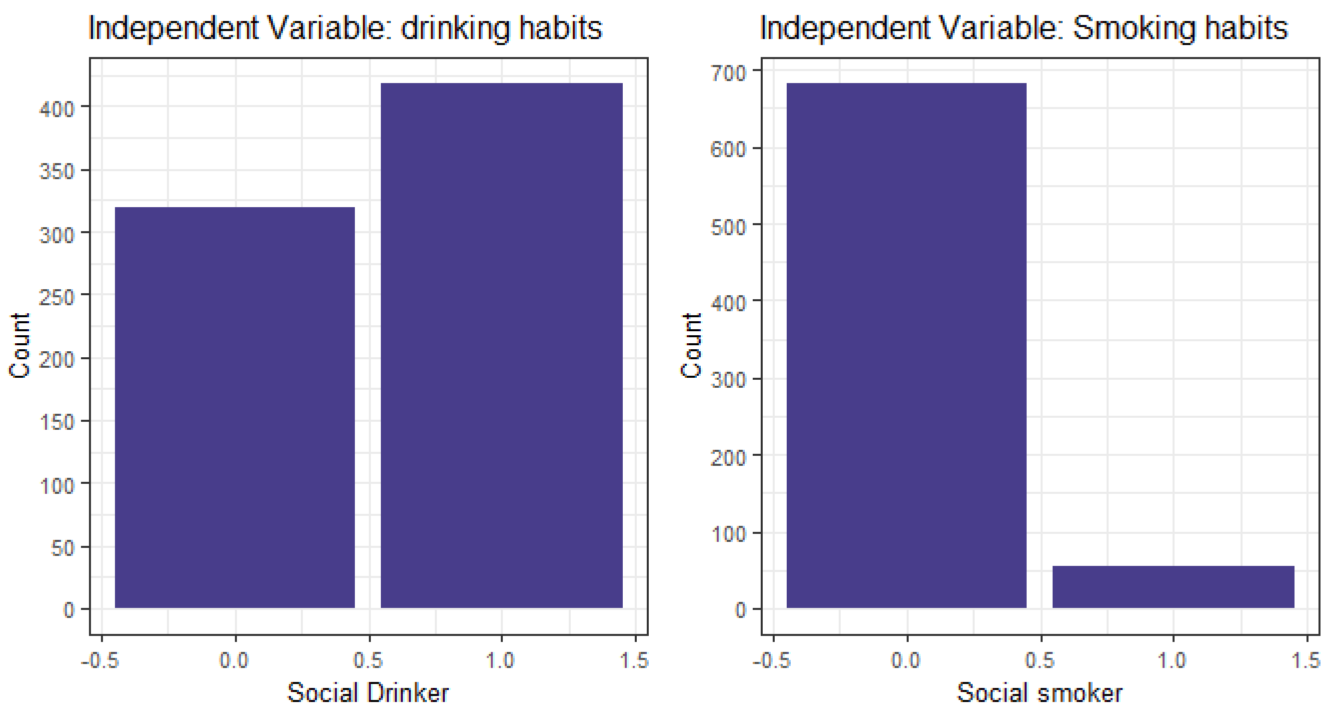
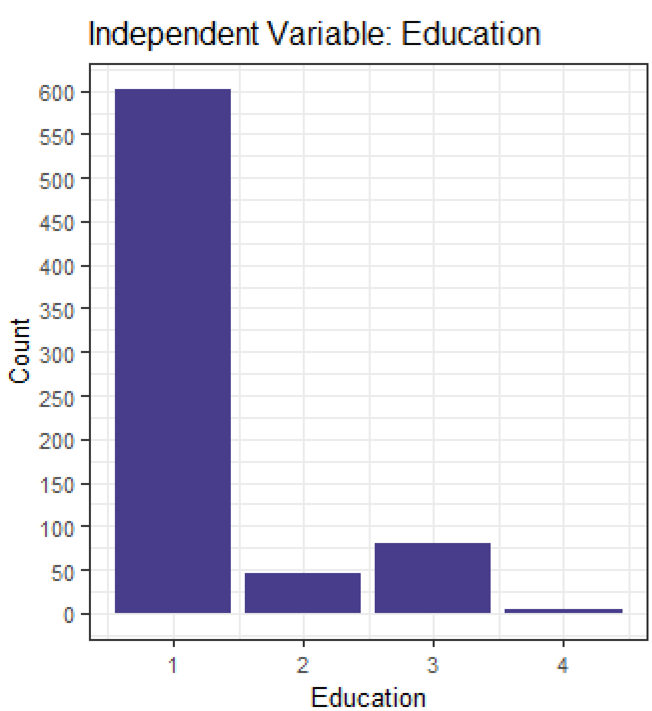


Fig.2.11

Observation:

- There is no clear-cut pattern in the “social drinker” variable.
- Most of the absentees are non-smokers.



Observation:

- Most of the absentees are high school pass outs and very few amongst them have gone for higher studies.

Fig.2.12

2.1.2. Bivariate Analysis:

After looking features individually, let’s explore the independent variables with respect to the target variable using scatter plots to discover hidden relationships between the independent variable and the target variable and use those findings in missing data imputation and feature engineering.

A. Dependent target Variable Vs Independent Variables:

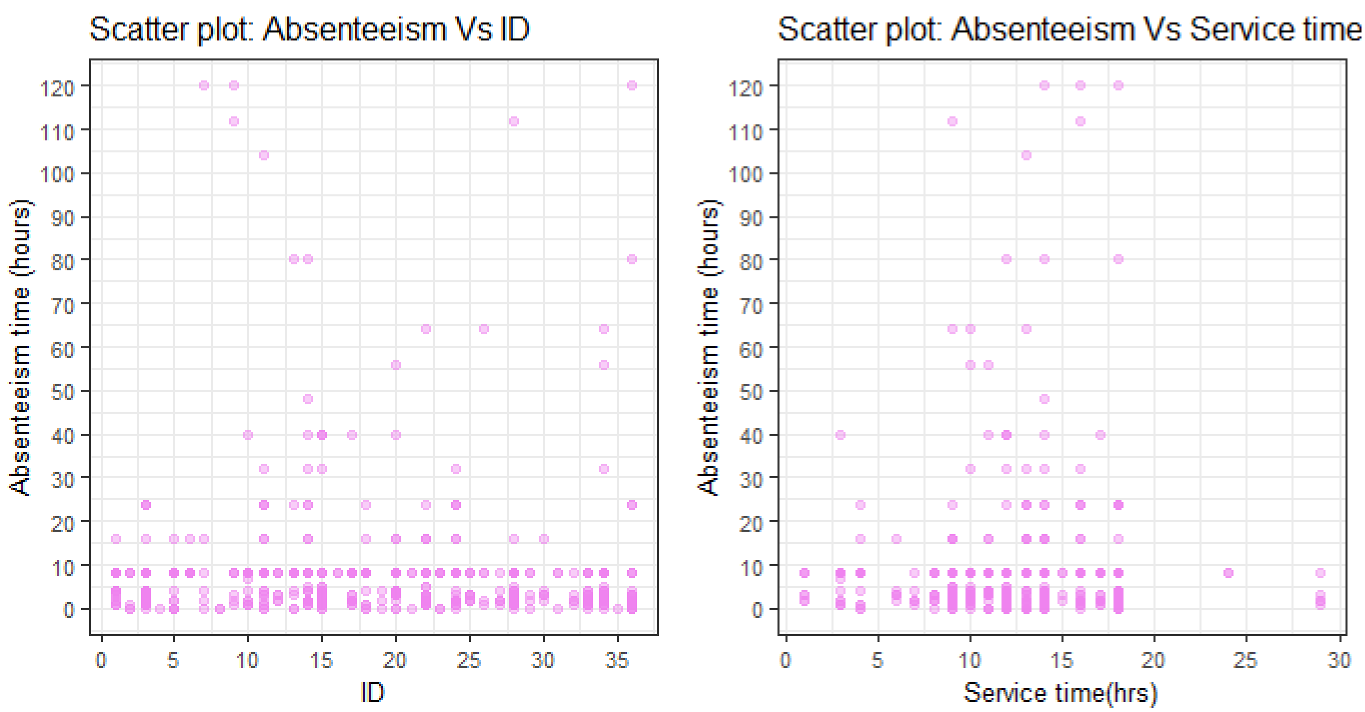
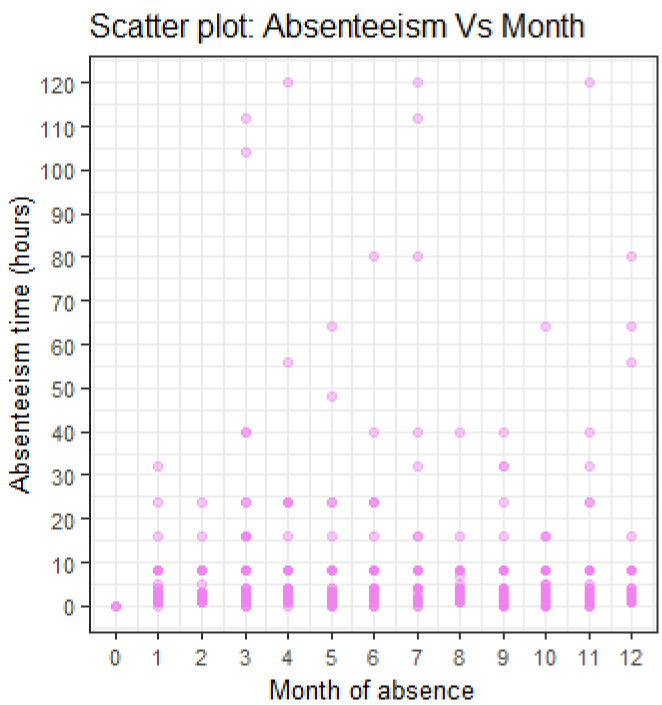
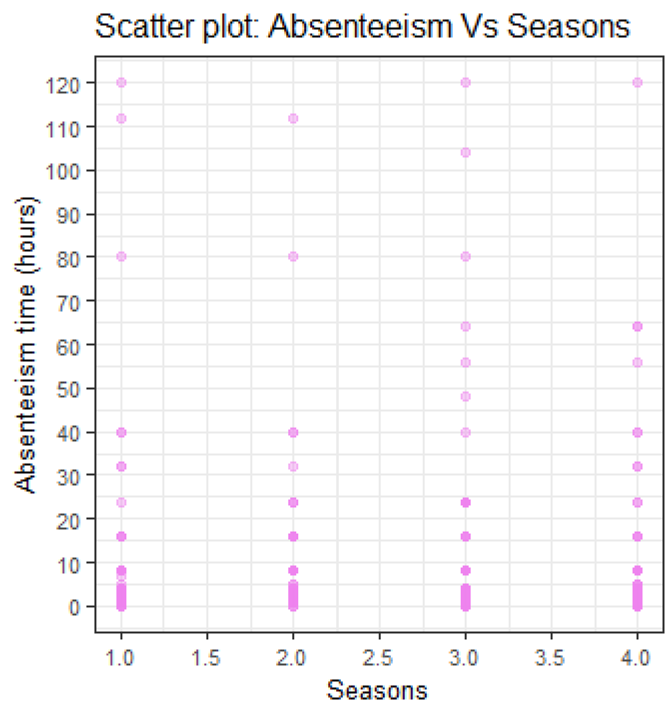


fig.2.13

Observation:

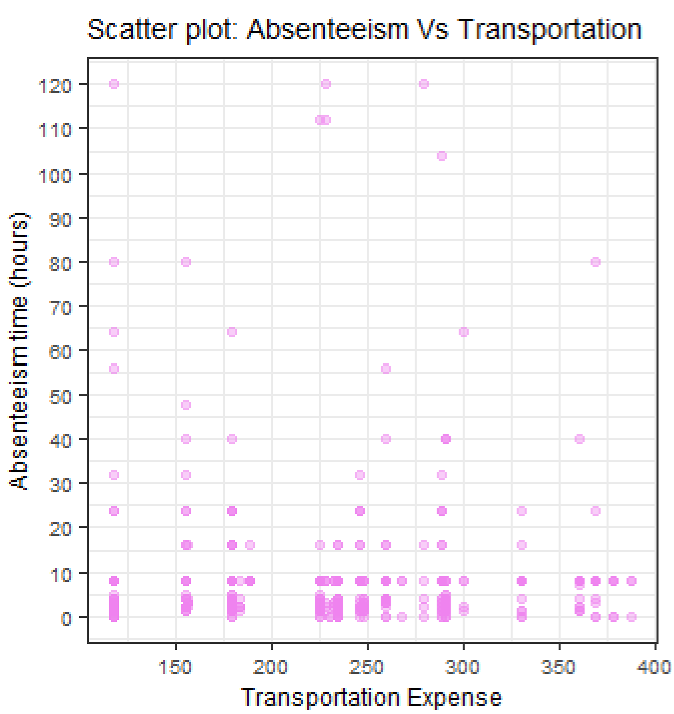
- “Absenteeism time in hours” target variable Vs “ID” can help the company to manage absentees who took a long period of leave at a time and to counsel or warn them, if needed.
- There are more & high absents observed in the mid-range of “Service time”.

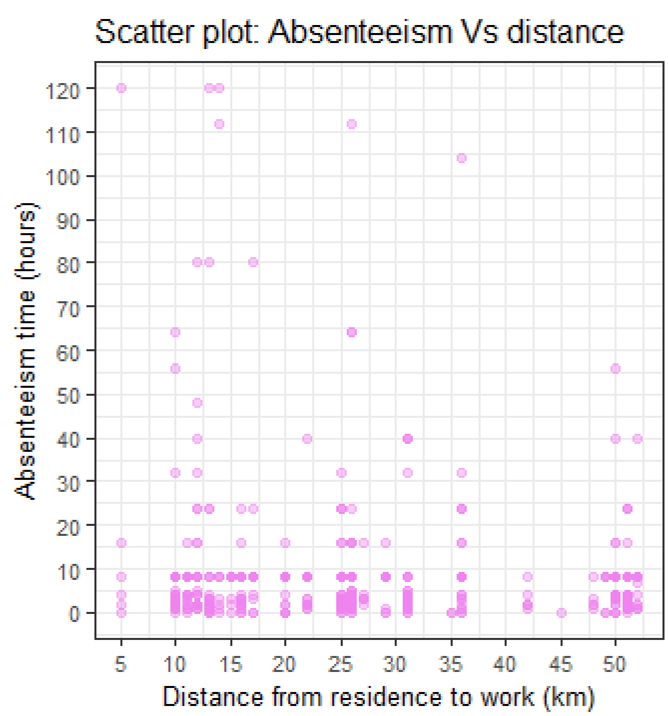




Observation:

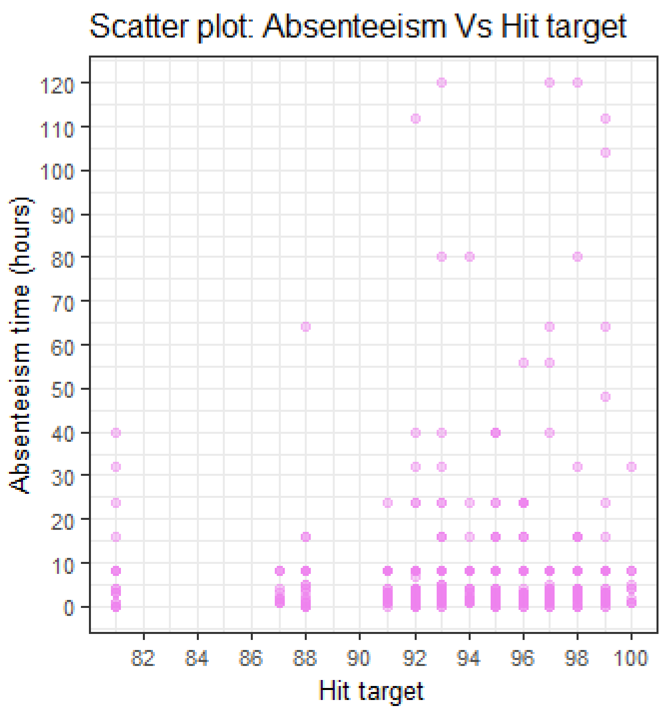
- "Month of Absence" & "Seasons" are quite spread well across the entire range of the Absenteeism time without any obvious pattern.

