EMPLOYEE ABSENTEEISM

Project by: Abhishek Pandey

Linkedin: https://www.linkedin.com/in/abhishekpandey2505/

Email:abhishekpandey2505@gmail.com

Contents :

Chapter 1

Introduction

1.1 Problem Statement

XYZ is a courier company. As we appreciate that human capital plays an important role in collection, transportation and delivery. The company is passing through genuine issue of Absenteeism. The company has shared 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?

1.2 Data

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

Variable Information:

1. Individual identification (ID)

2. Reason for absence (ICD).

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

I Certain infectious and parasitic diseases II Neoplasms

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

IV Endocrine, nutritional and metabolic diseases V Mental and behavioural disorders

VI Diseases of the nervous system
VII Diseases of the eye and adnexa

VIII Diseases of the ear and mastoid process IX Diseases of the circulatory system

X Diseases of the respiratory system XI Diseases of the digestive system

XII Diseases of the skin and subcutaneous tissue

XIII Diseases of the musculoskeletal system and connective tissue XIV Diseases of the genitourinary system

XV Pregnancy, childbirth and the puerperium

XVI Certain conditions originating in the perinatal period

XVII Congenital malformations, deformations and chromosomal abnormalities

XVIII Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified XIX Injury, poisoning and certain other consequences of external causes

XX External causes of morbidity and mortality

XXI Factors influencing health status and contact with health services.

And 7 categories without (CID) patient follow-up (22), medical consultation (23), blood donation

(24), laboratory examination (25), unjustified absence (26), physiotherapy (27), dental consultation

(28).

3. Month of absence
4. Day of the week (Monday (2), Tuesday (3), Wednesday (4), Thursday (5), Friday (6))
5. Seasons (summer (1), autumn (2), winter (3), spring (4))
6. Transportation expense
7. Distance from Residence to Work (KMs)
8. Service time
9. Age
10. Work load Average/day
11. Hit target

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

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

14. Son (number of children)

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

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

17. Pet (number of pet)

18. Weight

19. Height

20. Body mass index

21. Absenteeism time in hours (target)

	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	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet
0	11	26.0	7.0	3	1	289.0	36.0	13.0	33.0	239,554	97.0	0.0	1.0	2.0	1.0	0.0	1.0
1	36	0.0	7.0	3	1	118.0	13.0	18.0	50.0	239,554	97.0	1.0	1.0	1.0	1.0	0.0	0.0
2	3	23.0	7.0	4	1	179.0	51.0	18.0	38.0	239,554	97.0	0.0	1.0	0.0	1.0	0.0	0.0
3	7	7.0	7.0	5	1	279.0	5.0	14.0	39.0	239,554	97.0	0.0	1.0	2.0	1.0	1.0	0.0
4	11	23.0	7.0	5	1	289.0	36.0	13.0	33.0	239,554	97.0	0.0	1.0	2.0	1.0	0.0	1.0
5	3	23.0	7.0	6	1	179.0	51.0	18.0	38.0	239,554	97.0	0.0	1.0	0.0	1.0	0.0	0.0
6	10	22.0	7.0	6	1	NaN	52.0	3.0	28.0	239,554	97.0	0.0	1.0	1.0	1.0	0.0	4.0
7	20	23.0	7.0	6	1	260.0	50.0	11.0	36.0	239,554	97.0	0.0	1.0	4.0	1.0	0.0	0.0
8	14	19.0	7.0	2	1	155.0	12.0	14.0	34.0	239,554	97.0	0.0	1.0	2.0	1.0	0.0	0.0
9	1	22.0	7.0	2	1	235.0	11.0	14.0	37.0	239,554	97.0	0.0	3.0	1.0	0.0	0.0	1.0

Chapter 2

Methodology

Pre Processing:

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. Data preprocessing prepares raw data for further processing.

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations. In this project we look at the distribution of categorical variables and continuous variables. We also look at the missing values in the data and the outliers present in the data

2.1>Missing Value Analysis

Missing data or missing values occur when there is no data value that has been stored for the variable in an observation. Missing value has very important and significant impact on the final result. If a variable has more than 30% of its values missing, then those values can be ignored, or the column itself is ignored. In our case, none of the columns have a high percentage of missing values. But here we have the highest percentage is of Body mass index(4.189189%).

1	Reason for absence	3	0.405405
2	Month of absence	1	0.135135
3	Day of the week	0	0.000000
4	Seasons	0	0.000000
5	Transportation expense	7	0.945946
6	Distance from Residence to Work	3	0.405405
7	Service time	3	0.405405
8	Age	3	0.405405
9	Work load Average per day	10	1.351351
10	Hit target	6	0.810811
11	Disciplinary failure	6	0.810811
12	Education	10	1.351351
13	Son	6	0.810811
14	Social drinker	3	0.405405
15	Social smoker	4	0.540541
16	Pet	2	0.270270
17	Weight	1	0.135135
18	Height	14	1.891892
19	Body mass index	31	4.189189
20	Absenteeism time in hours	22	2.972973

To treat these missing values, we need to choose a method which should be closer. Here we are trying to treat these by mean method, median method and interpolate method. So by this we will get to know which is better. We have made three copies of the data frame (df1,df2,df3). And we have made entry as NaN and then we tried all the methods so that we can check which is better.

```
df1= absent.copy()
df2= absent.copy()
df3= absent.copy()
```

```
df1.iloc[2,5]=np.nan
df2.iloc[2,5]=np.nan
df3.iloc[2,5]=np.nan
```

```
df1.iloc[2,5]
```

```
nan
```

```
#using mean method
df1['Transportation expense'] = df1['Transportation expense'].fillna(df1['Transportation expense'].mean())
df1.iloc[2,5]
```

```
221.0928961748634
```

```
#using median method
df2['Transportation expense'] = df2['Transportation expense'].fillna(df2['Transportation expense'].median())
df2.iloc[2,5]
```

```
225.0
```

```
#using interpolate
df3['Transportation expense'] = df3['Transportation expense'].interpolate(method = 'nearest', limit_direction = 'both')
df3.iloc[2,5]
```



```
118.0
```

Here Interpolate result is much closer to our actual result, so we will use interpolate to for missing values because it doesn't make another category in categorical variable(so we are using this) .
After doing the missing value treatment, we can see the missing value in data.

```
Out[416]:  ID                                  0
           Reason for absence                  0
           Month of absence                    0
           Day of the week                     0
           Seasons                             0
           Transportation expense              0
           Distance from Residence to Work     0
           Service time                        0
           Age                                 0
           Work load Average per day           0
           Hit target                          0
           Disciplinary failure                0
           Education                           0
           Son                                 0
           Social drinker                      0
           Social smoker                       0
           Pet                                 0
           Weight                              0
           Height                              0
           Body mass index                     0
           Absenteeism time in hours           0
           dtype: int64
```

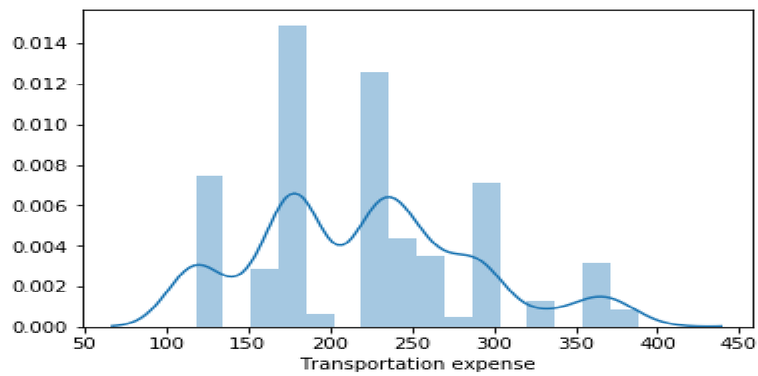
2.2>Outlier Analysis:

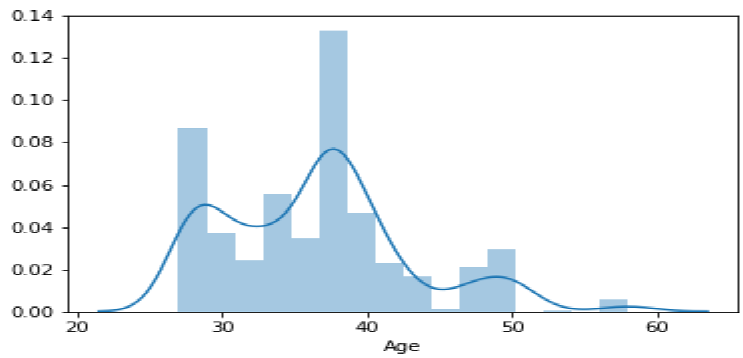
An outlier may also be explained as a piece of data or observation that deviates drastically from the given norm or average of the data set. An outlier may be caused simply by chance, but it may also indicate measurement error or that the given data set has a heavy-tailed distribution.

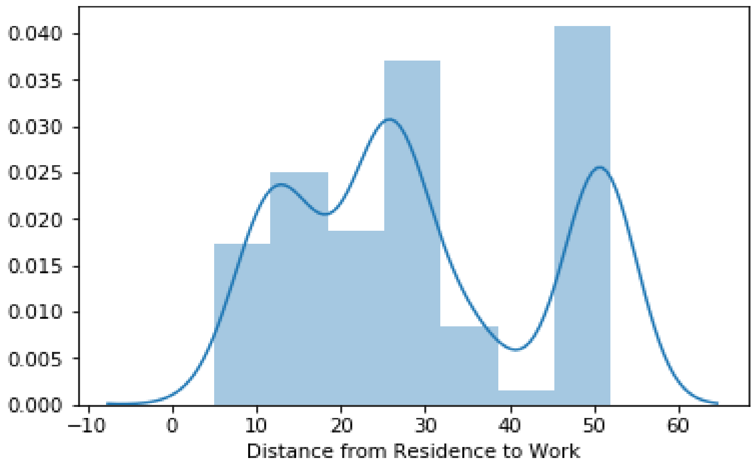
Here we have to methods to treat outliers which depends on the graph of the column , If the data we have is of normal type then we have to use the Z score method else for skwed type we have to use Numerical outlier.
We have made num_colnames(for names of column which are numeric)

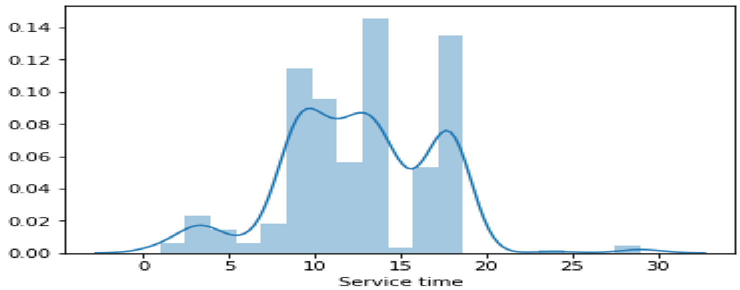
num_colnames= ['Transportation expense' ,'Distance from Residence to Work', 'Service time', 'Age' ,'Work load Average per day','Hit target', 'Height','Weight','Body mass index'] cat_colnames = ['Reason for absence','Month of absence', 'Day of the week', 'Seasons', 'Disciplinary failure', 'Education', 'Son', 'Social drinker', 'Social smoker', 'Pet']

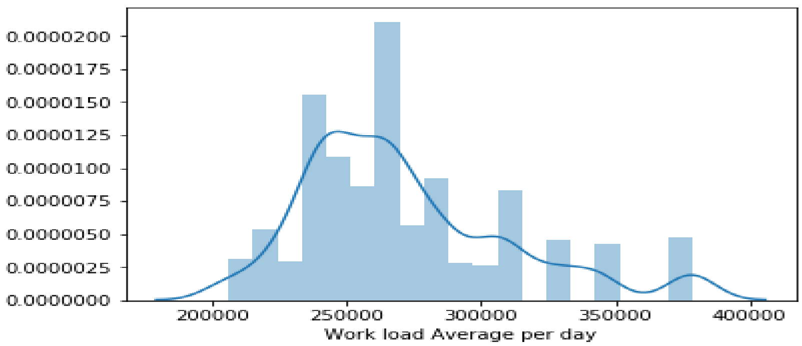
2.3>Plotting all the graphs (numeric columns)

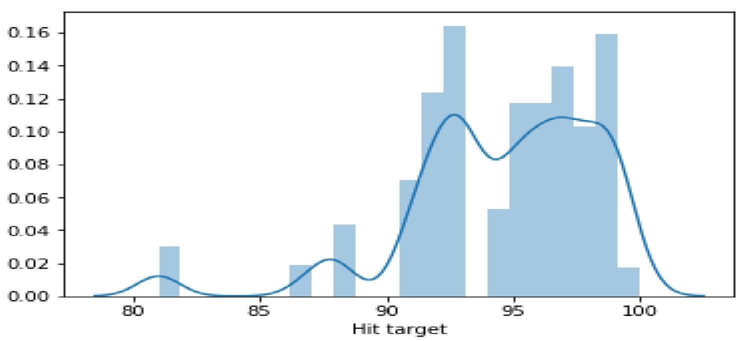


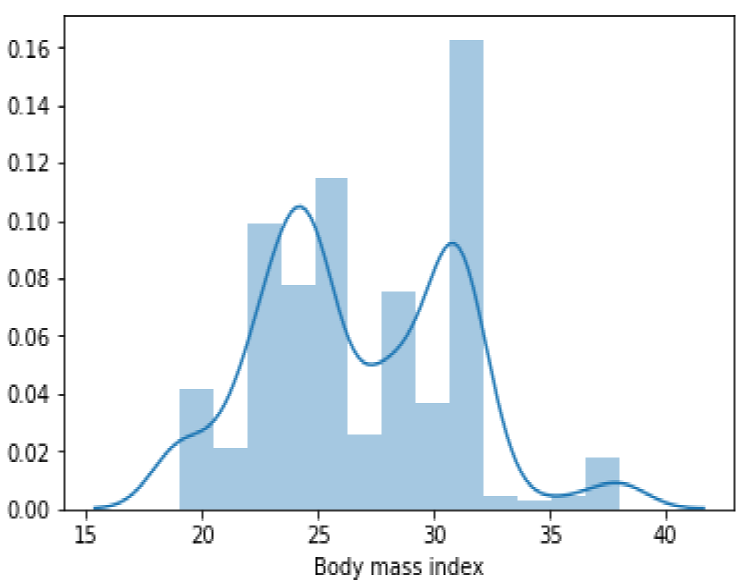


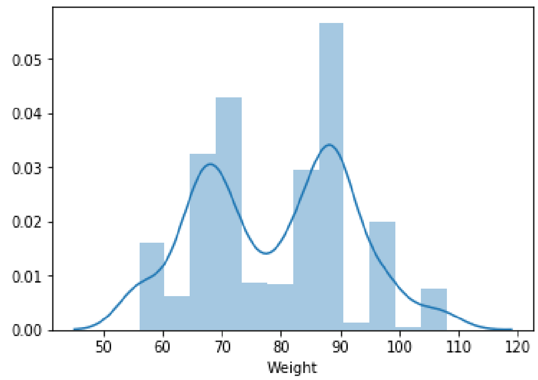


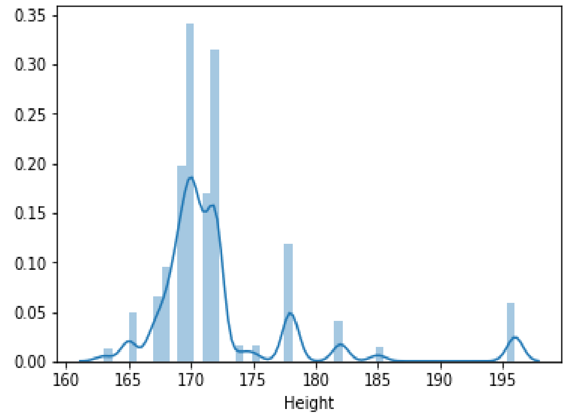

0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
5
10
15
20
25
30
Service time


0.0000200
0.0000175
0.0000150
0.0000125
0.0000100
0.0000075
0.0000050
0.0000025
0.0000000
200000
250000
300000
350000
400000
Work load Average per day


0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
80
85
90
95
100
Hit target


0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
15
20
25
30
35
40
Body mass index

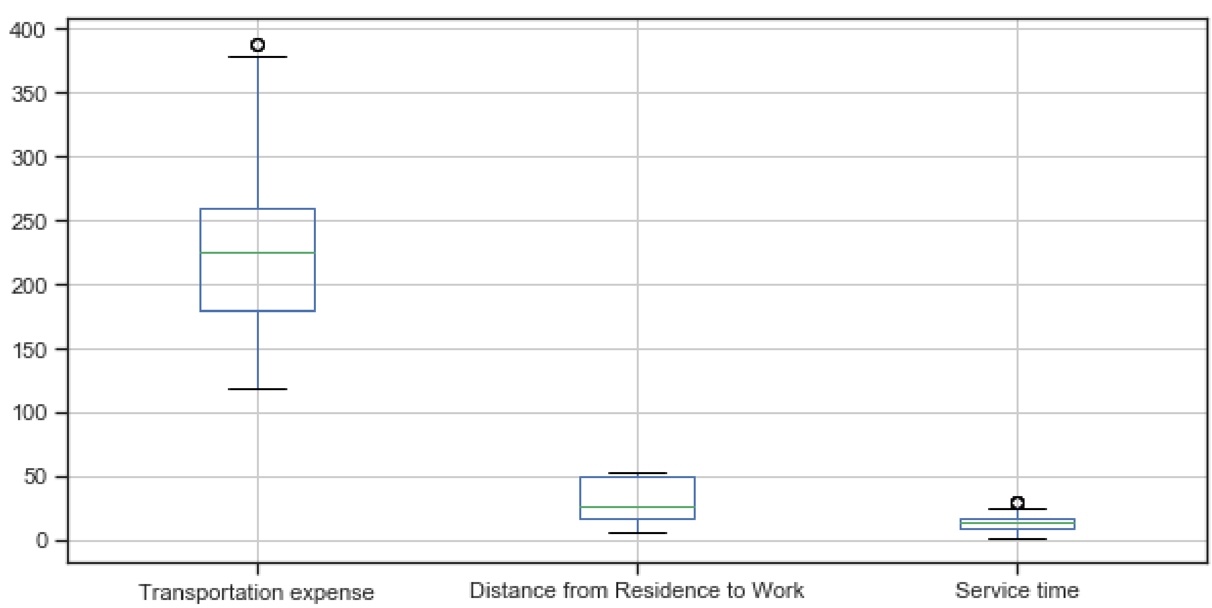


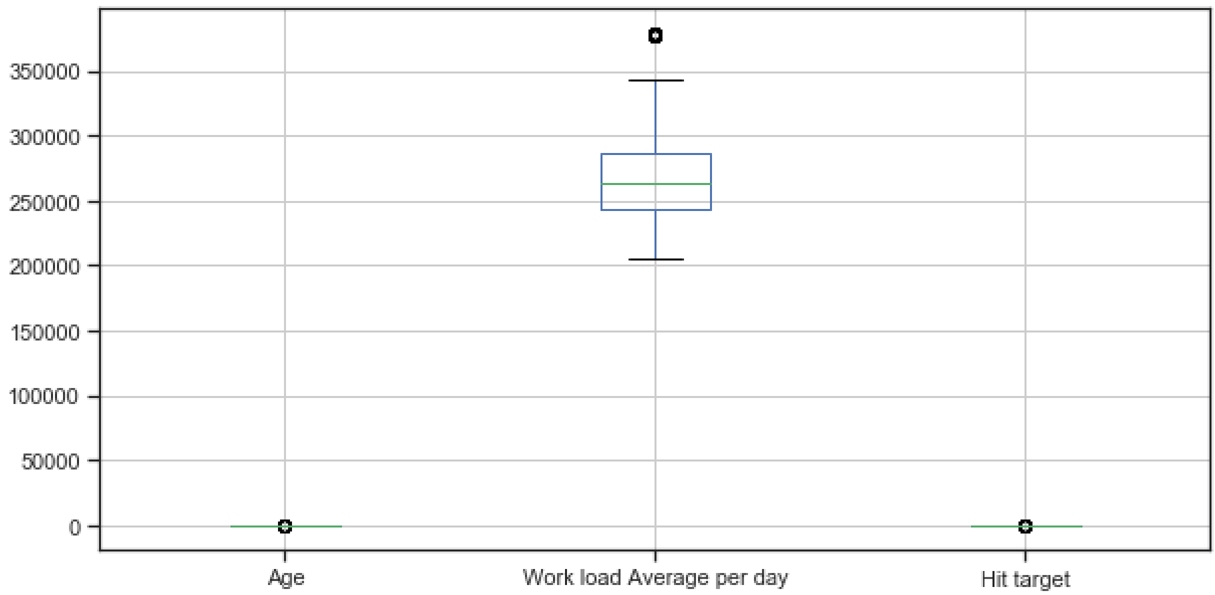


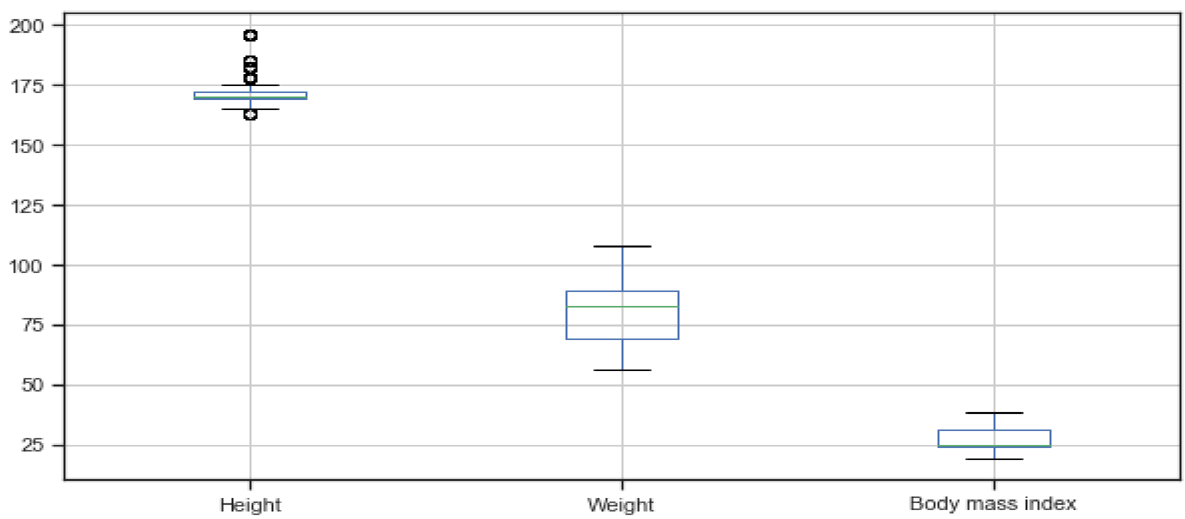
As we can see every variable from the 9 continious is skwed. So we need to use normalised method to remove outliers.(if it would have been normal , then we should have used the z score method)

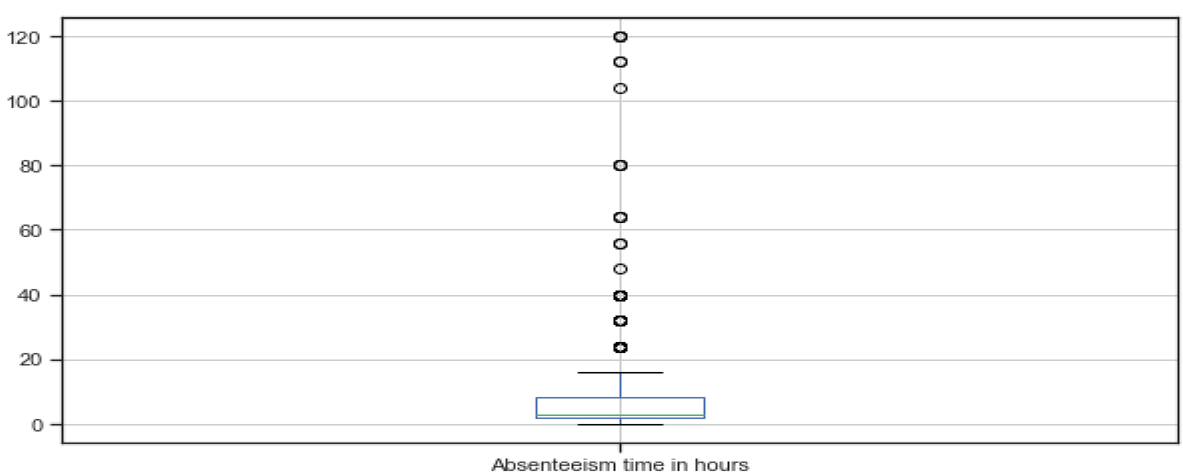
Now we need to see the box plot of each variable to see the outiers.