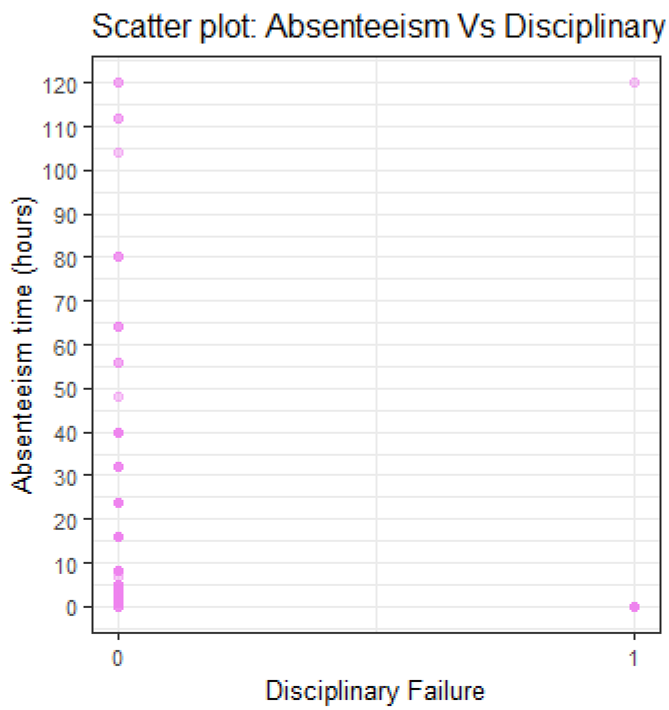




Observation:

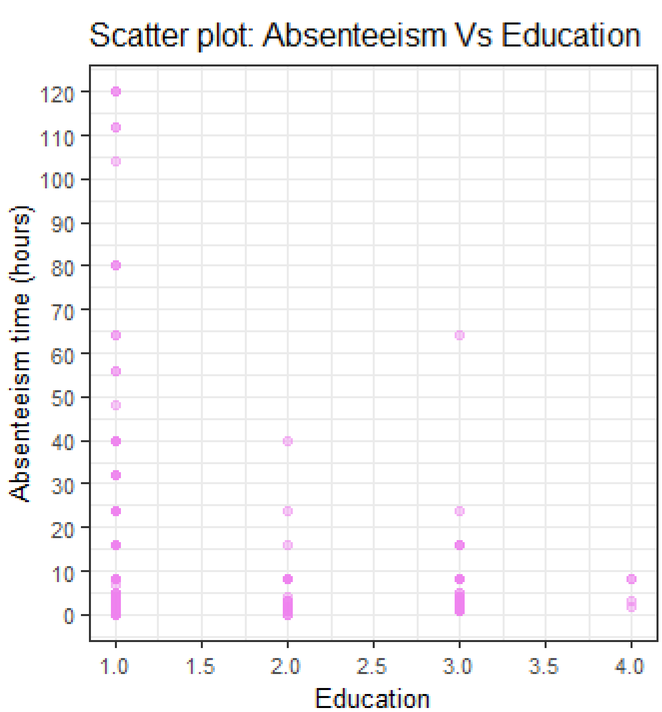
- Intuitively, it is expected that the absenteeism will increase with the increase in "distance from work" and "transportation expense", however, that is not the case for this company.

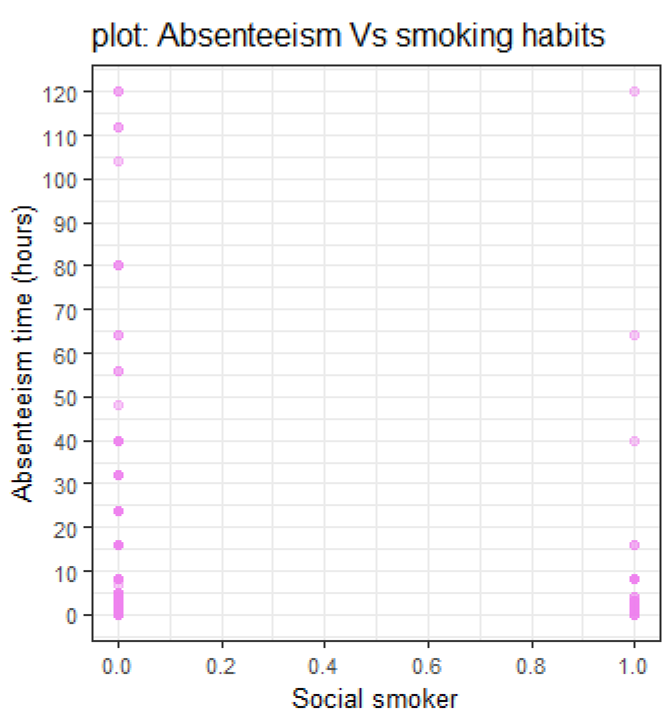




Observation:

- It can be noticed that absentees of this company have high hit targets, and therefore the employer might reconsider their individual targets for the future.
- Most of the absents are those having no disciplinary failure in their record.



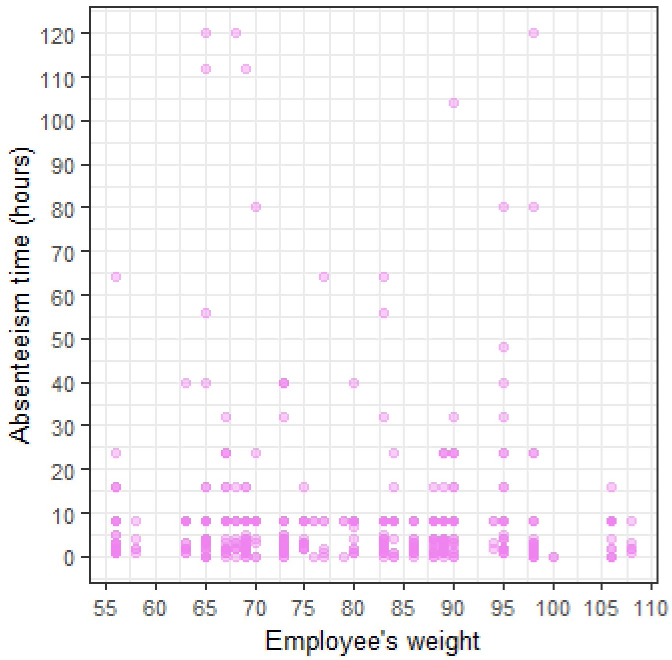


Observation:

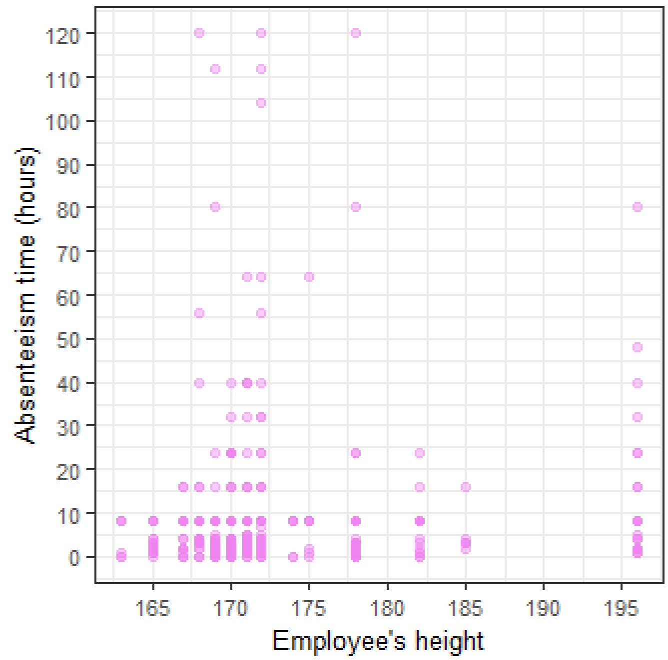
- Most the absentees are high school graduates, whereas employees with higher education degrees tend to take less leaves in this company.

- There is a pattern in “social smoker” vs Absenteeism that non-smokers are more absent.

Scatter plot: Absenteeism Vs Weight



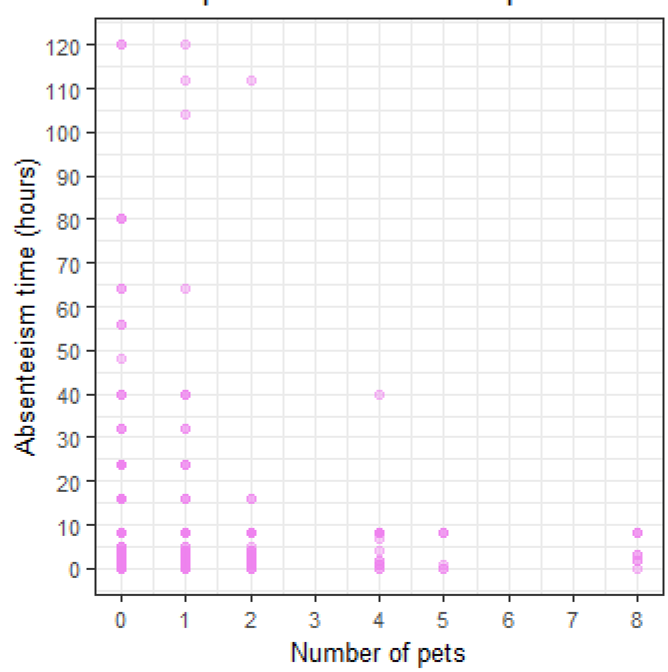
Scatter plot: Absenteeism Vs Height



- There is no obvious pattern in “Weight” Vs Absenteeism time.
- We can see that more absentees are short to medium heighted. Heights above 190cm looks like outliers.

Scatter plot: Absenteeism Vs BMI

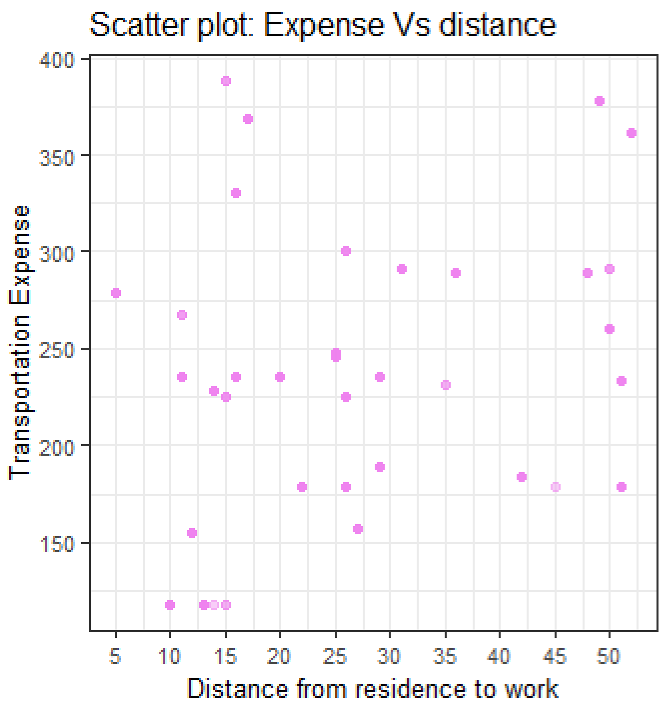
Scatter plot: Absenteeism Vs pets

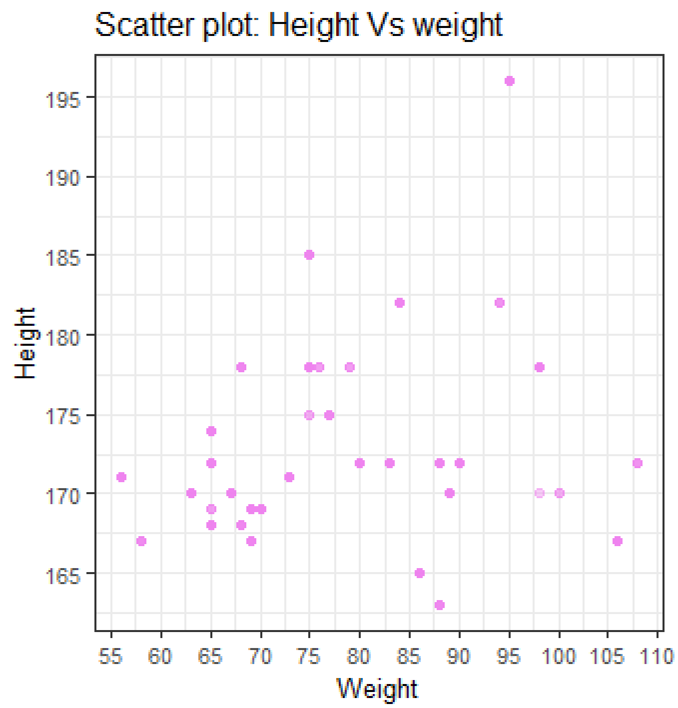


- ‘Body Mass Index” Vs “Absenteeism time” shows no such pattern except that most of the absentees have slightly obese BMI range.
- Most of the absentees have no or few pets at home.

B. Independent Variable Vs Independent Variable (Interdependencies):

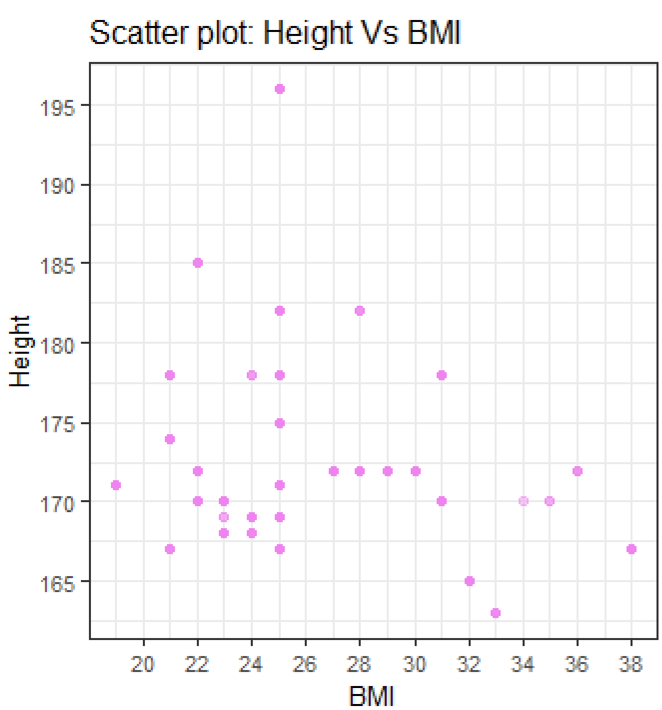
Let's explore the independent variables with respect to other independent variables using scatter plots to discover hidden relations or dependencies between them.