2.4>Box Plot of numerical variable to see outliers

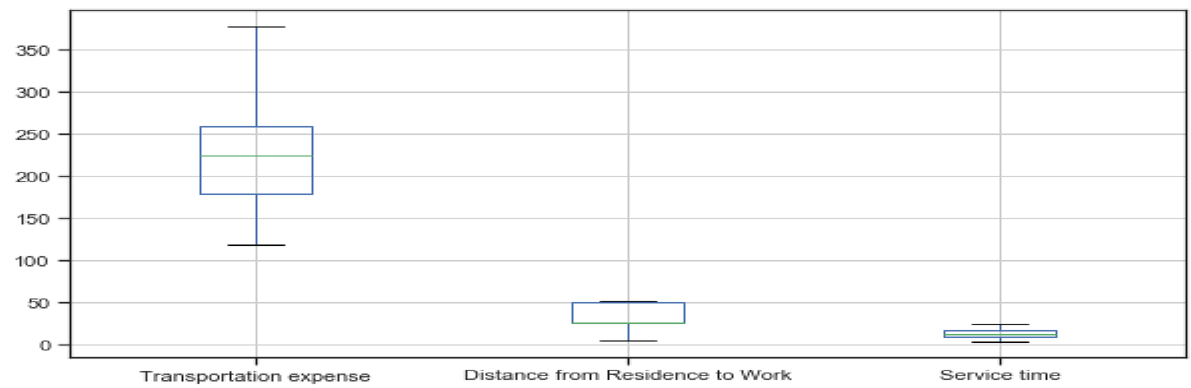


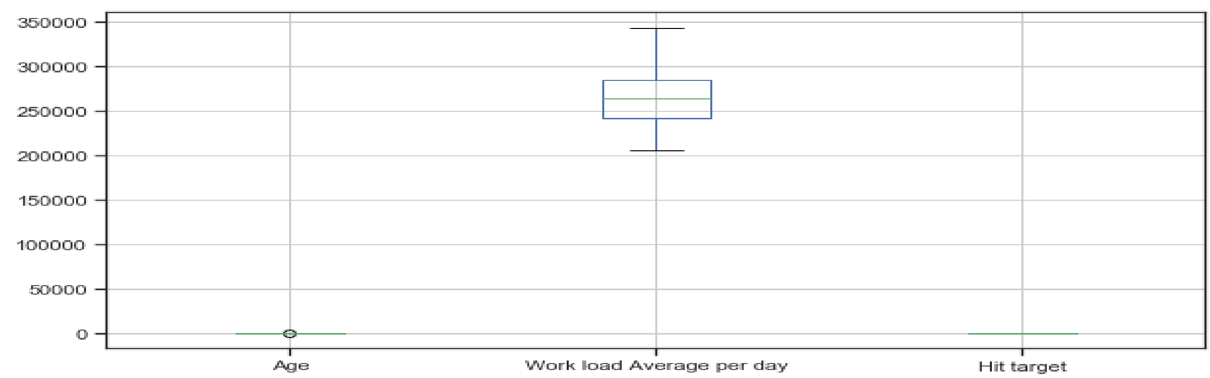

350000
300000
250000
200000
150000
100000
50000
0
Age
Work load Average per day
Hit target

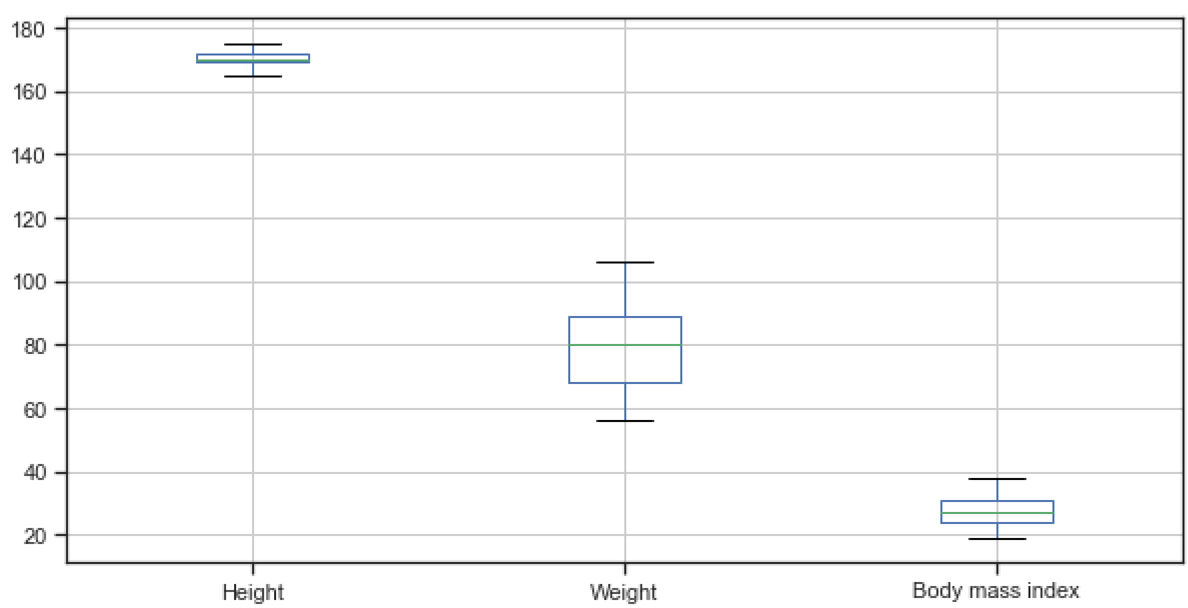

200
175
150
125
100
75
50
25
Height
Weight
Body mass index

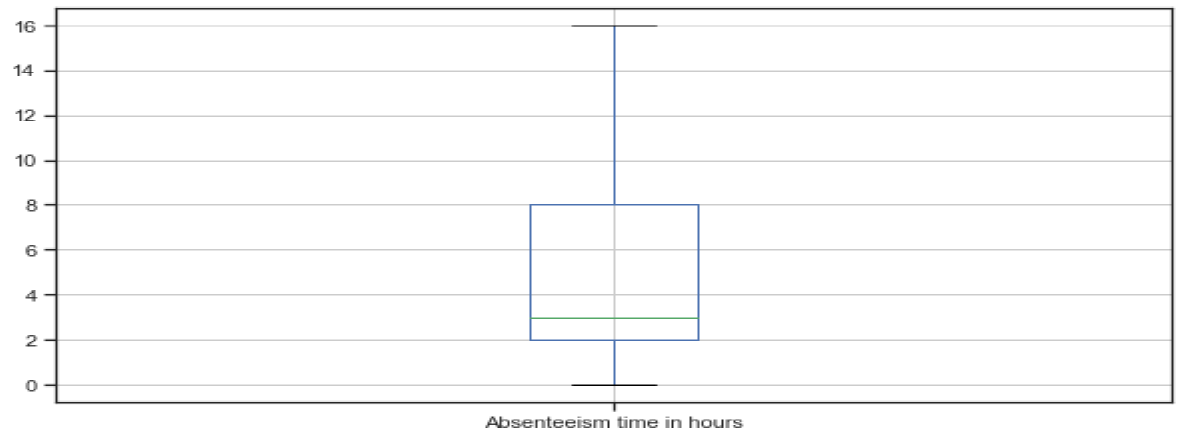

120
100
80
60
40
20
0
Absenteeism time in hours

We need to remove these outliers, after removal we can proceed further.(please see the python code).After removal of the outliers, here are our box plot.









2.5>Correlation among variables

2.5.1> Chi-square test

We also need to see the relation between the categorical columns. For that we need to use chi-square test. and if there is any other column which zero other than diagonal elements, it means our null hypothesis is false.

	Reason for absence	Month of absence	Day of the week	Seasons	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet
Reason for absence	0.000000e+00	1.241752e-15	0.096612	2.846325e-15	4.354055e-80	1.648742e-23	1.009881e-18	2.413486e-07	1.292721e-11	7.802177e-11
Month of absence	1.241752e-15	0.000000e+00	0.395669	1.092117e-254	5.648100e-03	2.541108e-03	2.791113e-04	6.177169e-04	7.991270e-02	9.785995e-27
Day of the week	9.661167e-02	3.956689e-01	0.000000	1.536856e-01	8.721748e-01	7.248537e-01	1.391017e-04	4.874375e-01	7.042249e-01	4.184699e-01
Seasons	2.846325e-15	1.092117e-254	0.153686	0.000000e+00	9.051402e-04	6.056720e-02	1.353561e-04	7.110981e-03	4.014934e-02	2.092395e-03
Disciplinary failure	4.354055e-80	5.648100e-03	0.872175	9.051402e-04	0.000000e+00	4.085935e-01	1.607179e-02	2.456006e-01	8.156237e-01	5.383857e-01
Education	1.648742e-23	2.541108e-03	0.724854	6.056720e-02	4.085935e-01	0.000000e+00	2.716903e-20	3.668963e-24	1.261443e-61	4.862738e-05
Son	1.009881e-18	2.791113e-04	0.000139	1.353561e-04	1.607179e-02	2.716903e-20	0.000000e+00	1.451098e-11	1.221988e-32	3.808292e-88
Social drinker	2.413486e-07	6.177169e-04	0.487438	7.110981e-03	2.456006e-01	3.668963e-24	1.451098e-11	0.000000e+00	5.320388e-03	4.201192e-22
Social smoker	1.292721e-11	7.991270e-02	0.704225	4.014934e-02	8.156237e-01	1.261443e-61	1.221988e-32	5.320388e-03	0.000000e+00	4.991711e-04
Pet	7.802177e-11	9.785995e-27	0.418470	2.092395e-03	5.383857e-01	4.862738e-05	3.808292e-88	4.201192e-22	4.991711e-04	0.000000e+00

However, here there is no element zero so it means our categorical column are independent.

2.5.2>Feature selection (Correlation in continuous variable)

Feature selection is used to determine which features are of really of use because if there are two variable which are highly correlated then means they both give same information so we need to get rid of one.