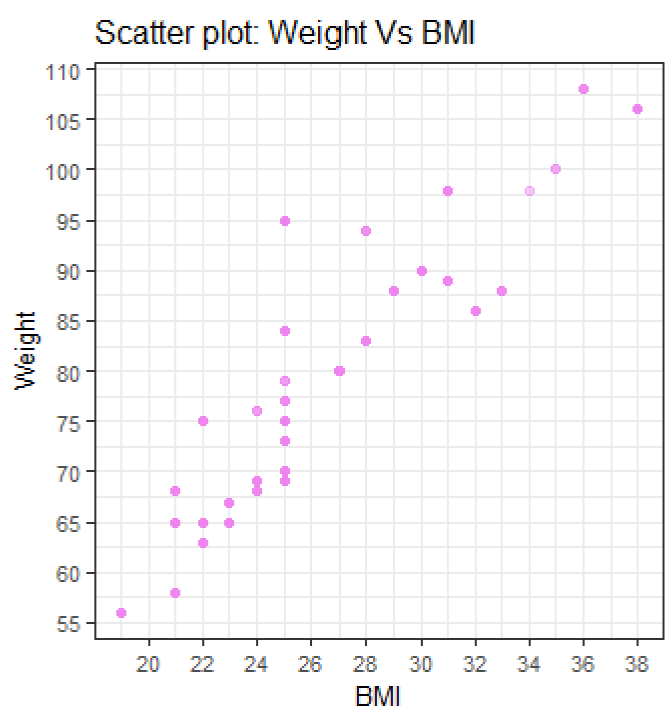




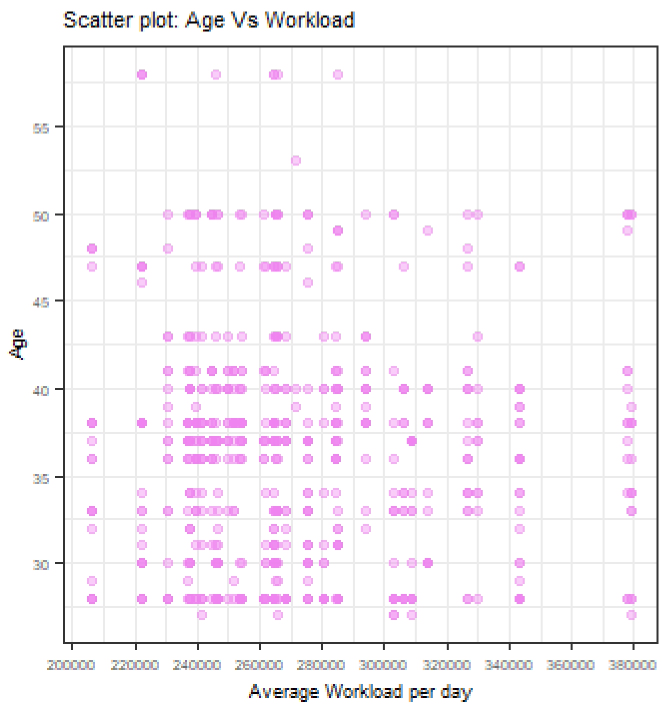
Observation:

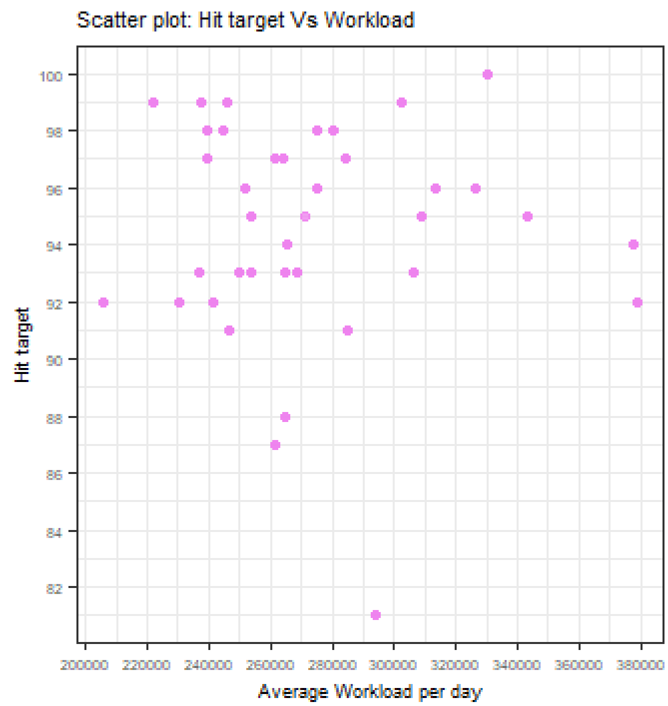
- Intuitively, it is expected that Transportation expense will roughly increase with increasing distance, but that is not the case here. Probably, the employees have different modes of transport.
- 'Height' & 'Weight' do not seem correlated much.



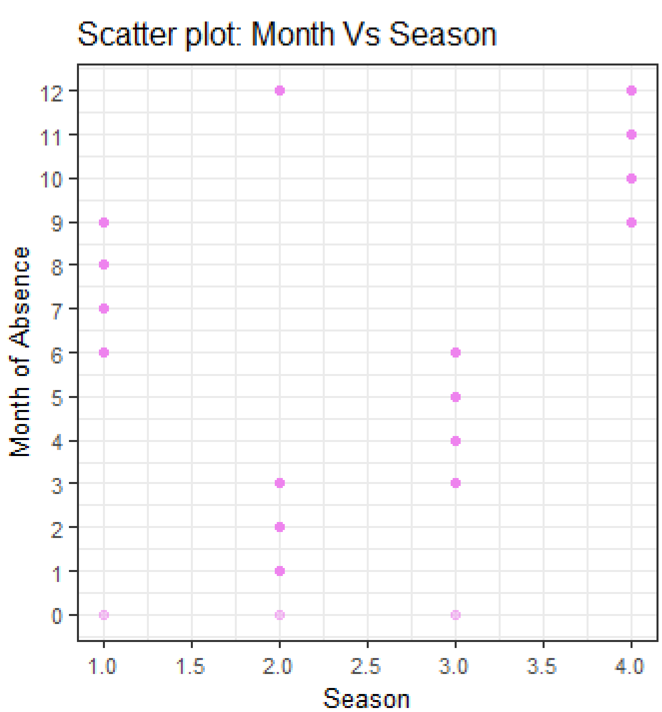


- Body Mass Index or BMI formula is proportional to body weight & inversely proportional to the square of height. There is a clear positive linear relationship between weight and BMI in this data and therefore they are redundant variables.
- On the other hand, it is hard to say anything about the pattern between height and BMI for this dataset.





- Most absentees have a higher Hit Target, but low to medium Average workload per day comparatively.
- Comparatively, younger absentees with low-medium average workload/day are more.



- The "Seasons" tend to stay for 4 months in a year, after which they disperse into a new season. This transition of season in a month is smoother and hence two seasons may overlap in a month. However, there is anomaly at 0^{th} month where 3 seasons overlap. We need to fix this before data processing.

2.1.3. Fixing Data Anomalies:

Since all data are in numeric type, even the categorical ones, we do not need to convert it for data consolidation.

From the pre-processing and visualizing the variables, it has been found that two variables, i.e. "Reason of Absences" & "Month of Absence", have invalid zero values in them, so we can treat them as missing values and them impute them further accordingly.

```
zero_index = which(data$Reason.for.absence == 0)
for(i in zero_index){
  data$Reason.for.absence[i]= NA
}
zero_index = which(data$Month.of.absence == 0)
for(i in zero_index){
  data$Month.of.absence[i]= NA
}
```

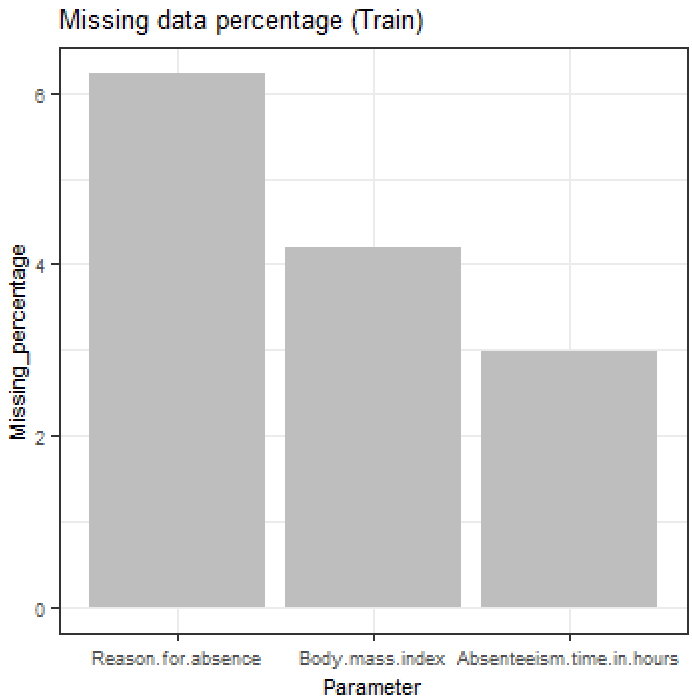
2.1.4. Missing Value Analysis:

Missing data can have a severe impact on building predictive models because the missing values might be contain some vital information, which could help in making better predictions. So, it becomes imperative to carry out missing data imputation.

The total missing values in the data are computed as follows:

	Columns	Missing_percentage	NumberOfMissingValues
1	Reason.for.absence	6.2162162	46
2	Body.mass.index	4.1891892	31
3	Absenteeism.time.in.hours	2.9729730	22
4	Height	1.8918919	14
5	Work.load.Average.day.	1.3513514	10
6	Education	1.3513514	10
7	Transportation.expense	0.9459459	7
8	Hit.target	0.8108108	6
9	Disciplinary.failure	0.8108108	6
10	Son	0.8108108	6
11	Month.of.absence	0.5405405	4
12	Social.smoker	0.5405405	4

The highest missing value percentage is about 6%, which is less than 10% and therefore can be acceptably replaced with statistical approximations or using KNN imputation. To find the suitable method of imputation, a missing value is created and mean/median/KNN imputation methods applied to find the closest predicted value to the actual value.




```
##To test for the best method to find missing values for this dataset
data[6,6]=179
data[6,6]=NA
#By median method:
data$Transportation.expense[is.na(data$Transportation.expense)]=median(data$Transportation.expense,
na.rm = T)
#data[6,6]=225 (median)
#By mean method:
#reupload data
data[6,6]=NA
data$Transportation.expense[is.na(data$Transportation.expense)]=mean(data$Transportation.expense,
na.rm = T)
#data[6,6]=221.129 (mean)
#By KNN Imputation:
#reupload data
data[6,6]=NA
data=knnImputation(data, k=5)
#data[6,6]=179 (KNN) , which is the closest to 179 (actual value) and hence, we freeze this method.
```

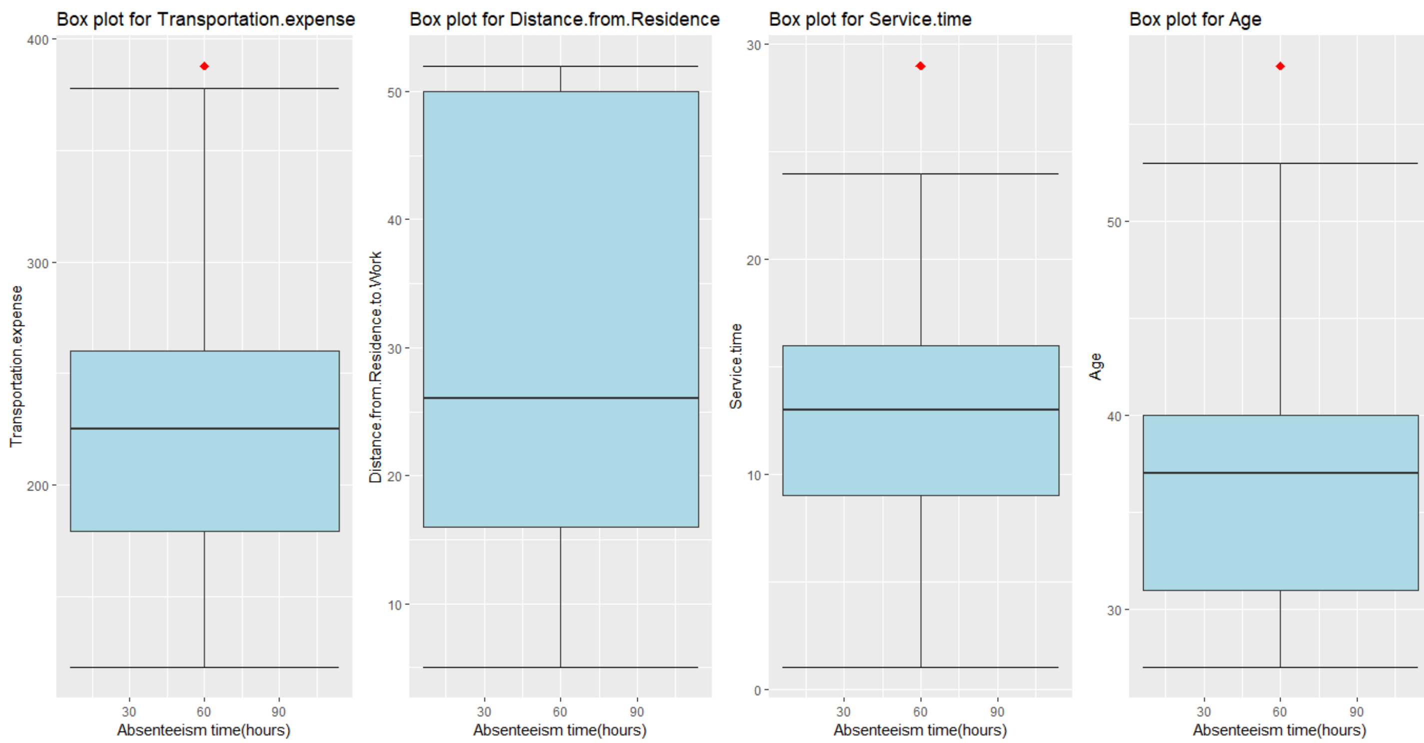
The missing data was successfully imputed in the features using KNN imputation.

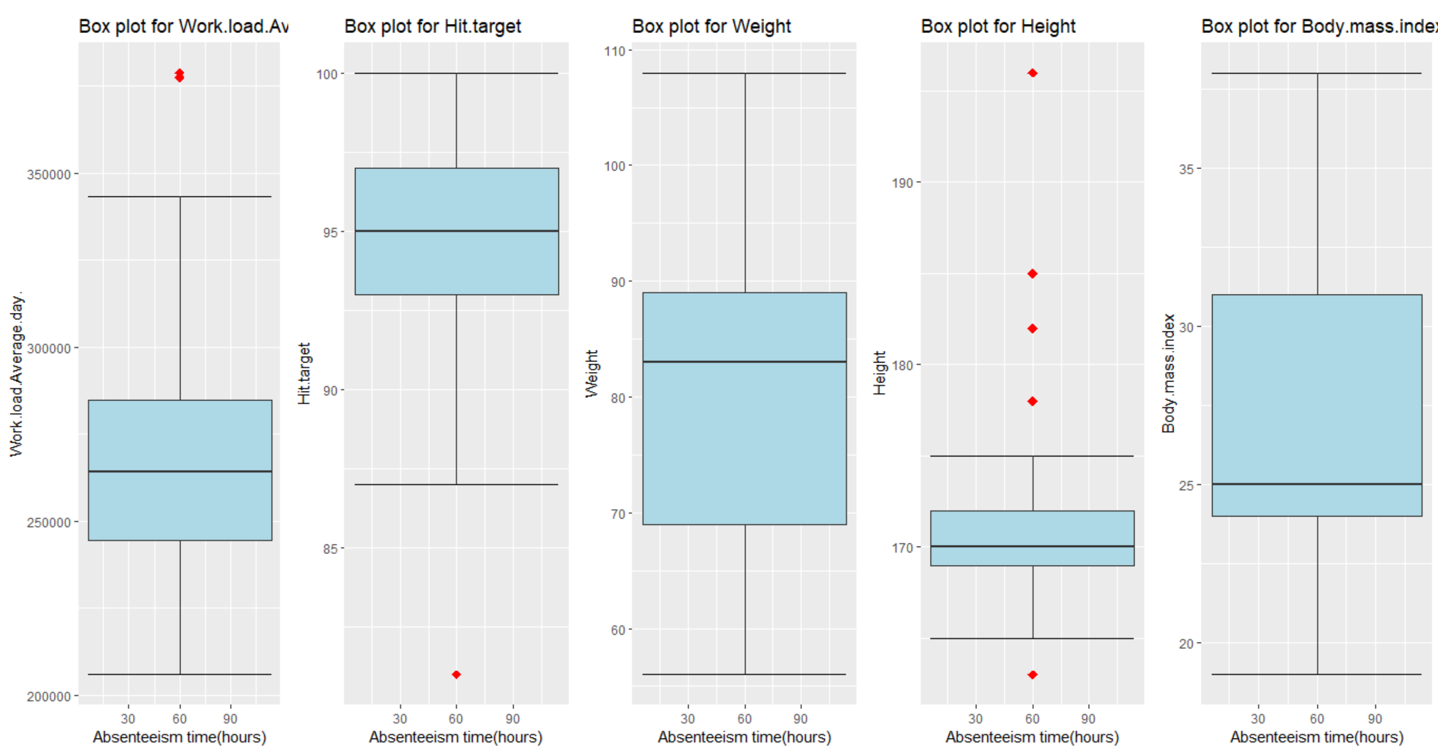
2.1.5. Outlier Analysis:

By definition, outliers are points that are distant from remaining observations. As a result, they can potentially skew or bias any analysis performed on the dataset. It is therefore important to detect and adequately deal with outliers.