	ID	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average per day	Hit target	Weight	Height	Body mass index	Absenteeism time in hours
ID	1	0.223261	0.483321	0.272346	0.0417766	0.0910246	0.0187892	0.253634	0.0775345	0.2934	0.0244201
Transportation expense	0.223261	1	0.261708	0.340092	0.222483	0.00789677	0.0810645	0.20645	0.19542	0.131905	0.0527856
Distance from Residence to Work	0.483321	0.261708	1	0.129386	0.14715	0.0657419	0.0144214	0.0464721	0.355421	0.124452	0.0904486
Service time	0.272346	0.340092	0.129386	1	0.667365	0.00201613	0.00889825	0.455719	0.0512381	0.489023	0.00979923
Age	0.0417766	0.222483	0.14715	0.667365	1	0.045969	0.0370413	0.4183	0.0625423	0.448094	0.0705694
Work load Average per day	0.0910246	0.00789677	0.0657419	0.00201613	0.045969	1	0.0896744	0.036217	0.105995	0.0866531	0.0261617
Hit target	0.0187892	0.0810645	0.0144214	0.00889825	0.0370413	0.0896744	1	0.044884	0.0901376	0.0716061	0.0207251
Weight	0.253634	0.20645	0.0464721	0.455719	0.4183	0.036217	0.044884	1	0.303356	0.865847	0.0135156
Height	0.0775345	0.19542	0.355421	0.0512381	0.0625423	0.105995	0.0901376	0.303356	1	0.12773	0.0876316
Body mass index	0.2934	0.131905	0.124452	0.489023	0.448094	0.0866531	0.0716061	0.865847	0.12773	1	0.0523162
Absenteeism time in hours	0.0244201	0.0527856	0.0904486	0.00979923	0.0705694	0.0261617	0.0207251	0.0135156	0.0876316	0.0523162	1

Here we found out that Body mass index and weight have high correlation (0.944465), so we need to drop body mass index.
Another thing is about 'Distance from Residence to Work' because it has very low correlation with target(Absenteeism time in hours)[0.00663594]. so we need to drop both.

2.5.3>Feature scaling:

We need to scale our data because the dataset will contain features highly varying in magnitudes, units and range. But since, most of the machine learning algorithms use Eucledian distance between two data points in their computations, this is a problem.
After scaling , our data will be like this :

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Service time	Age	Work load Average per day	Hit target	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Weight
0	11	26.0	7.0	3	1	0.657692	0.476190	0.230769	0.244925	0.75	0.0	1.0	2.0	1.0	0.0	1.0	0.68
2	3	23.0	7.0	4	1	0.234615	0.714286	0.423077	0.244925	0.75	0.0	1.0	0.0	1.0	0.0	0.0	0.66
3	7	7.0	7.0	5	1	0.619231	0.523810	0.461538	0.244925	0.75	0.0	1.0	2.0	1.0	1.0	0.0	0.24
4	11	23.0	7.0	5	1	0.657692	0.476190	0.230769	0.244925	0.75	0.0	1.0	2.0	1.0	0.0	1.0	0.68
5	3	23.0	7.0	6	1	0.234615	0.714286	0.423077	0.244925	0.75	0.0	1.0	0.0	1.0	0.0	0.0	0.66

Chapter 3

Modeling

After doing these pre processing we need to train our model so that we can predict the outcome(absenteeism hours) in future. Here we need to split our data and then train our model.

3.1> Splitting data: we need to divide the data into train(80 percent) and test(20 percent).

Model selection: we need to decide which model we need to use for our data. The target variable in our model is a continuous variable(Absenteeism time in hour).So the models that we choose are Decision Tree and Random Forest, Linear Regression,OLS(python). The error metric chosen for the given problem statement is mean_absolute_error.

3.2> Decision Tree:

decision tree

```
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error

# decsion tree model
dt_model = DecisionTreeRegressor(random_state = 1).fit(x_train,y_train)

# Perdict for x_test
dt_pred = dt_model.predict(x_test)

# data frame for actual and predicted values
df_dt = pd.DataFrame({'actual': y_test, 'pred': dt_pred})
print(df_dt.head())

error_dt=mean_absolute_error(test.iloc[:,18], dt_pred)

# errors and accuracy calculation
print("MEAN ABSOLUTE ERROR "+ str(error_dt))
print("ACCURACY "+ str((1-(error_dt))*100))
```

Decision tree builds regression is in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.

	Mae	Accuracy= (1-mae)*100
Python	0.1328855140186916	86.71144859813084
R	0.1214914	87.85086

3.3>Random Forest: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees.

Random forest

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error

#forest model using RandomForestRegressor
rf_model = RandomForestRegressor(n_estimators = 500, random_state = 1).fit(x_train,y_train)

#Perdict for test cases
rf_pred = rf_model.predict(x_test)

#Create data frame for actual and predicted values
df_rf = pd.DataFrame({'actual': y_test, 'pred': rf_pred})

print(df_rf.head())

error_rf=mean_absolute_error(test.iloc[:,18], rf_pred)

# errors and accuracy calculation
print("MEAN ABSOLUTE ERROR "+ str(error_rf))
print("ACCURACY "+ str((1-(error_rf))*100))
```

	Mae	Accuracy= (1-mae)*100
Python	0.10759835057854916	89.24016494214509
R	0.1139508	88.60492

3.4>Linear : The technique uses statistical calculations to plot a trend line in a set of data points. The trend line could be anything from the number of people diagnosed with skin cancer to the financial performance of a company. Linear regression shows a relationship between an independent variable and a dependent variable being studied.

Linear

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error

#Train the model
lr_model = LinearRegression().fit(x_train , y_train)

#Perdiction for x_test
lr_pred = lr_model.predict(x_test)

#Cdata frame for actual and predicted values
df_lr = pd.DataFrame({'actual': y_test, 'pred': lr_pred})
print(df_lr.head())

error_lr=mean_absolute_error(test.iloc[:,18], lr_pred)

# errors and accuracy calculation
print("MEAN ABSOLUTE ERROR "+ str(error_lr))
print("ACCURACY "+ str((1-(error_lr))*100))

lr_model.score(x_test,y_test)
plt.plot()
```

	Mae	Accuracy= (1-mae)*100
Python	0.1279764773553122	87.20235226446877
R	0.1451811	85.48189

3.5>OLS method : Ordinary least squares (OLS) regression is a statistical method of analysis that estimates the relationship between one or more independent variables and a dependent variable; the method estimates the relationship by minimizing the sum of the squares in the difference between the observed and predicted values of the dependent variable configured as a straight line

OLS

```
]: #importing statsmodels.api
import statsmodels.api as sm

#model.score(x_test, y_test)
model = sm.OLS(y_train, x_train.astype(float)).fit()

#parameters of model(OLS)
model.params

#Perdict for test data
ols_pred= model.predict(x_test)

#Create data frame for actual and predicted values
df_ols = pd.DataFrame({'actual': y_test, 'pred': ols_pred})
print(df_ols.head())


#mean asbolute error
error_ols=mean_absolute_error(test.iloc[:,18], ols_pred)

# errors and accuracy calculation
print("MEAN ABSOLUTE ERROR "+ str(error_ols))
print("ACCURACY "+ str((1-(error_ols))*100))
```

	Mae	Accuracy= (1-mae)*100
Python	0.1312617922849797	86.87382077150203

Chapter 4

Conclusion

4.1> Model Evaluation

Now that we have a few models for predicting the target variable, we need to decide which one to choose. We need to decide our model on basis of *Predictive performance* as the criteria to compare and evaluate 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.

4.1.1> Mean Absolute Error (MAE)

MAE is one of the error measures used to calculate the predictive performance of the model. We will apply this measure to our models that we have generated in the previous section.

```
from   sklearn.metrics  import  mean_absolute_error

 error_dt = mean_absolute_error(test.iloc[:,18], dt_pred)
```

4.1.2> Model Selection

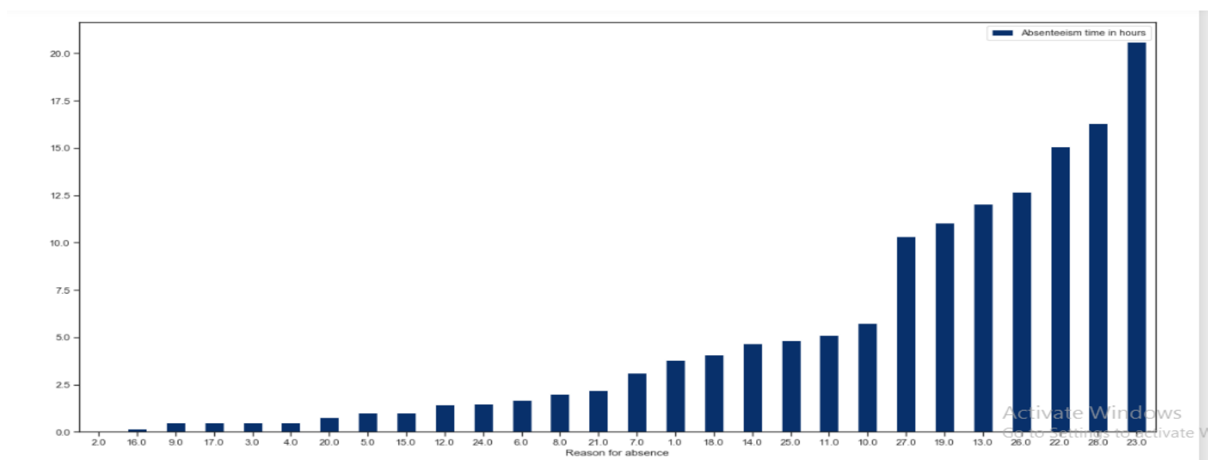
We can see that all the models perform well on the test data, However Random forest is working well and giving more accuracy comparative to them. Therefore, we can select either of the two models without any loss of information.

4.2> Answer to asked questions

Q1> what changes a company should bring to reduce the number of absenteeism?

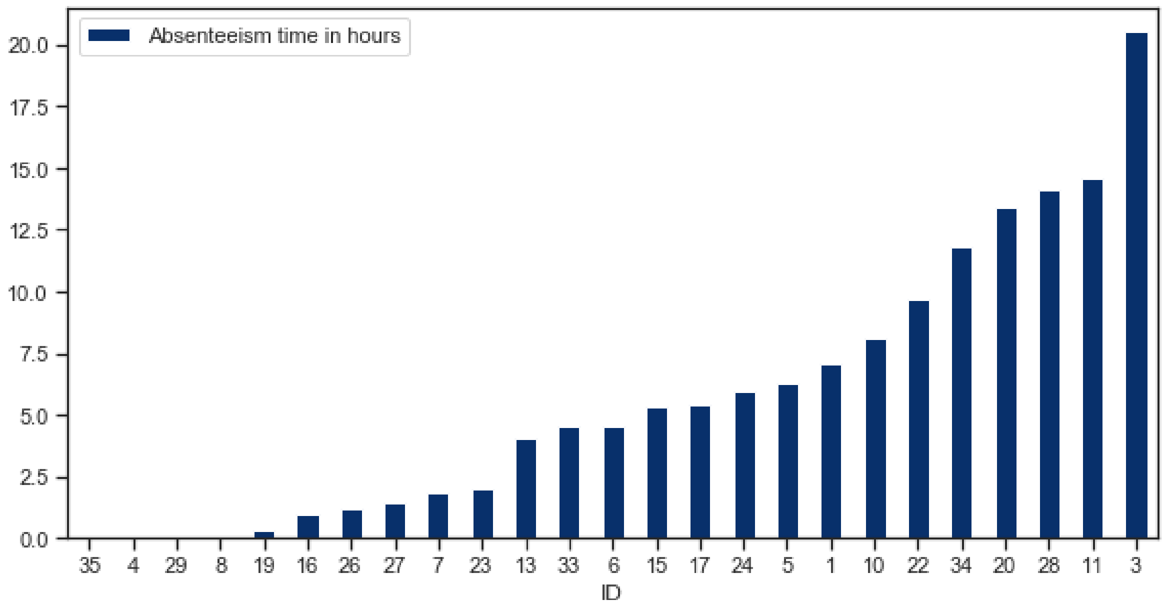
Ans : 1>> Reason for absence

Here 23 (medical consultation) is the main factor that is 14.39 %. and other is 28(dental consultation) so the company need to have a doctor and dentist in office so that employee could go there first. and other thing is about having low work pressure . we should need to give employees health guidelines of the dental issues , We can conduct seminars and sessions for dental issues and work life balance so that they can take better care of their health.

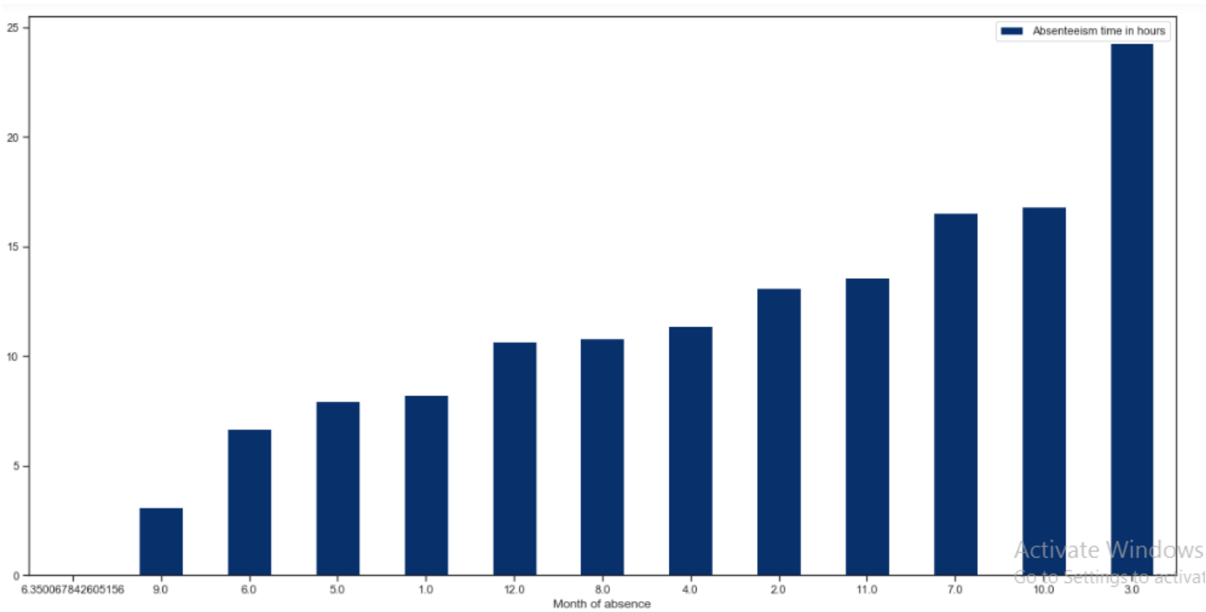


2>> the IDs

The top five ids (3,11,28,20,34) should be given proper warning or the company can take actions against them.



3>> Month of absence



Top 3 months in order of Absenteeism time are:

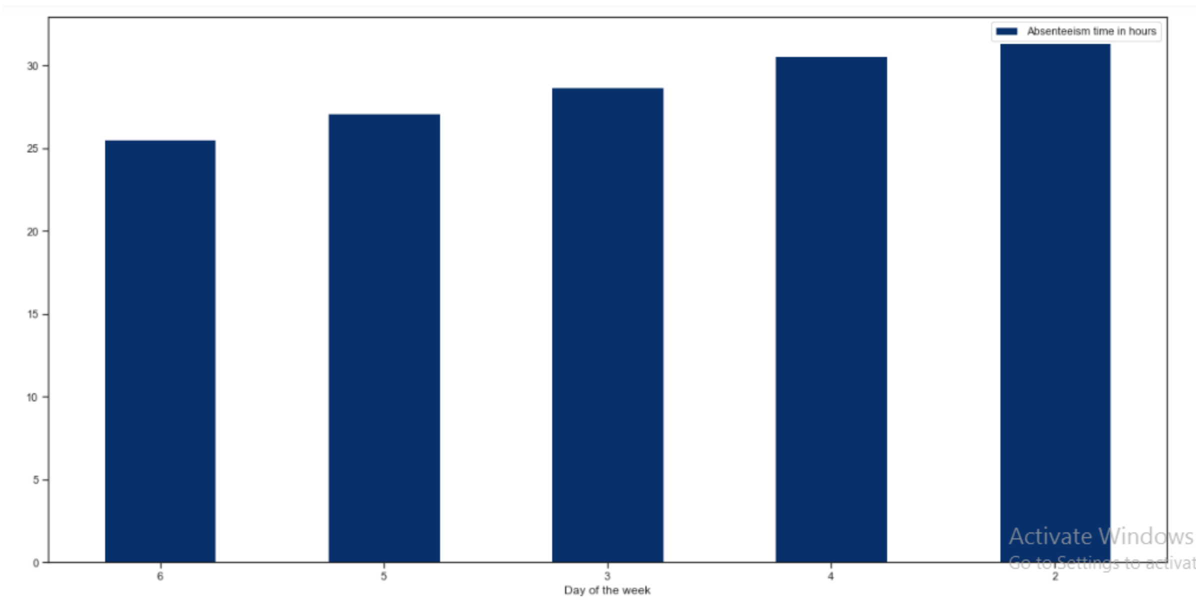
Month 3:March - 16.97 % of total time

Month 10:October - 11.73 % of total time

Month 7:July - 11.56 % of total time

The company can ask the employees to not take leave in march specifically. As march is end of financial year , that might be reason for leaves. The company can keep specific sessions(motivational) for employees , so that they can stretch their limits for that month.

4>> Day of the week



Top 3 days in order of Absenteeism time are:

Day 2:Monday - 21.90 % of total time

Day 4:Wednesday - 21.33 % of total time

Day 3:Tuesday - 20.02% of total time

Company can ask the employees to not take leave on Mondays. The company can also keep any activity (related to fun) so that employee come on Monday with some motivation.

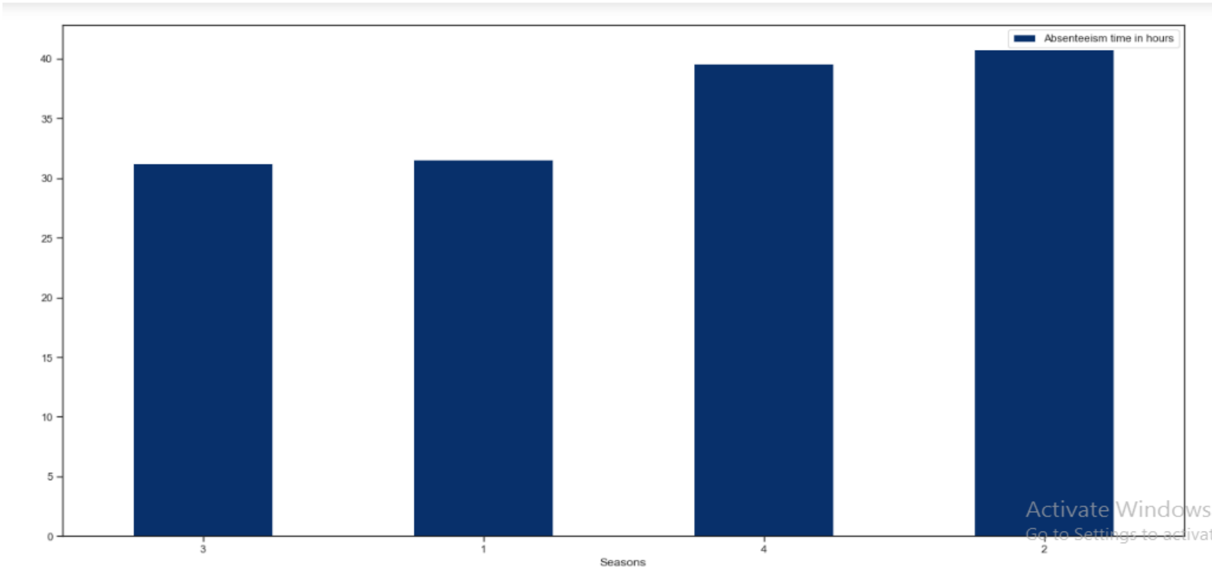
5>> Seasons

Top 2 Season in order of Absenteeism time are:

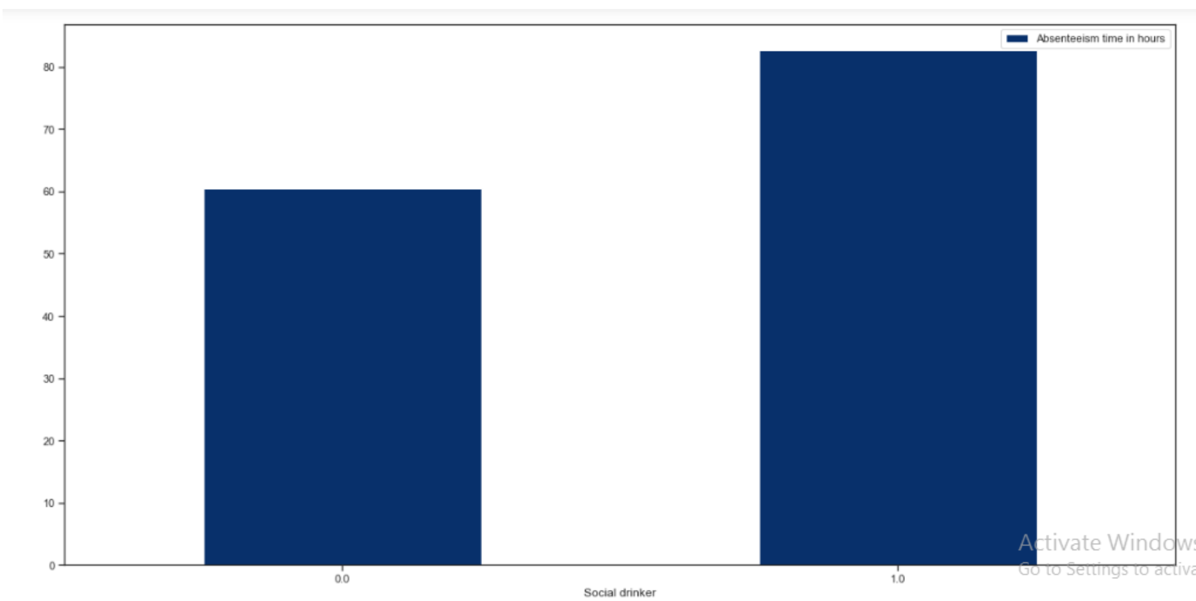
Season 2: Autumn - 28.49% of total time

Season 4: Spring - 27.66% of total time

Company should ask the employees to reduce their leaves for this season.



6>> Social drinker



Top reason in order of Absenteeism time are:

social drinker (yes) - 57.76% of total time

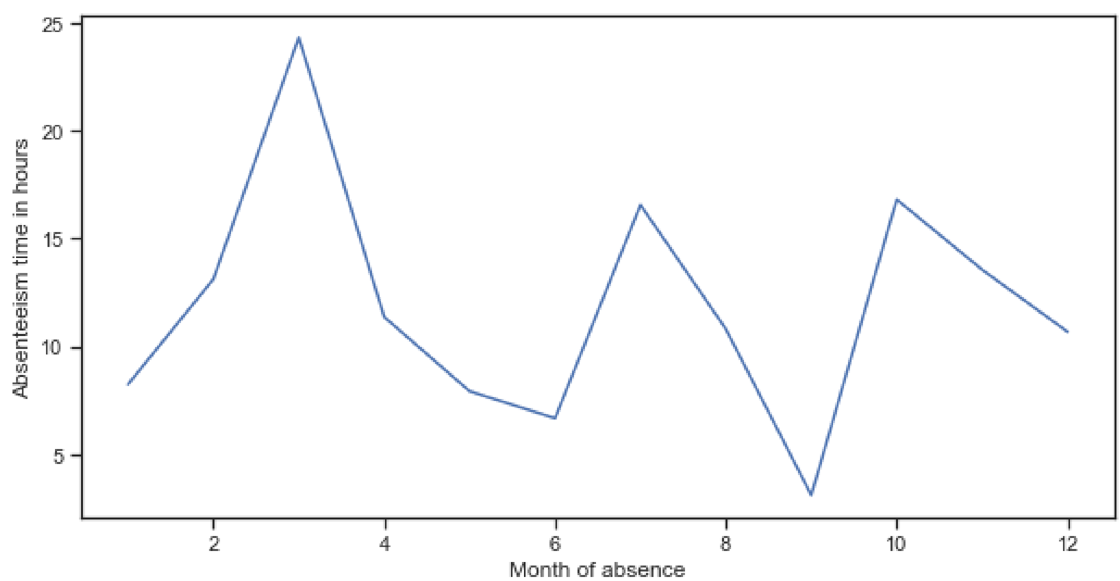
social drinker (No) - 42.23% of total time

>>company should ask these employees to reduce the amount to alcohol during weekdays.