We draw box plots to check the distribution & outliers in the dataset.

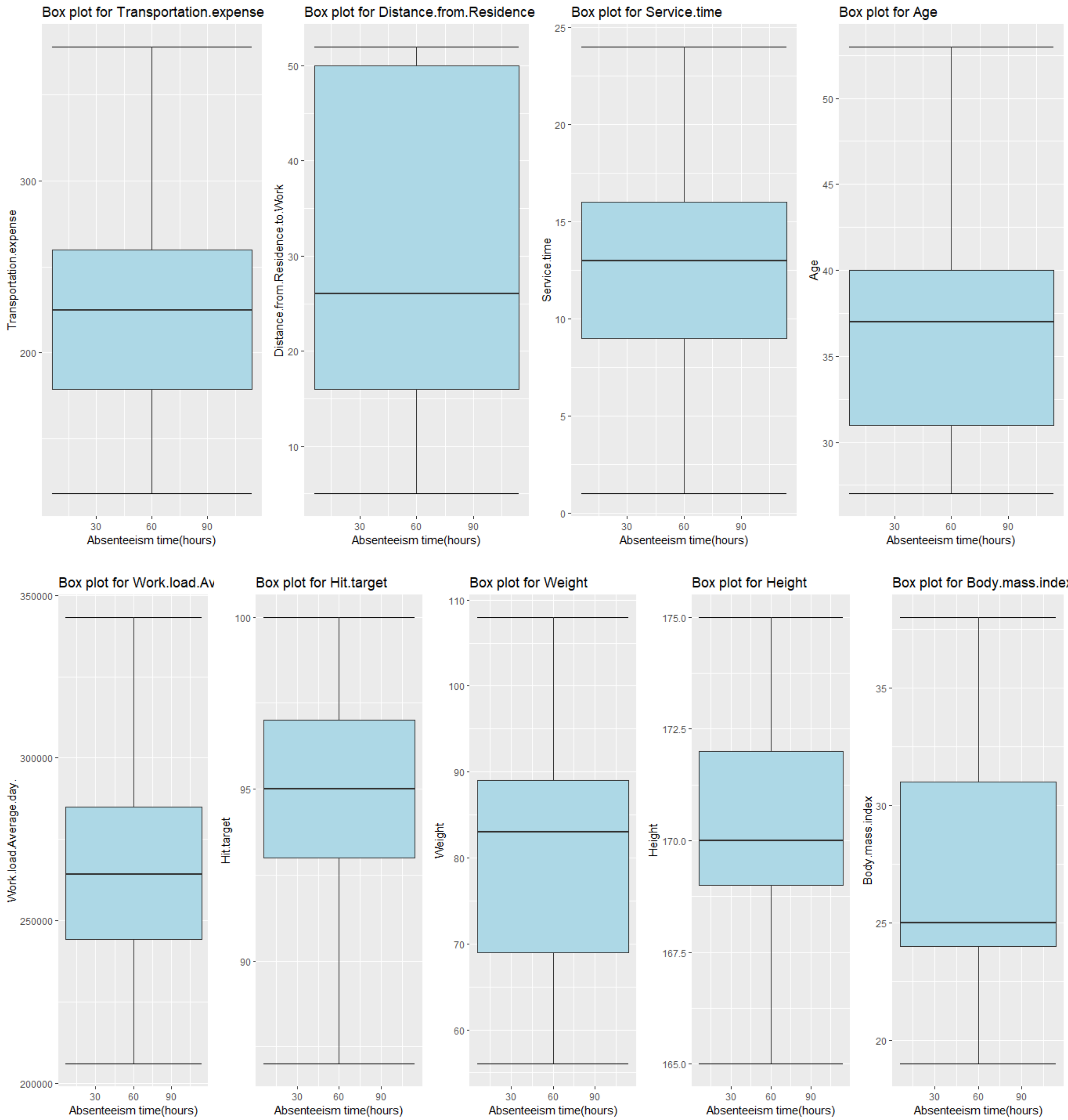
Absenteeism time Box plots for Independent Variables (before Outlier removal):





Outliers make sense only in numeric or continuous data for this dataset. The “Absenteeism time” variable consists of the labels to be used to train and test the predictive models and hence it should be left untouched by further manipulation by outlier analysis.

Box plots of the variables after Outlier removal:



After performing outlier Analysis and removing outliers using Boxplot Method, Predictor vs Absenteeism boxplots is plotted again to compare the differences.

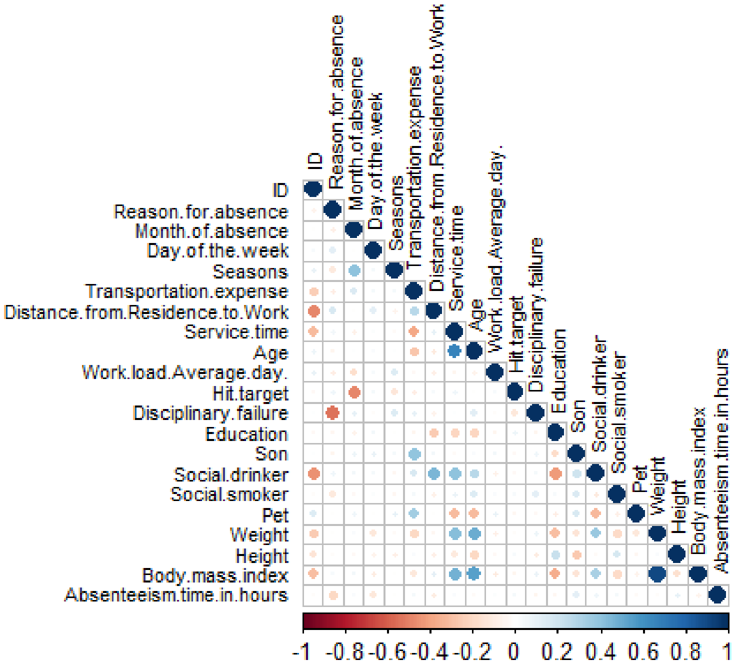
2.1.6. Feature Selection:

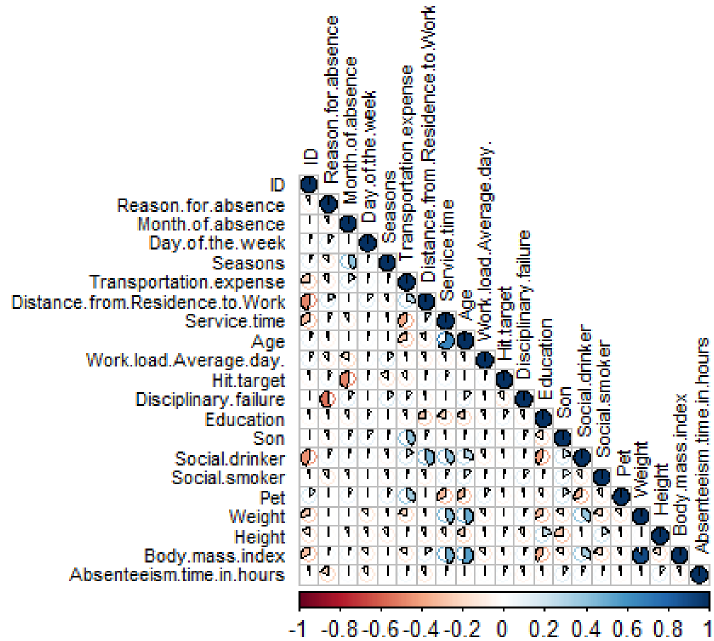
It is needed that we assess the importance of each predictor variable in our analysis, as there is a possibility that many variables in our analysis are not important at all to predict the 'Absenteeism time' values.

A. Using Correlation plots:


```
corrgram(data, order = F, upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot", font.labels = 1)
mat = cor(data)
corrplot(as.matrix(mat),method= 'pie',type = "lower", tl.col = "black", tl.cex = 0.7)
```

- If |r|>0.8 for two variables, those variables are considered redundant variables and one of them can be removed from the dataset.
- Output: "Weight" & "Body.Mass.Index" variables are highly positively correlated as expected after performing the pre-processing of the data.





B. Using Chi-square test of Independence (relationship between categorical variables):

```
n = c(1,2,3,4,5,12,13,15,16)          #indices of the categorical variables
for(i in n){
  print(names(data[i]))
  print(chisq.test(table(data$Absenteeism.time.in.hours,data[,i])))
 }
```

- If p-value<0.05 (Reject Null Hypothesis) => Target variable depends on the independent variable.
- If p-value>0.05 (Do Not Reject Null Hypothesis) =>Target variable & independent variable are independent of each other.
- Output:

```
"ID": X-squared = 1577.1, df = 1330, p-value = 2.868e-06
"Reason.for.absence": X-squared = 6085.7, df = 2698, p-value < 2.2e-16
"Month.of.absence": X-squared = 576.12, df = 570, p-value = 0.4206
"Day.of.the.week": X-squared = 185.04, df = 152, p-value = 0.03512
"Seasons": X-squared = 178.62, df = 114, p-value = 0.000105
"Disciplinary.failure": X-squared = 615.77, df = 38, p-value < 2.2e-16
"Education": X-squared = 55.066, df = 114, p-value = 1
"Social.drinker": X-squared = 57.514, df = 38, p-value = 0.022
"Social.smoker": X-squared = 45.661, df = 38, p-value = 0.1837
```

- Target Variable "Absenteeism.time.in.hours" depends on all the categorical variables, except “month of absence”, "Education" & “Social.smoker” attributes.

C. Using Random Forest Algorithm:

```
data.rf=randomForest(data$Absenteeism.time.in.hours~.,data = data, ntree=1000, keep.forest= F, importance= T)
importance(data.rf,type = 1)
```

- Output:

```
                                 %IncMSE
ID                            10.6189921
Reason.for.absence            13.9939070
Month.of.absence               8.9653443
Day.of.the.week                1.0176514
Seasons                        9.0015197
Transportation.expense         4.4330720
Distance.from.Residence.to.Work  5.8995292
Service.time                   9.8489478
Age                           11.7781713
Work.load.Average.day.         2.2867752
Hit.target                     0.3832429
Disciplinary.failure          -0.8874999
Education                      3.0991756
Son                            8.3011303
Social.drinker                 5.3789409
Social.smoker                 -1.2207280
Pet                            2.6435399
Weight                         7.2855691
Height                        14.4292822
Body.mass.index                6.6481634
```

- "Hit Target", “Disciplinary.failure”, "Social Smoker" & "day of the week" have the least variable importance with 0-1% (approx.).

D. Dimensional Reduction:

We should remove those features that do not contribute to predicting the target variable as it will only lead to increase in the complexity of the model and incorrect prediction.

- From Correlation Plot --> We can drop "Weight" or "Body.Mass.Index" column. From RF variable importance output, we see that “weight” has higher importance, so we can drop “Body.Mass.Index” variable.
- From Chi square test & Random Forest --> We drop "Education" & “Social.smoker” variables.

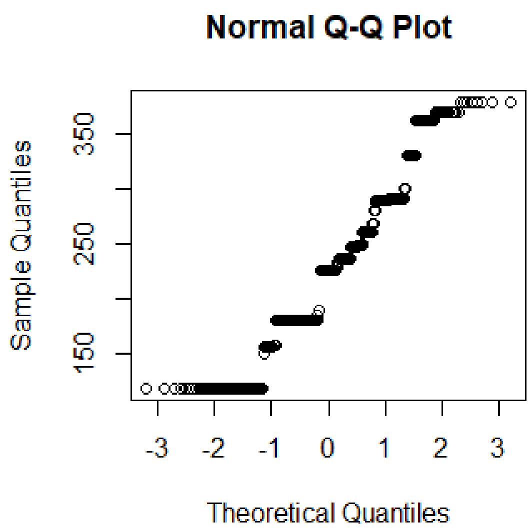
2.1.7. Feature Scaling:

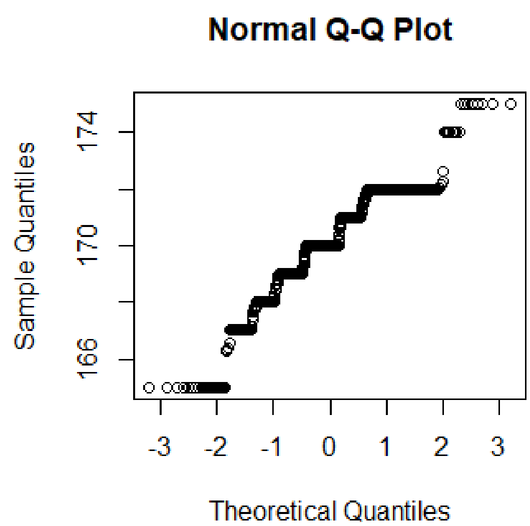
The dataset contains features that are highly varying in magnitudes, units and range. Feature Scaling (Normalization/Standardization) is a step of Data Pre-Processing, which is applied to independent variables or features of data. It helps to normalize the data within a particular range and sometimes helps in speeding up the calculations in distance-based algorithms. Standardization works well if the data is normally distributed, otherwise we scale data by normalization method.

A. Normality Check:

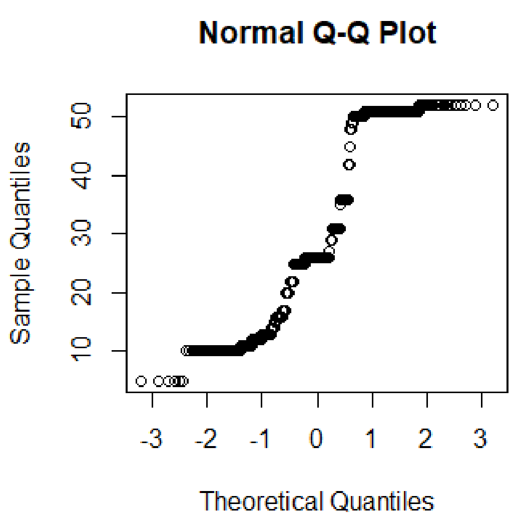
- Q-Q Plot for continuous variables:

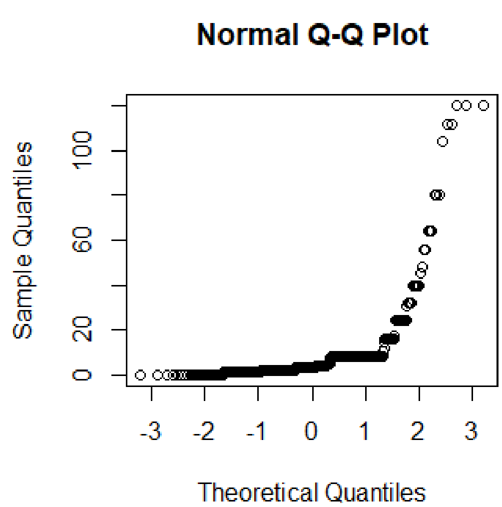
```
qqnorm(data$Transportation.expense)
qqnorm(data$Height)
```

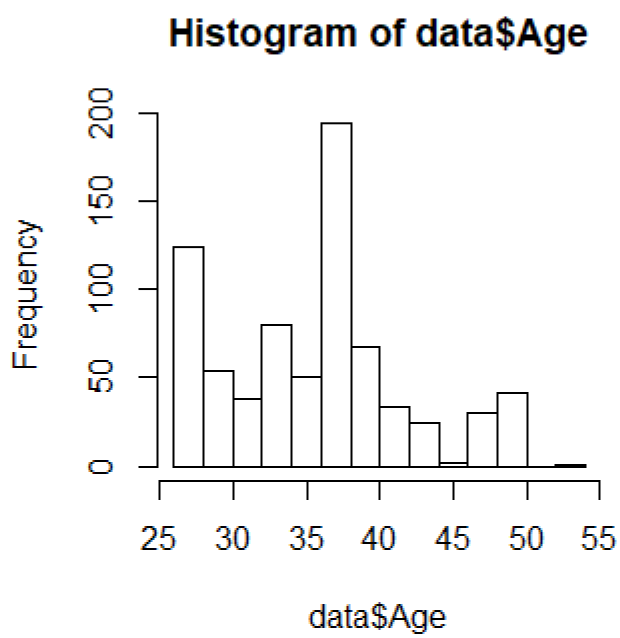


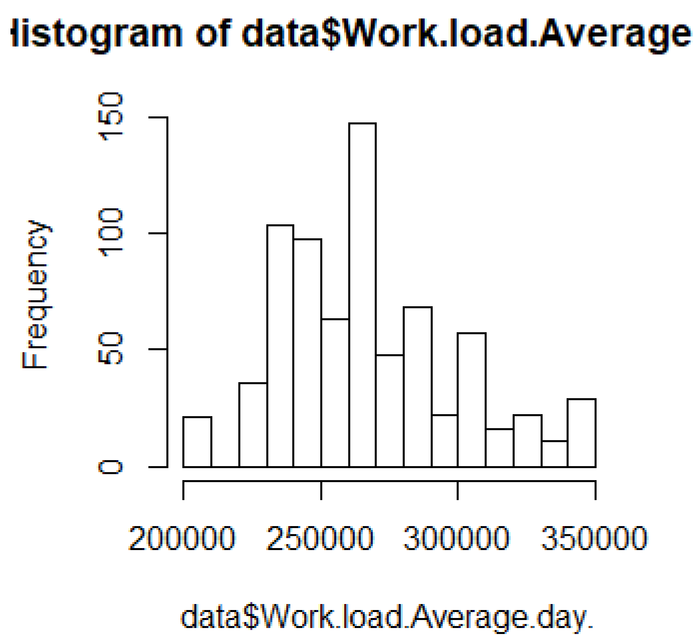



```
qqnorm(data$Distance.from.residence.to.work)
qqnorm(data$Absenteeism.time.in.hours)
```









None of the continuous variables is normally distributed. So, we scale the data using Normalization method.