Q2> How much losses every month can we project in 2011 if same trend of absenteeism continues?

Ans: **Here, in this case if the same trends continue then,**

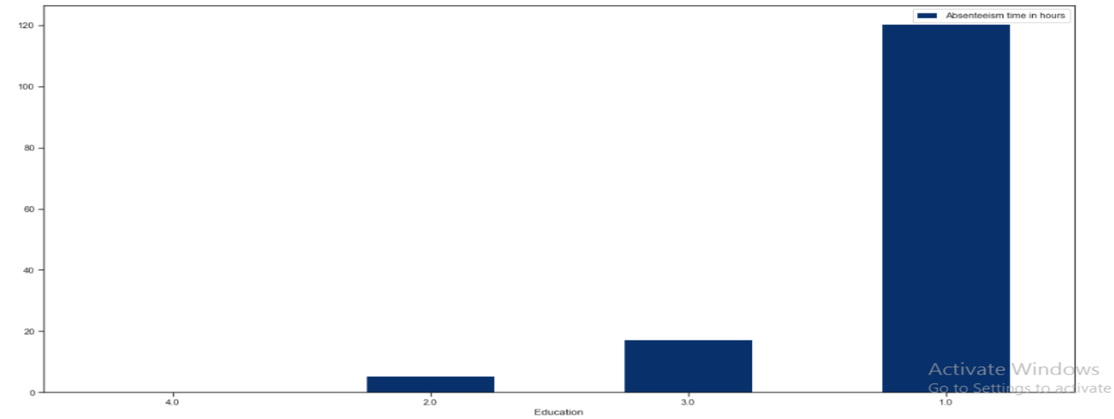
Absenteeism time in hours	Month of absence
8.2500	1
13.1250	2
24.3125	3
11.3750	4
7.9375	5
6.6875	6
16.5625	7
10.8125	8
3.1250	9
16.8125	10
13.5625	11
10.6875	12

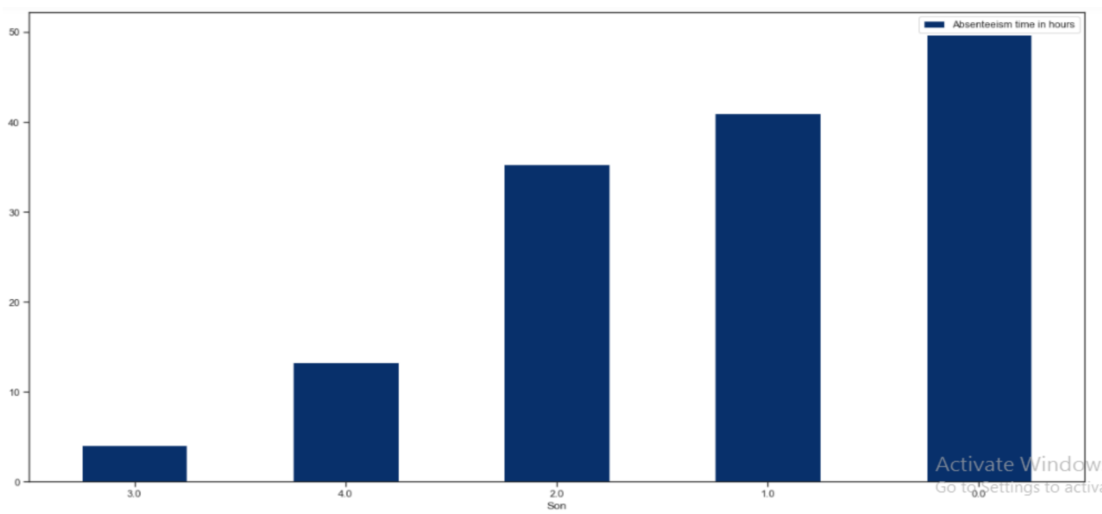


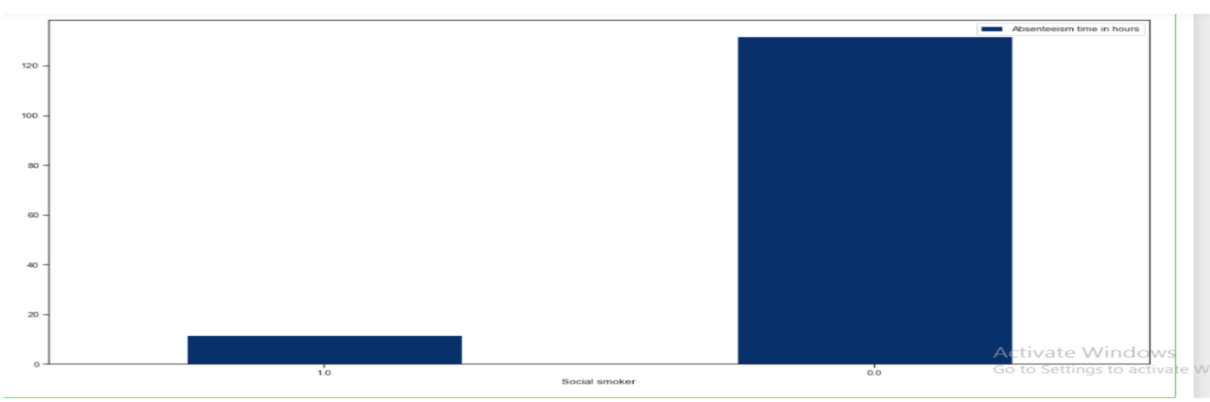
The maximum number of absenteeism hours will be in month of march and minimum will be in month of September

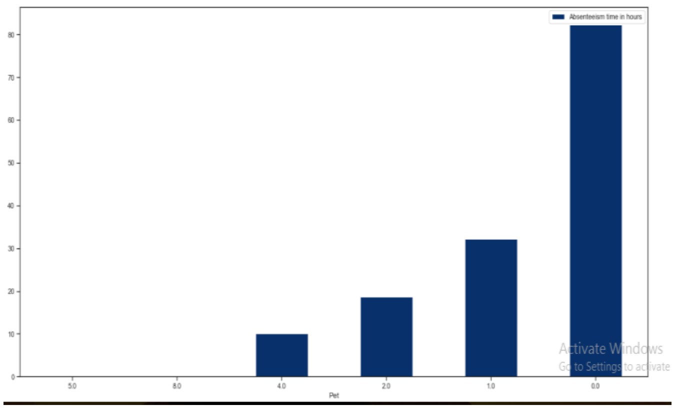
Chapter 5

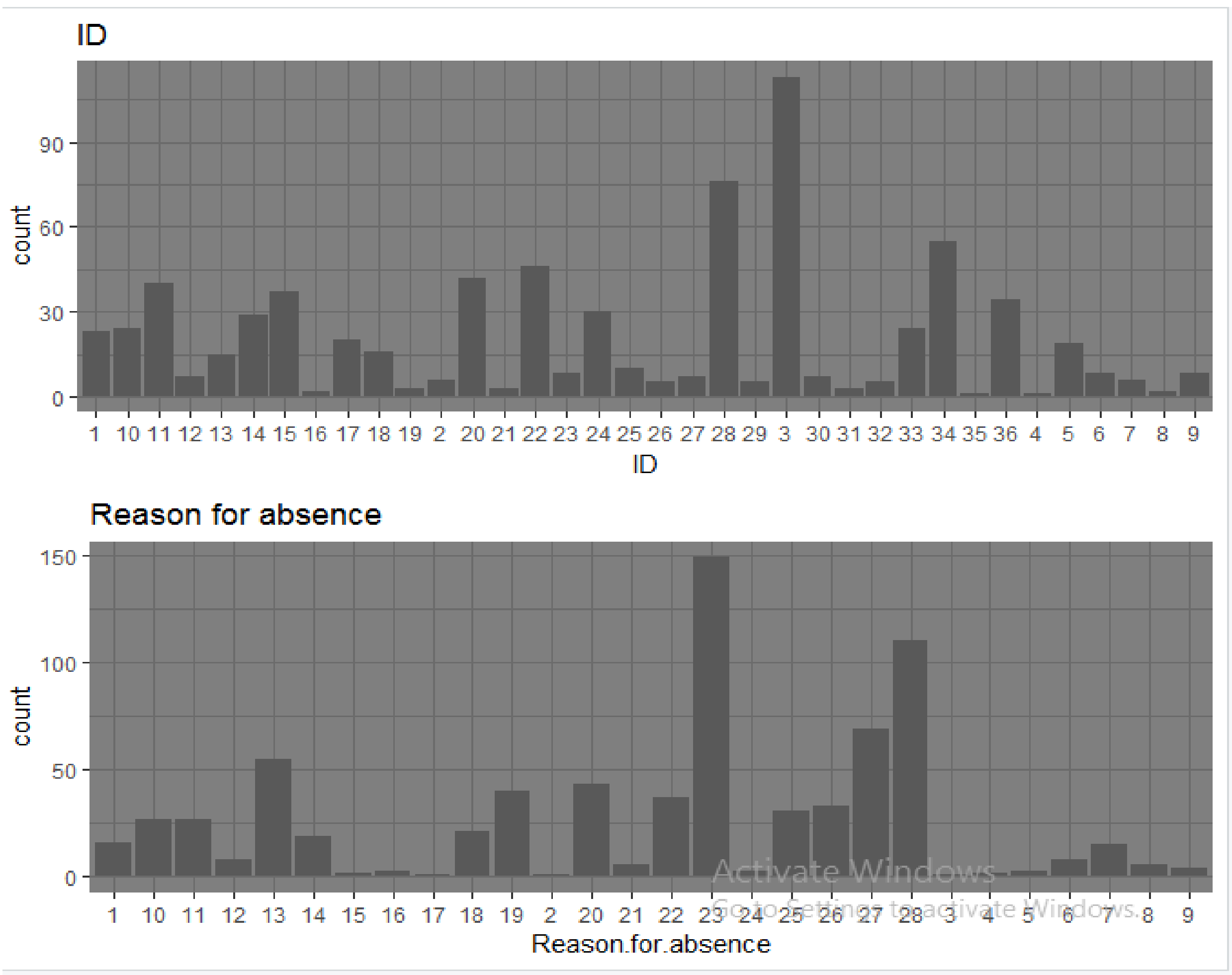
5.1>>Appendix A - Extra Figures

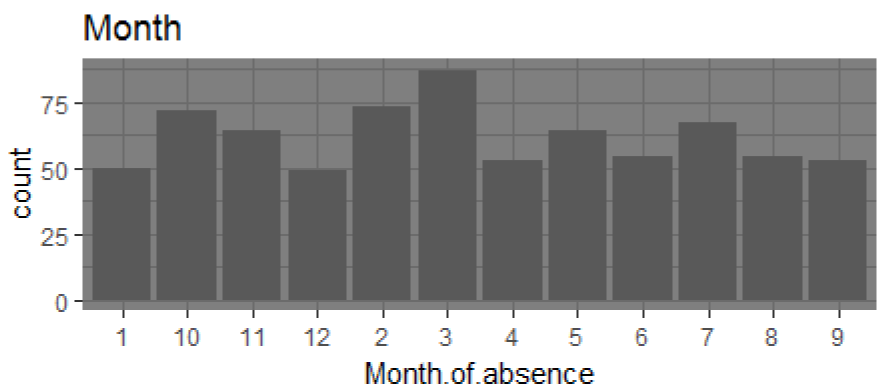

Absenteeism time in hours
120
100
80
60
40
20
0
4.0
2.0
3.0
1.0
Education

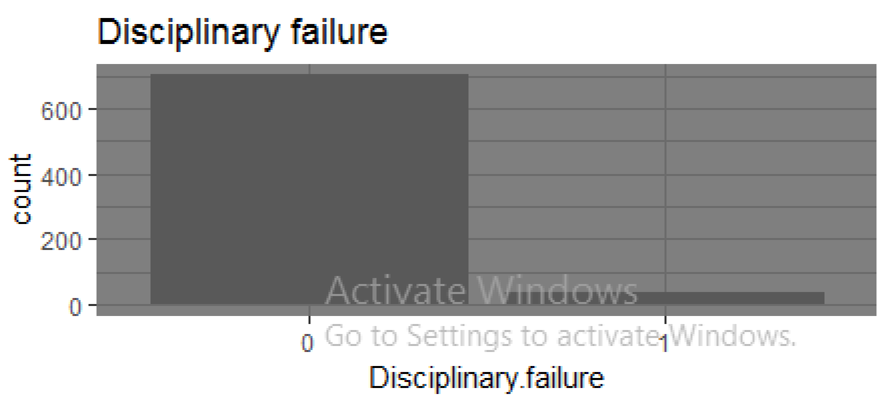

Absenteeism time in hours
50
40
30
20
10
0
3.0
4.0
2.0
1.0
0.0
Son

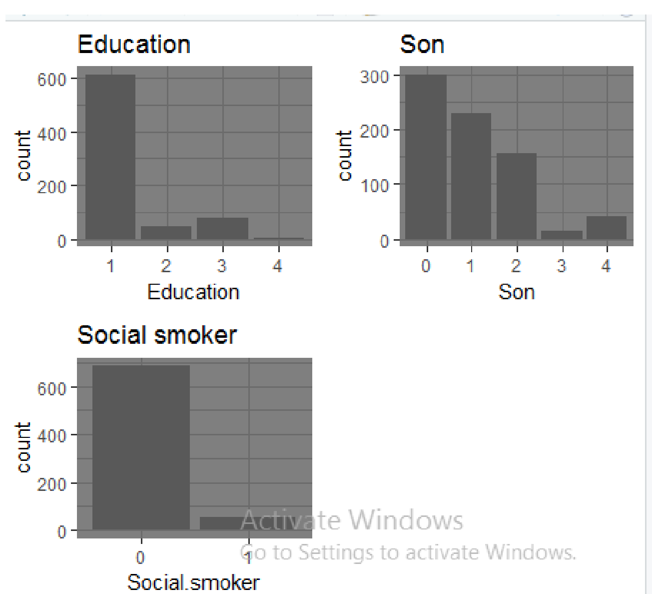

Absenteeism time in hours
120
100
80
60
40
20
0
1.0
0.0
Social smoker


Absenteeism time in hours
80
70
60
50
40
30
20
10
0
5.0
8.0
4.0
2.0
1.0
0.0
Pet
Activate Windows
Go to Settings to activate

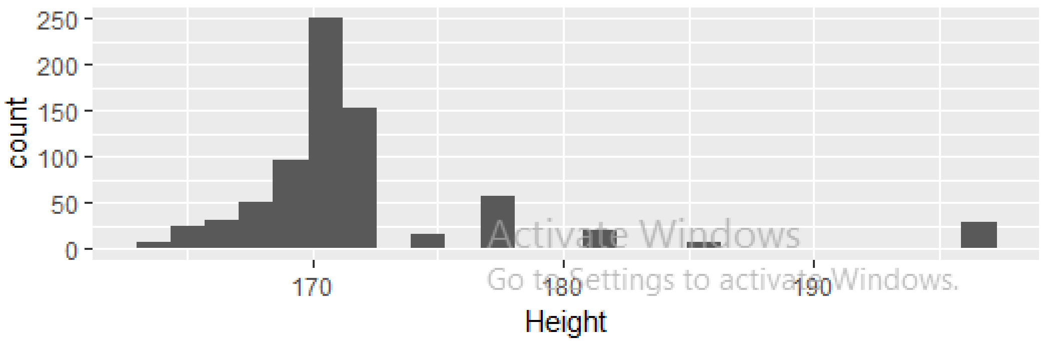

ID
count
90
60
30
0
1 10 11 12 13 14 15 16 17 18 19 2 20 21 22 23 24 25 26 27 28 29 3 30 31 32 33 34 35 36 4 5 6 7 8 9
ID
Reason for absence
count
150
100
50
0
1 10 11 12 13 14 15 16 17 18 19 2 20 21 22 23 24 25 26 27 28 3 4 5 6 7 8 9
Reason.for.absence
Activate Windows
Go to Settings to activate Windows.


Month
count
75
50
25
0
1
10
11
12
2
3
4
5
6
7
8
9
Month.of.absence

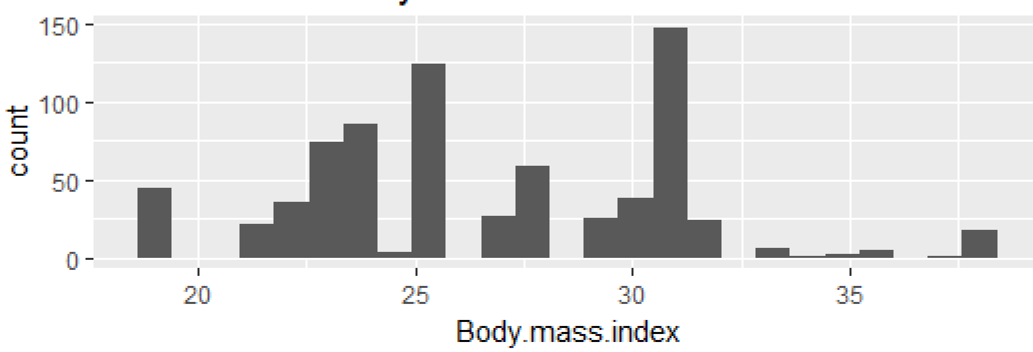

Disciplinary failure
count
600
400
200
0
0
1
Disciplinary.failure
Activate Windows
Go to Settings to activate Windows.


Education
count
600
400
200
0
1
2
3
4
Education
Son
count
300
200
100
0
0
1
2
3
4
Son
Social smoker
count
600
400
200
0
0
1
Social.smoker
Activate Windows
Go to Settings to activate Windows.