Normalization formula:

$$z = \frac{x \ - \min(x)}{\max(x) - \min(x)}$$

```
#Only for continuous variables
for(i in OutCol){
  print(i)
  data[,i] = (data[,i] - min(data[,i]))/(max(data[,i]) - min(data[,i]))
}
```

2.1.8. Data Sampling:

The whole dataset is divided into train and test split sets so that there is data from which the model can learn and there is a part of the data set using which we can do unbiased evaluation of the trained model.

Random sampling without replacement is used to split 80% of the data into training set and remaining 20% into test set.

```
sample.index = sample(nrow(data), 0.8*nrow(data), replace = F)
train = data[sample.index,]
test = data[-sample.index,]
```

2.2. Modeling

2.2.1. Model Selection:

The dataset of XYZ Company indicates that this is a supervised learning problem as there is the task of inferring a function or values from the labeled training data. Secondly, the dependent variable “Absenteeism time in hours” is of real valued or continuous type and therefore our prediction is of a quantity & it is a regression problem. Since we have many input variables, we shall perform a **multivariate regression analysis** on the given dataset.

2.2.2. Decision Tree Algorithm

```
dt=rpart(Absenteeism.time.in.hours~.,data = train,method= "anova")
> summary(dt)
Call:
rpart(formula = Absenteeism.time.in.hours ~ ., data = train,
    method = "anova")
  n= 592

          CP nsplit rel error    xerror      xstd
1 0.12020120      0 1.0000000 1.0064746 0.2705362
2 0.04928046      1 0.8797988 0.8915125 0.2404877
3 0.02205094      3 0.7812379 0.9141441 0.2155769
4 0.01065252      5 0.7371360 0.9371219 0.2190678
5 0.01054222      7 0.7158310 0.9627655 0.2255687
6 0.01000000      8 0.7052887 0.9626555 0.2255695

Variable importance
             Reason.for.absence                             Age                             Son         Transportation.expense
                             33                              16                               9                              8
Distance.from.Residence.to.Work                          Height                          Weight                   Service.time
                              8                               7                               5                              4
           Work.load.Average.day                            Pet                   Social.drinker                     Hit.target
                              4                               2                               2                              1
                             ID
                              1

Node number 1: 592 observations,    complexity param=0.1202012
  mean=6.991777, MSE=173.0504
  left son=2 (378 obs) right son=3 (214 obs)
  Primary splits:
      Reason.for.absence              < 19.10219   to the right, improve=0.12020120, (0 missing)
      Son                             < 0.375      to the left,  improve=0.03804571, (0 missing)
      Height                          < 0.6174526  to the left,  improve=0.02067571, (0 missing)
      Distance.from.Residence.to.Work < 0.287234   to the right, improve=0.01763882, (0 missing)
      Transportation.expense          < 0.5980769  to the left,  improve=0.01338666, (0 missing)
```

```
  Surrogate splits:
      Height                       < 0.7086694  to the left,  agree=0.659, adj=0.056, (0 split)
      Work.load.Average.day          < 0.9519718  to the left,  agree=0.655, adj=0.047, (0 split)
      Transportation.expense         < 0.9826923  to the left,  agree=0.645, adj=0.019, (0 split)
      Distance.from.Residence.to.Work < 0.05319149 to the right, agree=0.644, adj=0.014, (0 split)
      Age                            < 0.8269231  to the left,  agree=0.644, adj=0.014, (0 split)

Node number 2: 378 observations
  mean=3.560138, MSE=9.371554

Node number 3: 214 observations,    complexity param=0.04928046
  mean=13.05327, MSE=404.6228
  left son=6 (137 obs) right son=7 (77 obs)
  Primary splits:
      Son                            < 0.375      to the left,  improve=0.04478752, (0 missing)
      ID                             < 15.5       to the right, improve=0.02774764, (0 missing)
      Service.time                   < 0.4565217  to the left,  improve=0.02342015, (0 missing)
      Distance.from.Residence.to.Work < 0.2021277  to the right, improve=0.02322409, (0 missing)
      Social.drinker                 < 0.5        to the left,  improve=0.01759270, (0 missing)
  Surrogate splits:
      Transportation.expense < 0.4961538  to the left,  agree=0.748, adj=0.299, (0 split)
      Height                 < 0.310058   to the right, agree=0.720, adj=0.221, (0 split)
      Service.time           < 0.4130435  to the left,  agree=0.706, adj=0.182, (0 split)
      Weight                 < 0.6442308  to the left,  agree=0.696, adj=0.156, (0 split)
      Reason.for.absence     < 14.18197   to the left,  agree=0.668, adj=0.078, (0 split)

Node number 6: 137 observations,    complexity param=0.01065252
  mean=9.861817, MSE=196.9078
  left son=12 (101 obs) right son=13 (36 obs)
  Primary splits:
      Age             < 0.4615385  to the left,  improve=0.02954384, (0 missing)
      Height          < 0.5875101  to the left,  improve=0.02574266, (0 missing)
      Day.of.the.week < 4.5        to the right, improve=0.02398209, (0 missing)
      Social.drinker  < 0.5        to the left,  improve=0.01599686, (0 missing)
      Pet             < 0.5853116  to the right, improve=0.01461495, (0 missing)
  Surrogate splits:
      Weight       < 0.7211538  to the left,  agree=0.847, adj=0.417, (0 split)
      ID           < 34.5       to the left,  agree=0.825, adj=0.333, (0 split)
      Height       < 0.3980134  to the right, agree=0.788, adj=0.194, (0 split)
      Service.time < 0.4347826  to the left,  agree=0.774, adj=0.139, (0 split)

Node number 7: 77 observations,    complexity param=0.04928046
  mean=18.73158, MSE=723.8284
  left son=14 (69 obs) right son=15 (8 obs)
  Primary splits:
```

```
    Age                          < 0.1757286  to the right, improve=0.11158270, (0 missing)
    Day.of.the.week              < 3.5        to the right, improve=0.08138058, (0 missing)
    Distance.from.Residence.to.Work < 0.2021277  to the right, improve=0.08000780, (0 missing)
    ID                           < 15         to the right, improve=0.05764600, (0 missing)
    Work.load.Average.day        < 0.2005483  to the right, improve=0.03857452, (0 missing)

Node number 12: 101 observations
  mean=8.42184, MSE=102.3121

Node number 13: 36 observations,    complexity param=0.01065252
  mean=13.90175, MSE=440.163
  left son=26 (23 obs) right son=27 (13 obs)
  Primary splits:
    Height                < 0.5697937  to the left,  improve=0.08744407, (0 missing)
    Day.of.the.week       < 4.5        to the right, improve=0.07850019, (0 missing)
    Reason.for.absence    < 13.11839   to the right, improve=0.06744882, (0 missing)
    Social.drinker        < 0.5        to the left,  improve=0.04207122, (0 missing)
    Work.load.Average.day < 0.244397   to the right, improve=0.03724410, (0 missing)
  Surrogate splits:
    Distance.from.Residence.to.Work < 0.4680851  to the left,  agree=0.917, adj=0.769, (0 split)
    Service.time                    < 0.6086957  to the right, agree=0.917, adj=0.769, (0 split)
    Pet                             < 0.1234224  to the left,  agree=0.917, adj=0.769, (0 split)
    Transportation.expense          < 0.575      to the left,  agree=0.833, adj=0.538, (0 split)
    Social.drinker                  < 0.5        to the right, agree=0.833, adj=0.538, (0 split)

Node number 14: 69 observations,    complexity param=0.02205094
  mean=15.67147, MSE=459.3171
  left son=28 (54 obs) right son=29 (15 obs)
  Primary splits:
    Distance.from.Residence.to.Work < 0.1702128  to the right, improve=0.07076549, (0 missing)
    Social.drinker                  < 0.5        to the left,  improve=0.06729260, (0 missing)
    Hit.target                      < 0.7307692  to the left,  improve=0.06373516, (0 missing)
    Seasons                         < 2.5        to the left,  improve=0.05947428, (0 missing)
    Reason.for.absence              < 18.82234   to the left,  improve=0.04594533, (0 missing)
  Surrogate splits:
    Transportation.expense < 0.1884615  to the right, agree=0.928, adj=0.667, (0 split)
    Weight                 < 0.7403846  to the left,  agree=0.928, adj=0.667, (0 split)
    ID                     < 7.5        to the right, agree=0.797, adj=0.067, (0 split)

Node number 15: 8 observations
  mean=45.125, MSE=2227.859

Node number 26: 23 observations
  mean=9.237525, MSE=56.28467
```

```
Node number 27: 13 observations
  mean=22.15385, MSE=1012.746

Node number 28: 54 observations,    complexity param=0.02205094
  mean=12.66667, MSE=313.4815
  left son=56 (39 obs) right son=57 (15 obs)
  Primary splits:
      Reason.for.absence < 18.82234   to the left,  improve=0.13441000, (0 missing)
      Hit.target         < 0.8846154  to the left,  improve=0.07125832, (0 missing)
      Day.of.the.week    < 5.5        to the left,  improve=0.06361566, (0 missing)
      Month.of.absence   < 5.5        to the right, improve=0.06091959, (0 missing)
      Social.drinker     < 0.5        to the left,  improve=0.05751834, (0 missing)
  Surrogate splits:
      Hit.target < 0.8076923  to the left,  agree=0.778, adj=0.2, (0 split)

Node number 29: 15 observations
  mean=26.48875, MSE=834.8078

Node number 56: 39 observations,    complexity param=0.01054222
  mean=8.641026, MSE=70.94806
  left son=112 (31 obs) right son=113 (8 obs)
  Primary splits:
      Work.load.Average.day  < 0.8204879  to the left,  improve=0.39032040, (0 missing)
      Reason.for.absence     < 13.5       to the right, improve=0.11775290, (0 missing)
      ID                     < 21.5       to the right, improve=0.09704926, (0 missing)
      Month.of.absence       < 3.5        to the right, improve=0.08930142, (0 missing)
      Transportation.expense < 0.6788462  to the right, improve=0.07056163, (0 missing)

Node number 57: 15 observations
  mean=23.13333, MSE=792.3822

Node number 112: 31 observations
  mean=5.967742, MSE=9.257024

Node number 113: 8 observations
  mean=19, MSE=175
```

Now, we predict for new test cases:

```
predict.dt=predict(dt,test[,-18])
```

2.2.3. Random Forest Algorithm

```
rf = randomForest(Absenteeism.time.in.hours~., train, importance = TRUE, ntree = 500)
> summary(rf)
                Length Class  Mode
```

```
call            5    -none- call
type            1    -none- character
predicted     592    -none- numeric
mse           500    -none- numeric
rsq           500    -none- numeric
oob.times     592    -none- numeric
importance     34    -none- numeric
importanceSD   17    -none- numeric
localImportance  0   -none- NULL
proximity       0    -none- NULL
ntree           1    -none- numeric
mtry            1    -none- numeric
forest         11    -none- list
coefs           0    -none- NULL
y             592    -none- numeric
test            0    -none- NULL
inbag           0    -none- NULL
terms           3    terms  call
```

We predict for test set:

```
predict.rf <- data.frame(predict(rf, subset(test, select = -c(Absenteeism.time.in.hours))))
```

2.2.4. Multiple Linear Regression

Multicollinearity is when independent variables in a regression model are correlated. It tries to inflate or resist the variance of different strong regressors in the data. Therefore, we need to do a collinearity check before performing linear regression.

```
> vif.data= vif(data[,-18])
> vifcor(data[,-18], th = 0.8)
No variable from the 17 input variables has collinearity problem.

The linear correlation coefficients ranges between:
min correlation ( Month.of.absence ~ ID ):  -4.542897e-05
max correlation ( Age ~ Service.time ):  0.661807

---------- VIFs of the remained variables --------
                    Variables      VIF
1                          ID 2.509781
2           Reason.for.absence 1.102944
3             Month.of.absence 1.720037
4              Day.of.the.week 1.078431
5                     Seasons 1.335310
6       Transportation.expense 2.458060
7  Distance.from.Residence.to.Work 1.841511
```

```
8                  Service.time 3.445202
9                           Age 3.134506
10          Work.load.Average.day 1.151894
11                   Hit.target 1.355962
12          Disciplinary.failure 1.090212
13                          Son 1.493890
14                Social.drinker 2.387001
15                          Pet 1.689840
16                       Weight 2.032082
17                       Height 1.238693
```

Now that multicollinearity is not an issue, we can build our regression model.

```
lr = lm(Absenteeism.time.in.hours~., data = train)
> #summary of the model
> summary(lr)

Call:
lm(formula = Absenteeism.time.in.hours ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-21.417  -4.351  -1.436   1.195 106.290

Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                     14.82960    5.58499   2.655 0.008145 **
ID                              -0.05373    0.07170  -0.749 0.453955
Reason.for.absence              -0.49346    0.07486  -6.592 9.86e-11 ***
Month.of.absence                 0.01028    0.19447   0.053 0.957861
Day.of.the.week                 -0.79728    0.36362  -2.193 0.028734 *
Seasons                          0.27226    0.52728   0.516 0.605818
Transportation.expense           1.68389    3.10483   0.542 0.587791
Distance.from.Residence.to.Work -7.19587    2.16700  -3.321 0.000955 ***
Service.time                     8.01568    5.05313   1.586 0.113226
Age                             -4.91859    3.76004  -1.308 0.191356
Work.load.Average.day           -1.47516    2.30282  -0.641 0.522045
Hit.target                       3.67282    2.48753   1.476 0.140360
Disciplinary.failure            -1.92652    2.39856  -0.803 0.422194
Son                              6.74366    2.23318   3.020 0.002642 **
Social.drinker                   3.67838    1.58240   2.325 0.020444 *
Pet                             -1.02093    1.82474  -0.559 0.576041
Weight                          -3.75747    2.86896  -1.310 0.190822
Height                           5.97834    3.03563   1.969 0.049389 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 12.23 on 574 degrees of freedom
Multiple R-squared:  0.1621,	Adjusted R-squared:  0.1373
F-statistic: 6.532 on 17 and 574 DF, p-value: 2.675e-14
```

Then we predict for the test set:

```
predict.lr=predict(lr, test[,-18])
```

2.2.5. KNN Implementation

KNN is distance based non-parametric algorithm and it never stores patterns from the training data, but classifies for new test cases based on a similarity measure.

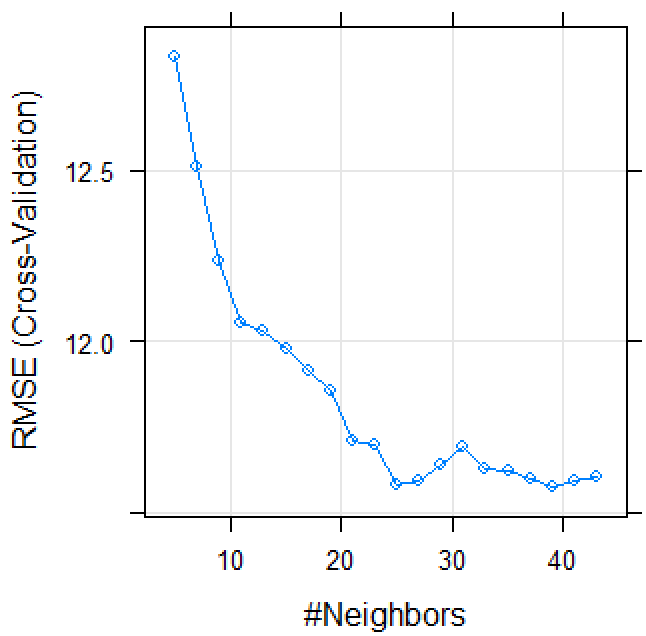
First, we need to check for the best no. of neighbors (k):

```
#K=5:
> predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k= 5)
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape 
  6.712344 199.341057  14.118819        Inf 
> #K=7:
> predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k= 7)
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape 
  6.418315 196.195775  14.006990        Inf 
> #K=9:
> predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k= 9)
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape 
  6.167826 187.593597  13.696481        Inf 
> #K=13:
> predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k= 13)
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape 
  6.039239 192.793157  13.884998        Inf 
> #K=15:
> predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k= 15)
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape 
  6.081862 195.226633  13.972352        Inf 
> #K=17:
> predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k= 17)
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape 
  5.944805 194.947266  13.962352        Inf 
```

```
> #K=19:
> predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k= 19)
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae       mse      rmse      mape
 5.853058 193.726139  13.918554       Inf
> #K=21:
> predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k= 21)
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae       mse      rmse      mape
 5.855254 194.307576  13.939425       Inf
```

Another way to check best k value:

```
model <- train(Absenteeism.time.in.hours~., data = train, method = "knn",
+              trControl = trainControl("cv", number = 10),
+              tuneLength = 20)
> model$bestTune
   k
18 39
> plot(model)
```



After checking both methods, it is best to choose k=19 as it gives us the least prediction error.

III. Conclusion

3.1 Model Evaluation:

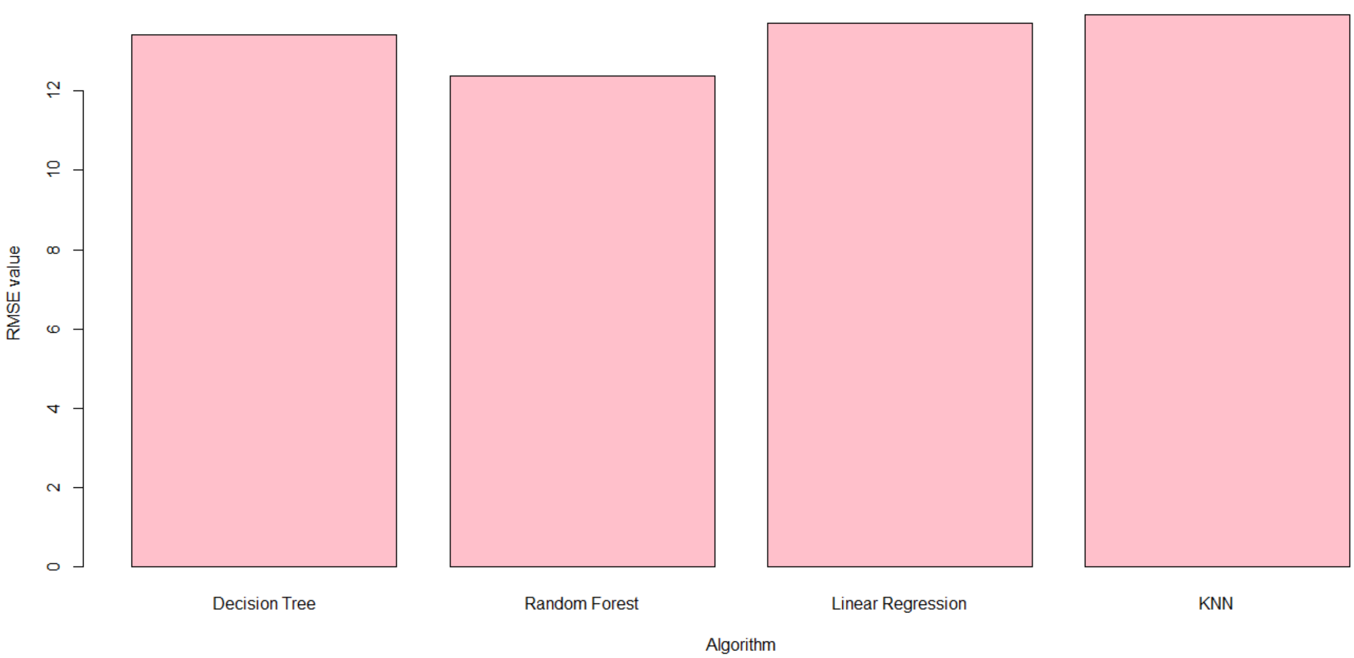
Now that we have a few models for predicting the target variable, we need to decide which one to choose. Several criterias exist for evaluating and comparing models; here we can compare the models by using assessing the 'Predictive Performance' of the models. Predictive performance can be measured by comparing Predictions of the models with real values of the target variables, and calculating some average error measure like MAE, MSE, RMSE or MAPE.

```
> #Error metric for Decision Tree:
> regr.eval(test[,18], predict.dt, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape
  5.753875 180.407076  13.431570        Inf
> #Output:
> #mae       mse       rmse       mape
> #5.753875 180.407076  13.431570   Inf
> postResample(predict.dt,test[,18])
     RMSE   Rsquared        MAE
13.4315701  0.1213995  5.7538750


> #Error metric for Random Forest:
> regr.eval(test[,18], predict.rf, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape
  5.295667 153.418298  12.386214        Inf
> #Output:
> #mae       mse        rmse       mape
> #5.295667 153.418298  12.386214     Inf
> postResample(predict.rf,test[,18])
     RMSE   Rsquared        MAE
12.3862140  0.2561696  5.2956666


> #Error metric for Multiple Linear Regression:
> regr.eval(test[,18], predict.lr, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape
  6.527533 188.009158  13.711643        Inf
> #Output:
> #mae       mse       rmse       mape
> #6.527533 188.009158  13.711643   Inf
> postResample(predict.lr,test[,18])
      RMSE    Rsquared         MAE
13.71164317  0.08126101  6.52753308


> #Error metric for KNN Implementation:
> regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
      mae        mse       rmse       mape
```

```
 5.853058 193.726139  13.918554       Inf
> #Output:
> #mae      mse       rmse      mape
> #5.853058 193.726139  13.918554   Inf
> postResample(predict.knn$pred,test[,18])
     RMSE    Rsquared        MAE
13.91855378  0.05935665  5.85305761
```

MAPE (mean absolute percentage error) values come infinity because the denominator of the MAPE formula has actual values of the target variable which is '0 hours' too and it is obvious that MAPE won't work for this dataset. Therefore, we compare the predictive models using MAE, MSE and RMSE.

3.2 Final Model Selection:



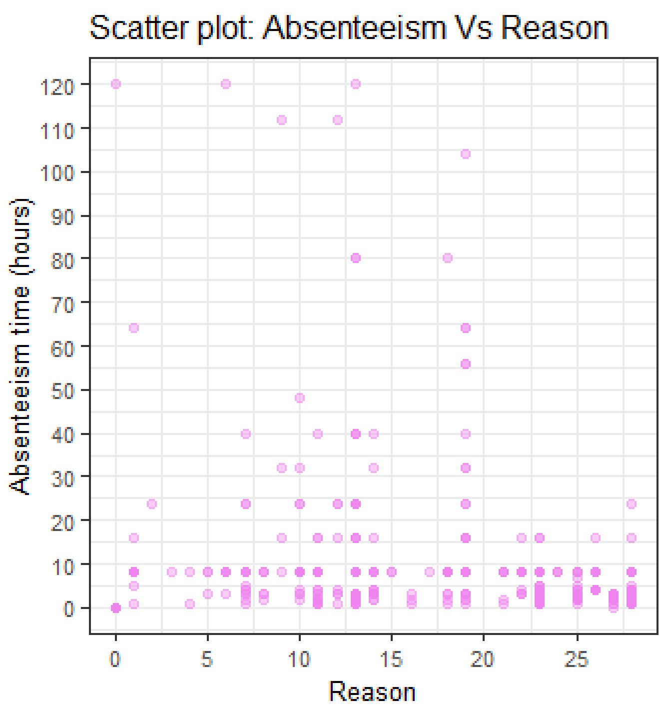
As we can observe that "Random Forest" algorithm produces the least error or RMSE (Root mean Square Error), we can freeze this algorithm as the model for analysis of new data or test cases of Absenteeism time of Company XYZ.

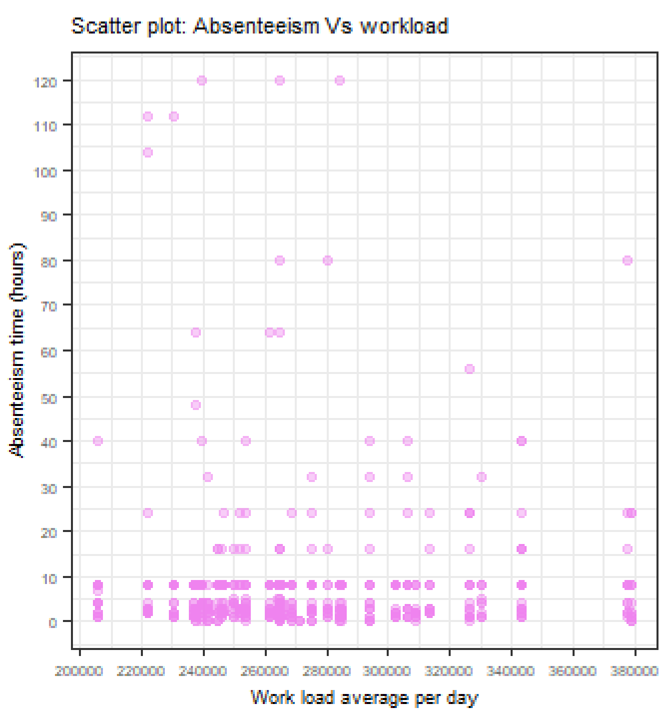
3.3 Solutions:

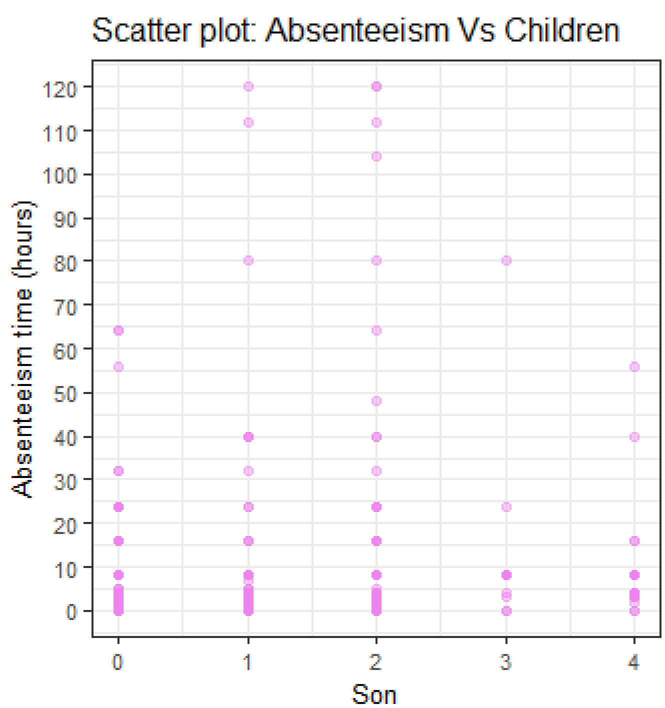
- A better model can be developed if there is more data available. More variables that are relevant can be added as the input variables and the dependent variable are less correlated and we can also see that R squared values are very low, i.e. the variance of target variable is not explained by the independent variables much.
- The employer can look into optimizing their employees' targets as more absentees have higher hit targets.

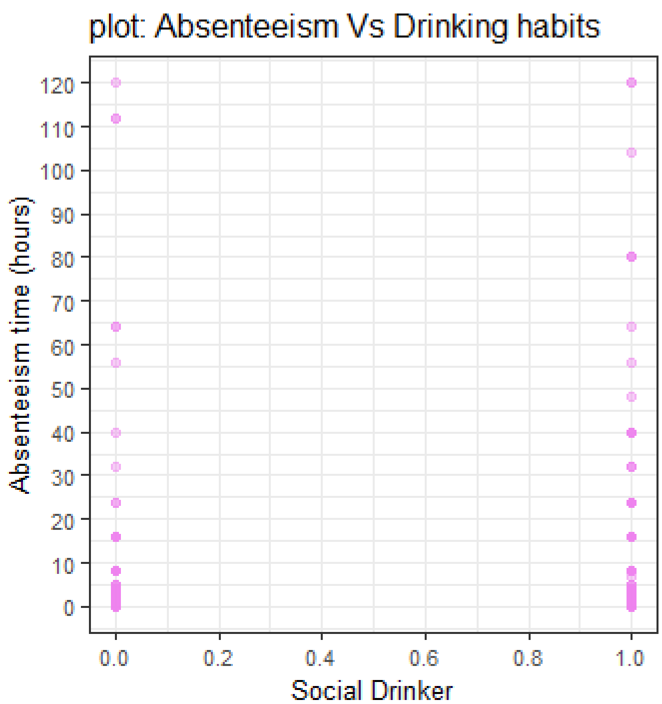
- Employer can consider hiring highly educated employees as according to the data, they seem to take less leaves.
- The company can arrange regular health checkups near or in their company premises, the losses due absenteeism for doctor consultation can be reduced.
- Some absentees were spending more money for less distance from work; therefore, the company should provide a common mode of transport for all employees, if reasonable.

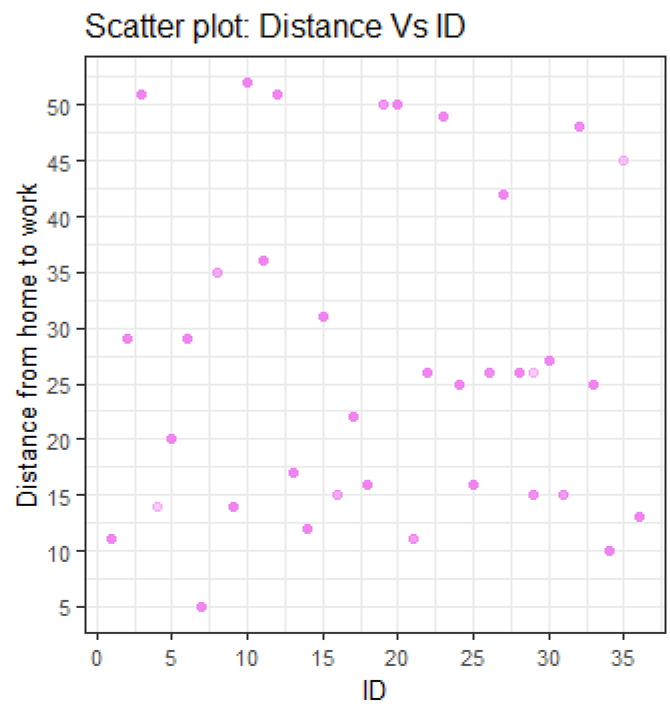
Appendix A: Extra plots

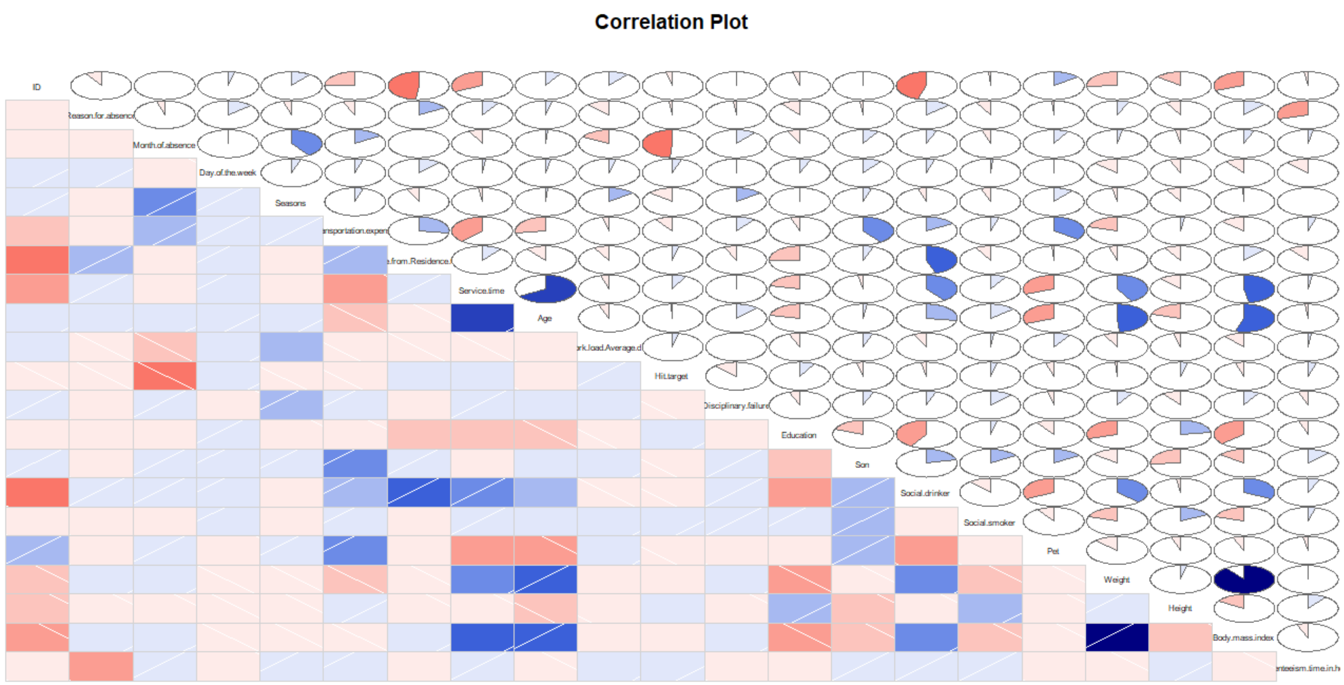












Appendix B: R code

```
#To clear the R environment of any predefined objects
rm(list=ls())
#To set working directory
setwd("F:/DS/edWisor/Project 1")
getwd()
```

```
#To load required libraries
library(xlsx)
library(dplyr)     # used for data manipulation and joining
library(ggplot2)   # used for ploting
library(caret)     # used for modeling
library(corrgram)
library(corrplot)  # used for making correlation plot
library(DMwR)
library(randomForest)
library(class)
library(FNN)
library(scales)


#To load the data
data = read.xlsx("Absenteeism_at_work_Project.xls",sheetIndex = 1, header = T)
data[data == " " | data == "" | data == "NA"] = NA

############Data Exploration#############
str(data)
dim(data)
#Data has 9 categorical variables & 12 numeric variables
#Target variable is continuous in nature

###Univariate Analysis
#Since all data are in numeric type, we don't need to convert it for data consolidation
#Histogram for Target variable (continuous variable)
ggplot(data, aes_string(x = data$Absenteeism.time.in.hours)) +
  geom_histogram(fill="cornsilk", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Absenteeism time (hours)") + ylab("Frequency") + ggtitle("Target 
Variable Histogram") +
  theme(text=element_text(size=10))

#Histogram for Independent Continuous Variables
ggplot(data, aes_string(x = data$Transportation.expense)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
```

```
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Transport cost") + ylab("Frequency") +
ggtitle("IndependentVariable:Transportation Cost") +
  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Distance.from.Residence.to.Work)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Distance from Residence to work (km)") + ylab("Frequency") +
ggtitle("Independent Variable:Distance") +
  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Service.time)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Service time (hours)") + ylab("Frequency") + ggtitle("Independent
Variable: Service time") +
  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Age)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Age (years)") + ylab("Frequency") + ggtitle("Independent Variable:
Employee Age") +
  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Work.load.Average.day.)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Average Work load per day") + ylab("Frequency") + ggtitle("Independent
Variable: Work load") +
  theme(text=element_text(size=7))
ggplot(data, aes_string(x = data$Hit.target)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Hit target") + ylab("Frequency") + ggtitle("Independent Variable: Hit
target") +
  theme(text=element_text(size=10))
```

```
ggplot(data, aes_string(x = data$Son)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Number of children") + ylab("Frequency") + ggtitle("Independent
Variable:Employee's children") +
  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Weight)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Weight (kg)") + ylab("Frequency") + ggtitle("Independent
Variable:Employee's weight") +
  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Height)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Height (cm)") + ylab("Frequency") + ggtitle("Independent
Variable:Employee's height") +
  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Body.mass.index)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Body Mass Index") + ylab("Frequency") + ggtitle("Independent
Variable:Employee's BMI") +
  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Pet)) +
  geom_histogram(fill="blue", colour = "black") + geom_density() +
  scale_y_continuous(breaks=pretty_breaks(n=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10))+
  theme_bw() + xlab("Number of pets") + ylab("Frequency") + ggtitle("Independent
Variable:Employee's pets") +
  theme(text=element_text(size=10))

#Bar graph for Independent Categorical Variables
ggplot(data, aes_string(x = data$ID)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
```

```
  xlab("ID") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10)) +
  ggtitle("Independent Variable: Employee ID ") +  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Reason.for.absence)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
  xlab("Reason of Absence") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10))
+
  ggtitle("Independent Variable: Reason of Absence ") +  theme(text=element_text(size=9))
ggplot(data, aes_string(x = data$Month.of.absence)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
  xlab("Month of Absence") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10))
+
  ggtitle("Independent Variable: Month ") +  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Day.of.the.week)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
  xlab("Day of the week") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10)) +
  ggtitle("Independent Variable: Day ") +  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Seasons)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
  xlab("Seasons") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10)) +
  ggtitle("Independent Variable: Seasons ") +  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Disciplinary.failure)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
  xlab("Disciplinary Failure") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10))
+
  ggtitle("Independent Variable: Disciplinary Failure ") +  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Social.drinker)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
  xlab("Social Drinker") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10)) +
  ggtitle("Independent Variable: drinking habits ") +  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Social.smoker)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
  xlab("Social smoker") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10)) +
  ggtitle("Independent Variable: Smoking habits ") +  theme(text=element_text(size=10))
ggplot(data, aes_string(x = data$Education)) +
  geom_bar(stat="count",fill =  "DarkSlateBlue") + theme_bw() +
  xlab("Education") + ylab('Count') + scale_y_continuous(breaks=pretty_breaks(n=10)) +
  ggtitle("Independent Variable: Education ") +  theme(text=element_text(size=10))

###Bivariate Analysis
```

```
##Target Variable Vs Independent Categorical Variables
length(unique(data$ID))
#1.Absenteeism time Vs ID
ggplot(data) +
  geom_point(aes(data$ID, data$Absenteeism.time.in.hours),colour = "violet", alpha = 0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("ID") + ggtitle("Scatter plot:
Absenteeism Vs ID") +
  theme(text=element_text(size=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
  scale_y_continuous(breaks=pretty_breaks(n=10))

#2.Absenteeism time Vs Reason of Absence
ggplot(data) +
  geom_point(aes(data$Reason.for.absence, data$Absenteeism.time.in.hours),colour =
"violet", alpha = 0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("Reason") + ggtitle("Scatter plot:
Absenteeism Vs Reason") +
  theme(text=element_text(size=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
  scale_y_continuous(breaks=pretty_breaks(n=10))

#3.Absenteeism time Vs Month of Absence
ggplot(data) +
  geom_point(aes(data$Month.of.absence, data$Absenteeism.time.in.hours),colour = "violet",
alpha = 0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("Month of absence") + ggtitle("Scatter
plot: Absenteeism Vs Month") +
  theme(text=element_text(size=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
  scale_y_continuous(breaks=pretty_breaks(n=10))

#4.Absenteeism time Vs Month of Absence
ggplot(data) +
  geom_point(aes(data$Seasons , data$Absenteeism.time.in.hours),colour = "violet", alpha =
0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("Seasons") + ggtitle("Scatter plot:
Absenteeism Vs Seasons") +
  theme(text=element_text(size=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
```

```
  scale_y_continuous(breaks=pretty_breaks(n=10))

#5.Absenteeism time Vs Transportation Expense
ggplot(data) +
  geom_point(aes(data$Transportation.expense , data$Absenteeism.time.in.hours),colour =
"violet", alpha = 0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("Transportation Expense") +
ggtitle("Scatter plot: Absenteeism Vs Transportation") +
  theme(text=element_text(size=9)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
  scale_y_continuous(breaks=pretty_breaks(n=10))

#6.Absenteeism time Vs Distance from residence to work
ggplot(data) +
  geom_point(aes(data$Distance.from.Residence.to.Work ,
data$Absenteeism.time.in.hours),colour = "violet", alpha = 0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("Distance from residence to work
(km)") + ggtitle("Scatter plot: Absenteeism Vs distance") +
  theme(text=element_text(size=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
  scale_y_continuous(breaks=pretty_breaks(n=10))

#7.Absenteeism time Vs Service time
ggplot(data) +
  geom_point(aes(data$Service.time , data$Absenteeism.time.in.hours),colour = "violet", alpha
= 0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("Service time(hrs)") + ggtitle("Scatter
plot: Absenteeism Vs Service time") +
  theme(text=element_text(size=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
  scale_y_continuous(breaks=pretty_breaks(n=10))

#8.Absenteeism time Vs Age
ggplot(data) +
  geom_point(aes(data$Age , data$Absenteeism.time.in.hours),colour = "violet", alpha = 0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("age (years)") + ggtitle("Scatter plot:
Absenteeism Vs Age") +
  theme(text=element_text(size=10)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
```

```
  scale_y_continuous(breaks=pretty_breaks(n=10))

#9.Absenteeism time Vs workload
ggplot(data) +
  geom_point(aes(data$Work.load.Average.day. , data$Absenteeism.time.in.hours),colour =
"violet", alpha = 0.4) +
  theme_bw()+ ylab("Absenteeism time (hours)") + xlab("Work load average per day") +
ggtitle("Scatter plot: Absenteeism Vs workload") +
  theme(text=element_text(size=7)) +
  scale_x_continuous(breaks=pretty_breaks(n=10)) +
  scale_y_continuous(breaks=pretty_breaks(n=10))

#And so on. The graphs are plotted and recorded in the project report.

##########Fixing Data Anomalies##########
#Treating the 0's in 'Reason of absences' & 'Month of absence' as missing values
zero_index = which(data$Reason.for.absence == 0)
for(i in zero_index){
  data$Reason.for.absence[i]= NA
}
zero_index = which(data$Month.of.absence == 0)
for(i in zero_index){
  data$Month.of.absence[i]= NA
}

##################Missing Value Analysis##################
missing_val = data.frame(apply(data,2,function(x){sum(is.na(x))}))
missing_val$Columns = row.names(missing_val)
names(missing_val)[1] =  "Missing_percentage"
missing_val$NumberOfMissingValues = missing_val$Missing_percentage
missing_val$Missing_percentage = (missing_val$Missing_percentage/nrow(data)) * 100
missing_val = missing_val[order(-missing_val$Missing_percentage),]
row.names(missing_val) = NULL
missing_val = missing_val[,c(2,1,3)]
write.csv(missing_val, "Missing_percentage.csv", row.names = T)

ggplot(data = missing_val[1:3,], aes(x=reorder(Columns, -Missing_percentage),y =
Missing_percentage))+
  geom_bar(stat = "identity",fill = "grey")+xlab("Parameter")+
```

```
  ggtitle("Missing data percentage (Train)") + theme_bw() + theme(text=element_text(size=8))

##To test for the best method to find missing values for this dataset
#data[6,6]=179 (actual)
data[6,6]=NA
#By median method:
data$Transportation.expense[is.na(data$Transportation.expense)]=median(data$Transportati
on.expense, na.rm = T)
data[6,6]
#data[6,6]= 225 (median)
#reupload data again
rm(list = ls())
data = read.xlsx("Absenteeism_at_work_Project.xls",sheetIndex = 1, header = T)
data[data == " " | data == "" | data == "NA"] = NA
zero_index = which(data$Reason.for.absence == 0)
for(i in zero_index){
  data$Reason.for.absence[i]= NA
}
zero_index = which(data$Month.of.absence == 0)
for(i in zero_index){
  data$Month.of.absence[i]= NA
}
#By mean method:
data[6,6]=NA
data$Transportation.expense[is.na(data$Transportation.expense)]=mean(data$Transportatio
n.expense, na.rm = T)
data[6,6]
#data[6,6]= 221.0929 (mean)
#reupload data again
rm(list = ls())
data = read.xlsx("Absenteeism_at_work_Project.xls",sheetIndex = 1, header = T)
data[data == " " | data == "" | data == "NA"] = NA
zero_index = which(data$Reason.for.absence == 0)
for(i in zero_index){
  data$Reason.for.absence[i]= NA
}
zero_index = which(data$Month.of.absence == 0)
for(i in zero_index){
  data$Month.of.absence[i]= NA
```

```
}
#By KNN Imputation:
data[6,6]=NA
data=knnImputation(data, k=5)
data[6,6]
#data[6,6]= 179 (KNN), which is the closest to 179 and hence, we freeze this method.

#reload the data & replace the missing values
rm(list = ls())
data = read.xlsx("Absenteeism_at_work_Project.xls",sheetIndex = 1, header = T)
data[data == " " | data == "" | data == "NA"] = NA
zero_index = which(data$Reason.for.absence == 0)
for(i in zero_index){
  data$Reason.for.absence[i]= NA
}
zero_index = which(data$Month.of.absence == 0)
for(i in zero_index){
  data$Month.of.absence[i]= NA
}
data = knnImputation(data, k=5) #KNN
sum(is.na(data)) #To verify

write.csv(data, 'data_Missing.csv', row.names = F)

####################Outlier Analysis########################
####Box Plot distribution & outlier check####
#str(data)
#since all the variables are numeric data type, we don't need to change data type here

cnames = colnames(data)

for(i in 1:length(cnames)){
	assign(paste0("gn",i), ggplot(aes_string(y = (cnames[i]), x =
data$Absenteeism.time.in.hours), data = subset(data))+
	stat_boxplot(geom = "errorbar", width = 0.5) +
	geom_boxplot(outlier.colour="red", fill = "light blue",outlier.shape=18,outlier.size=3,
notch=FALSE) +
	theme(legend.position="bottom")+
	labs(y=cnames[i],x="Absenteeism time(hours)")+
```

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

#Plotting plots together
gridExtra::grid.arrange(gn2,gn3,gn4,gn5,ncol=4)
gridExtra::grid.arrange(gn6,gn7,gn8,gn9,ncol=4)
gridExtra::grid.arrange(gn10,gn11,gn18,gn19, gn20,ncol=5)

#Replace all outliers with NA and impute using KNN:

#for(i in cnames){
#   val = data[,i][data[,i] %in% boxplot.stats(data[,i])$out]
#   print(length(val))
#   data[,i][data[,i] %in% val] = NA
#}
#sum(is.na(data)) #To verify
#No. of NAs(outliers in whole dataset)= 501, which is high.


OutCol = colnames(data)
#We don't want some categorical columns to manipulated for outliers, like ID (Employee's ID is
unique and can't be an outlier).
#also, Target variable shouldn't be manipulated much.
OutCol = OutCol[c(-1,-2,-3,-4,-5,-12,-13,-15,-16,-21)]   #OutCol contains only continuous
variables now
for(i in OutCol){
  val = data[,i][data[,i] %in% boxplot.stats(data[,i])$out]
  print(length(val))
  data[,i][data[,i] %in% val] = NA
}
sum(is.na(data)) #To verify
#No. of NAs(outliers in numeric variables)= 231 (without categorical variables)
#highest no. of outliers found in "height" column, i.e. 119 NA in that column now
#119/740*100 = 16% > 10%, the part of the data to be manipulated.
#We fill in the NAs using KNN imputation.
data = knnImputation(data, k=5)
sum(is.na(data)) #to verify

write.csv(data, 'data_without Outliers.csv', row.names = F)
```

```
##############################Feature Selection##############################
#Correlation Plot
corrgram(data, order = F,
        upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot", font.labels = 1)
mat = cor(data)
corrplot(as.matrix(mat),method= 'pie',type = "lower", tl.col = "black", tl.cex = 0.7)
#If |r|>0.8, those two variables are redundant variables.
#Output: "Weight"&"Body.Mass.Index" are highly positively correlated.

#Chi-square Test of Independence
n = c(1,2,3,4,5,12,13,15,16)            #indices of the categorical variables
for(i in n){
  print(names(data[i]))
  print(chisq.test(table(data$Absenteeism.time.in.hours,data[,i])))
 }
#If p-value<0.05 (Reject Null Hypothesis) => Target variable depends on the independent
variable.
#If p-value>0.05 (Do Not Reject Null Hypothesis) =>Target variable & independent variable are
independent of each other.

#Using Random Forest Algorithm:
data.rf=randomForest(data$Absenteeism.time.in.hours~.,data = data, ntree=1000,
keep.forest= F, importance= T)
importance(data.rf,type = 1)

###################Dimensional Reduction######################
#From Correlation Plot --> We can drop "Weight" or "Body.Mass.Index" column.
#From Chi square test --> We drop "Education" & "Social.smoker"
data= subset(data, select= -c(Body.mass.index,Education,Social.smoker))

if("Body.mass.index" %in% OutCol) OutCol = OutCol[ - which(OutCol == "Body.mass.index")]
if("Education" %in% OutCol) OutCol = OutCol[ - which(OutCol == "Education")]
if("Social.smoker" %in% OutCol) OutCol = OutCol[ - which(OutCol == "Social.smoker")]

##################################Feature
Scaling####################################
#Only for continuous variables
#Normality check
```

```
qqnorm(data$Absenteeism.time.in.hours)
qqnorm(data$Transportation.expense)
qqnorm(data$Distance.from.Residence.to.Work)
qqnorm(data$Service.time)
qqnorm(data$Height)
qqnorm(data$Weight)
#None of the continuous variables have a normal distribution.
hist(data$Age)
hist(data$Work.load.Average.day.)
#We go for "Normalization", instead of standardization
data1 = data
for(i in OutCol){
  print(i)
  data[,i] = (data[,i] - min(data[,i]))/(max(data[,i]) - min(data[,i]))
}

##############################Sampling#################################
set.seed(1234)
sample.index = sample(nrow(data), 0.8*nrow(data), replace = F)   #80% data -->Train set, 20%-
-> Test set

###################################Model
Development###################################
rm(list= ls()[!(ls() %in% c('data','sample.index','data1'))])
train = data[sample.index,]
test = data[-sample.index,]
dim(train)
dim(test)

######1.Decision Tree######
library(rpart)
dt=rpart(Absenteeism.time.in.hours~.,data = train,method= "anova")
summary(dt)

#Predict for new test cases
predict.dt=predict(dt,test[,-18])

#Error metric:
regr.eval(test[,18], predict.dt, stats = c("mae","mse","rmse","mape"))
```

```
#Output:
#mae       mse       rmse       mape
#5.753875 180.407076  13.431570   Inf
postResample(predict.dt,test[,18])
#RMSE   Rsquared       MAE
#13.4315701  0.1213995  5.7538750

#MAPE is Inf as some actual values of target variable is 0 hours.
#MAPE is not usually used as a measure in case of time-series Analysis.
rms.dt = RMSE(predict.dt, test[,18], na.rm = F)


#######2.Random Forest Algorithm#######
rf = randomForest(Absenteeism.time.in.hours~., train, importance = TRUE, ntree = 500)
summary(rf)
#Predict for test case:
predict.rf <- data.frame(predict(rf, subset(test, select = -c(Absenteeism.time.in.hours))))
#Error metric:
regr.eval(test[,18], predict.rf, stats = c("mae","mse","rmse","mape"))
#Output:
#mae       mse        rmse       mape
#5.295667 153.418298  12.386214     Inf
postResample(predict.rf,test[,18])
#RMSE   Rsquared       MAE
#12.3862140  0.2561696  5.2956666

#MAPE is Inf as some actual values of target variable is 0 hours.
#Errors did decrease.
rms.rf = RMSE(predict.rf, test[,18], na.rm = F)


########3.Multiple Linear Regression########
#Check Multicollinearity
library(usdm)
vif.data= vif(data[,-18])
vifcor(data[,-18], th = 0.8)
#Output:
#No variable from the 17 input variables has collinearity problem.
```

```
#To run regression model
lr = lm(Absenteeism.time.in.hours~., data = train)
#summary of the model
summary(lr)

#Predict for test case:
predict.lr=predict(lr, test[,-18])
#Error metric:
regr.eval(test[,18], predict.lr, stats = c("mae","mse","rmse","mape"))
#Output:
#mae       mse       rmse       mape
#6.527533 188.009158  13.711643   Inf
postResample(predict.lr,test[,18])
#RMSE    Rsquared        MAE
#13.71164317  0.08126101  6.52753308

#Random Forest Algorithm got better result than Linear Regression.
rms.lr = RMSE(predict.lr, test[,18], na.rm = F)


##############4.KNN Implementation#############
#Predict for test data:
#K=5:
#predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
5)
#regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#Output:
#mae       mse       rmse      mape
#6.712344 199.341057  14.118819    Inf

#K=7:
#predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
7)
#regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#Output:
#mae      mse       rmse      mape
#6.418315 196.195775  14.006990   Inf

#K=9:
```

```
#predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
9)
#regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#Output:
#mae      mse       rmse      mape
#6.167826 187.593597  13.696481   Inf

#K=11:
#predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
11)
#regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#Output:
#mae      mse       rmse      mape
#6.136897 192.691006  13.881319   Inf

#K=13:
#predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
13)
#regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#Output:
#mae      mse      rmse       mape
#6.039239 192.793157  13.884998   Inf

#K=15:
#predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
15)
#regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#Output:
#mae      mse       rmse       mape
#6.081862 195.226633  13.972352   Inf

#K=17:
#predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
17)
#regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#Output:
#mae      mse       rmse      mape
#5.944805 194.947266  13.962352   Inf
```

```
#K=19:
predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
19)
print(predict.knn)
#Error metric:
regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#Output:
#mae      mse       rmse      mape
#5.853058 193.726139  13.918554   Inf
postResample(predict.knn$pred,test[,18])
#RMSE        Rsquared        MAE
#13.91855378  0.05935665  5.85305761

#K=21:
#predict.knn = knn.reg(train = train[,-18],test = test[,-18],train$Absenteeism.time.in.hours, k=
21)
#regr.eval(test[,18], predict.knn$pred, stats = c("mae","mse","rmse","mape"))
#mae      mse       rmse      mape
#5.855254 194.307576  13.939425   Inf

#To check for best k value:
model <- train(Absenteeism.time.in.hours~., data = train, method = "knn",
            trControl = trainControl("cv", number = 10),
            tuneLength = 20)
model$bestTune
plot(model)
rms.knn = RMSE(predict.knn$pred, test[,18], na.rm = F)

#A new dataframe to store results
algorithm <- c('Decision Tree','Random Forest','Linear Regression','KNN')
RMSE_val <- c(rms.dt,rms.rf,rms.lr,rms.knn)
results <- data.frame(algorithm, RMSE_val)
print(results)
barplot(results$RMSE_val, width = 1, names.arg = results$algorithm,
       ylab="RMSE value", xlab = "Algorithm",col='pink')
```

Appendix C: Python code

```
#Set working directory
import os
os.chdir("F:/DS/edWisor/Project 1")
os.getcwd()
Load libraries
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from random import randrange, uniform
from scipy.stats import chi2_contingency
from fancyimpute import KNN
from ggplot import *
#Load the data
data = pd.read_excel("Absenteeism_at_work_Project.xls")
data.shape
data.head(10)
Data Exploration
#Histogram for Target Variable
ggplot(data, aes(x = 'Absenteeism time in hours')) + geom_histogram(fill="DarkSlateBlue",
colour = "black") +\
   geom_density() +\
   theme_bw() + xlab("Absenteeism time (hours)") + ylab("Frequency") + ggtitle("Target
Variable Analysis") +\
   theme(text=element_text(size=20))
#fixing data anomalies
data['Reason for absence'] = data['Reason for absence'].replace(0.0,np.nan)
data['Month of absence'] = data['Month of absence'].replace(0.0,np.nan)
Missing Value Analysis
#create dataframe with missing percentage
missing_val=pd.DataFrame(data.isnull().sum())
missing_val=missing_val.reset_index()
missing_val=missing_val.rename(columns={'index':'variables',0: 'Missing %'})
missing_val['Missing %']=(missing_val['Missing %']/len(data))*100
missing_val=missing_val.sort_values('Missing %',ascending=False).reset_index(drop=True)
```

```
missing_val.to_csv('Missing_%.csv',index=False)
missing_val
#imputation method
data['Height'].loc[129]
#Actual value = 171
#create missing value
data['Height'].loc[129]=np.nan
#impute with mean
data['Height'] = data['Height'].fillna(data['Height'].mean())
data['Height'].loc[129]
#mean value = 172.154
#reload data
data = pd.read_excel("Absenteeism_at_work_Project.xls")
data['Reason for absence'] = data['Reason for absence'].replace(0.0,np.nan)
data['Month of absence'] = data['Month of absence'].replace(0.0,np.nan)
#impute with median
data['Height'].loc[129]=np.nan
data['Height'] = data['Height'].fillna(data['Height'].median())
data['Height'].loc[129]
#median value = 170.0
#reload data
data = pd.read_excel("Absenteeism_at_work_Project.xls")
data['Reason for absence'] = data['Reason for absence'].replace(0.0,np.nan)
data['Month of absence'] = data['Month of absence'].replace(0.0,np.nan)
#impute with KNN
#since we have all data in numeric, we can apply KNN.
data.dtypes
data['Height'].loc[129]=np.nan
data = pd.DataFrame(KNN(21).fit_transform(data),columns=data.columns)
#KNN value = 170.73 , closest to 171. so we freeze this method.
#to verify
pd.DataFrame(data.isnull().sum())
Outlier Analysis
#Plot boxplot to visualize Outliers
%matplotlib inline
plt.boxplot(data['Absenteeism time in hours'])
#save numeric names
```

```
cnames = ["Transportation expense", "Distance from Residence to Work", "Service time",
"Age", "Work load Average day", "Hit target","Son", "Pet", "Weight", "Height", "Body mass
index"]
#Detect and delete outliers from data
for i in cnames:
    print(i)
    q75, q25 = np.percentile(data.loc[:,i], [75 ,25])
    iqr = q75 - q25

    min = q25 - (iqr*1.5)
    max = q75 + (iqr*1.5)
    print(min)
    print(max)

    #Replace with NA
    data.loc[data[i] < min,:i] = np.nan
    data.loc[data[i] > max,:i] = np.nan

    #Calculate missing value
    pd.DataFrame(data.isnull().sum())

    #Impute with KNN
    data = pd.DataFrame(KNN(21).fit_transform(data), columns = data.columns)

Feature Selection
##Correlation analysis
#Correlation plot
df_corr = data.loc[:,cnames]
#Set the width and height of the plot
f, ax = plt.subplots(figsize=(7, 5))

#Generate correlation matrix
corr = df_corr.corr()

#Plot using seaborn library
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool), cmap=sns.diverging_palette(220,
10, as_cmap=True),
            square=True, ax=ax)
#Chisquare test of independence
```

```
#Save categorical variables
cat_names = ['ID', 'Reason for absence', 'Month of absence', 'Day of the week',
      'Seasons','Disciplinary failure', 'Education','Social drinker',
      'Social smoker']
#loop for chi square values
for i in cat_names:
   print(i)
   chi2, p, dof, ex = chi2_contingency(pd.crosstab(data['Absenteeism time in hours'], data[i]))
   print(p)
data = data.drop(['Body mass index'], axis=1)
Feature Scaling
#Normality check
%matplotlib inline
plt.hist(data['Height'], bins='auto')
cnames = ["Transportation expense", "Distance from Residence to Work", "Service time",
"Age", "Work load Average day",
        "Hit target","Son", "Pet", "Weight", "Height"]
#Nomalisation
for i in cnames:
   print(i)
   data[i] = (data[i] - data[i].min())/(data[i].max() - data[i].min())
Model Development
#Data Sampling
nrow= len(data.index)
train, test = train_test_split(data, test_size = 0.2)
train.shape
test.shape
#####Decision Tree Algortithm
from sklearn.tree import DecisionTreeRegressor
fit_dt= DecisionTreeRegressor(max_depth=2).fit(train.iloc[:,0:19],train.iloc[:,19])
fit_dt
predict_dt= fit_dt.predict(test.iloc[:,0:19])
#Calculate RMSE
def RMSE(actual, pred):
   return np.sqrt(((pred - actual) ** 2).mean())
RMSE(test.iloc[:,19],predict_dt)
#output = 13.009041545395686
######Random Forest Algorithm
from sklearn.ensemble import RandomForestRegressor
```

```
fit_rf = RandomForestRegressor(n_estimators = 100, random_state =
99).fit(train.iloc[:,0:19],train.iloc[:,19])
fit_rf
predict_rf= fit_rf.predict(test.iloc[:,0:19])
RMSE(test.iloc[:,19],predict_rf)
#output = 10.168282568103772
######Multiple Linear Regression
import statsmodels.api as sm
fit_lr = sm.OLS(train.iloc[:,19],train.iloc[:,0:19]).fit()
fit_lr.summary()
predict_lr = fit_lr.predict(test.iloc[:,0:19])
RMSE(test.iloc[:,19],predict_lr)
#output = 13.350487470207554
######KNN Implementation
from sklearn import neighbors
rmse_val = []          #to store rmse values for different k
for K in range(30):
   K = K+1
   fit_knn = neighbors.KNeighborsRegressor(n_neighbors = K)

   fit_knn.fit(train.iloc[:,0:19], train.iloc[:,19]) #fit the model

   predict_knn = fit_knn.predict(test.iloc[:,0:19]) #make prediction on test set
   error = RMSE(test.iloc[:,19] , predict_knn) #calculate rmse
   rmse_val.append(error) #store rmse values
   print('RMSE value for k= ' , K , 'is:', error)
#plotting the rmse values against k values
curve = pd.DataFrame(rmse_val)
curve.plot()
#K=12 is the value of neighbors for least RMSE.
#For K=12:
fit_knn = neighbors.KNeighborsRegressor(n_neighbors = 12)
fit_knn.fit(train.iloc[:,0:19], train.iloc[:,19]) #fit the model
predict_knn = fit_knn.predict(test.iloc[:,0:19]) #make prediction on test set
RMSE(test.iloc[:,19] , predict_knn)
#Thus, we find the "Random Forest Algorithm" gives us the best result with the least RMSE for
this dataset.
```

References

Mitchell, T. (1997). Machine Learning. McGraw Hill. p. 2

Brieman, Friedman, Olshen and Stone, Classification and Regression Trees, 1984