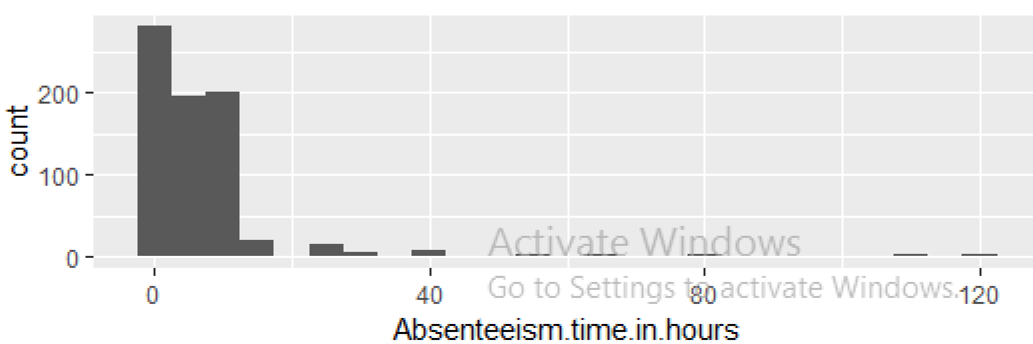
Distribution of: Height

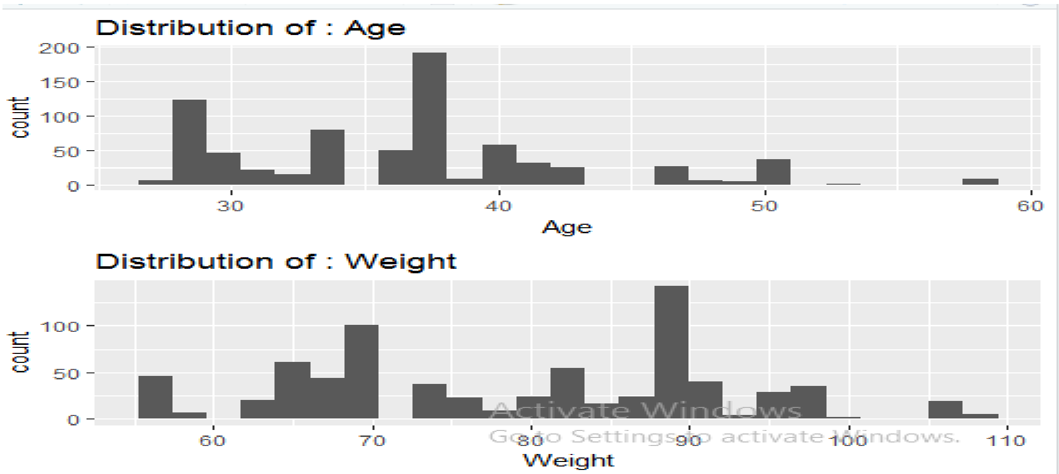


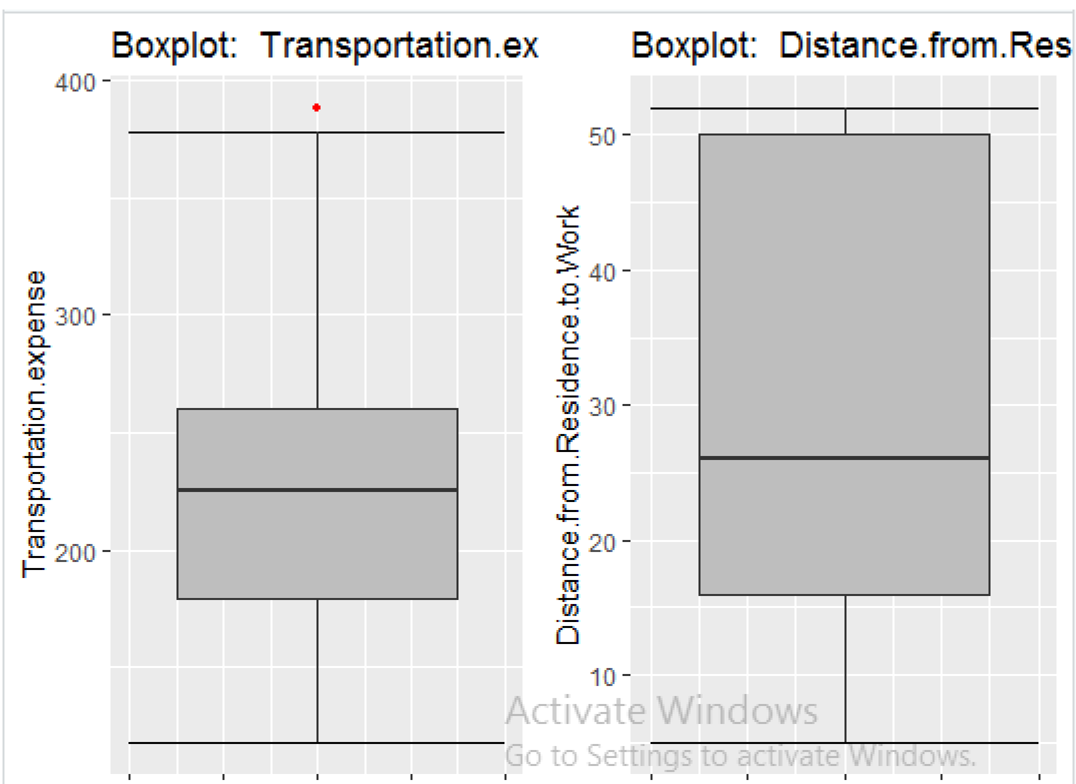
Distribution of: Body mass index

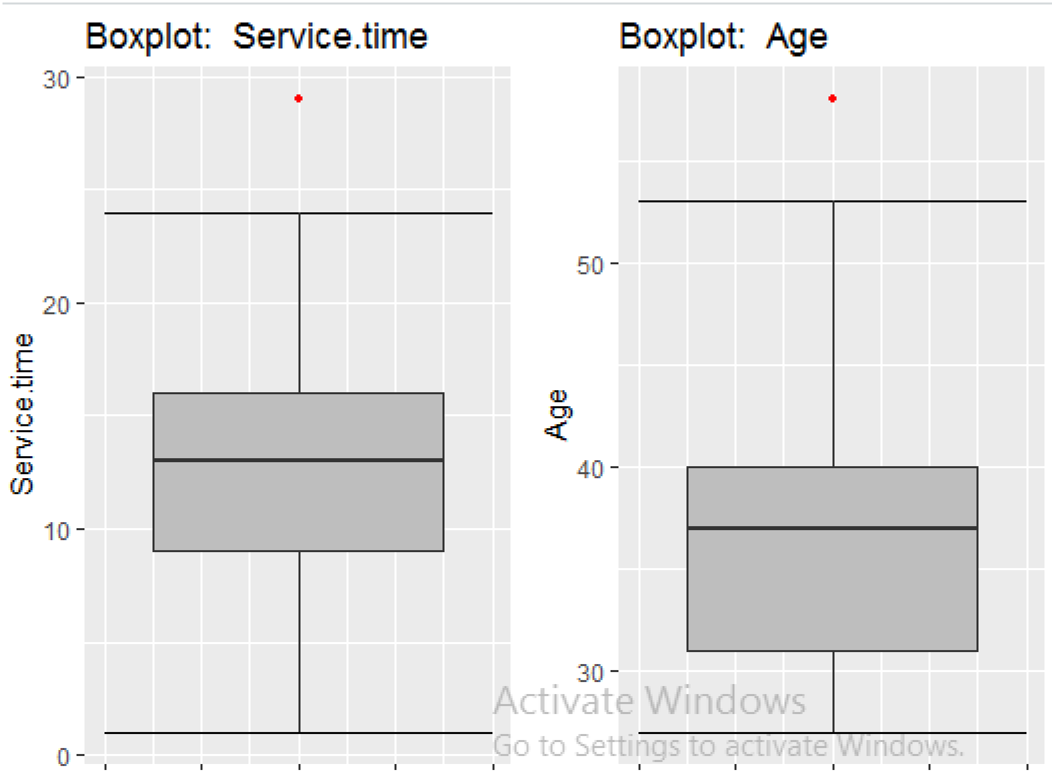


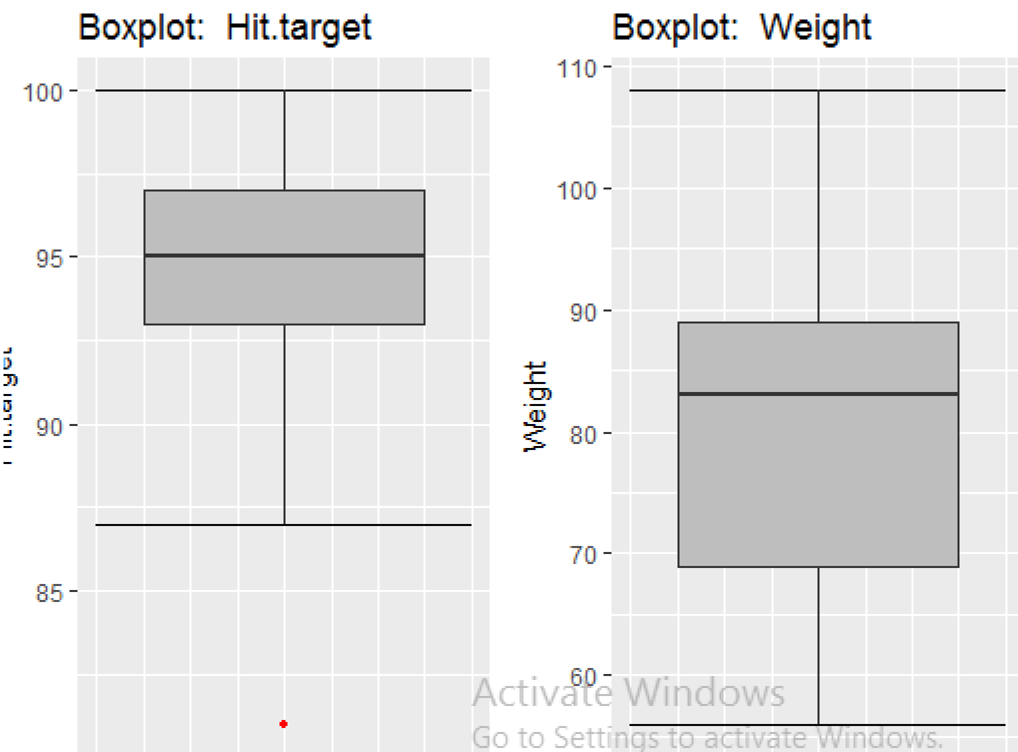
Distribution of : Absenteeism time in hours

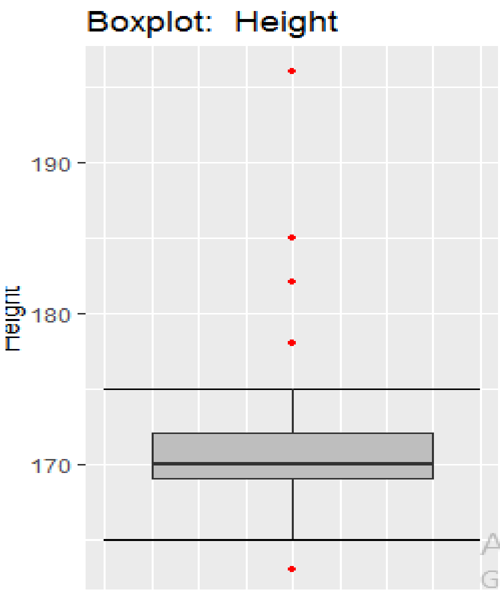


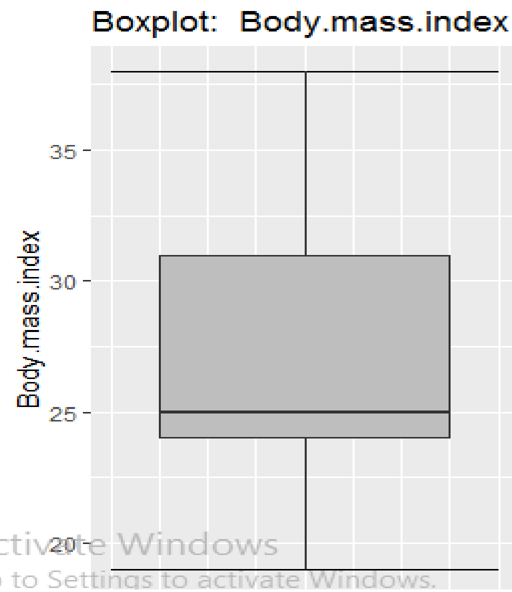

Distribution of : Age
count
200
150
100
50
0
30
40
50
60
Age
Distribution of : Weight
count
100
50
0
60
70
80
90
100
110
Weight
Activate Windows
Go to Settings to activate Windows.

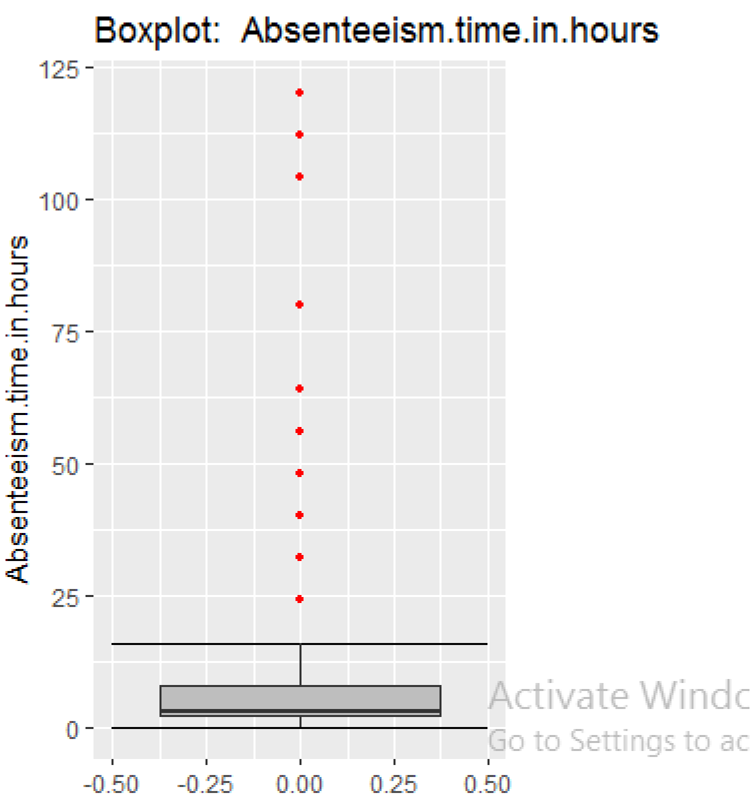

Boxplot: Transportation.ex
Transportation.expense
400
300
200
Boxplot: Distance.from.Res
Distance.from.Residence.to.Work
50
40
30
20
10
Activate Windows
Go to Settings to activate Windows.


Boxplot: Service.time
Service.time
0
10
20
30
Boxplot: Age
Age
30
40
50


Boxplot: Hit.target
85
90
95
100
Boxplot: Weight
Weight
60
70
80
90
100
110


Boxplot: Height
Height
190
180
170


Boxplot: Body.mass.index
Body.mass.index
35
30
25
20
Activate Windows
Go to Settings to activate Windows.


Boxplot: Absenteeism.time.in.hours
Absenteeism.time.in.hours
125
100
75
50
25
0
-0.50
-0.25
0.00
0.25
0.50
Activate Windc
Go to Settings to ac

5.2>R Code

```
#Clean the environment
rm(list = ls())

#Seting the  working directory
setwd("D:/Rfiles")

#Loading the librarires which would be needed
libraries = c("rpart.plot","plyr","dplyr",
"ggplot2","rpart","DMwR","randomForest","usdm","DataCombine")
lapply(X = libraries,require, character.only = TRUE)
rm(libraries)

#Read the csv file
absent = read.csv(file = "Absent.csv", header = T)

#####EXPLORE THE DATA

#number of rows and columns
dim(absent)

#Observe top 5 rows
head(absent)

#Structure of variables
str(absent)

#Transform data types
absent$ID = as.factor(as.character(absent$ID))
##as reason cant be zero
absent$Reason.for.absence[absent$Reason.for.absence %in% 0] = 20
absent$Reason.for.absence = as.factor(as.character(absent$Reason.for.absence))
```

```
#as month cant be zero
absent$Month.of.absence[absent$Month.of.absence %in% 0] = NA
absent$Month.of.absence = as.factor(as.character(absent$Month.of.absence))
absent$Day.of.the.week = as.factor(as.character(absent$Day.of.the.week))
absent$Seasons = as.factor(as.character(absent$Seasons))
absent$Disciplinary.failure = as.factor(as.character(absent$Disciplinary.failure))
absent$Education = as.factor(as.character(absent$Education))
absent$Son = as.factor(as.character(absent$Son))
absent$Social.drinker = as.factor(as.character(absent$Social.drinker))
absent$Social.smoker = as.factor(as.character(absent$Social.smoker))
absent$Pet = as.factor(as.character(absent$Pet))

#Make a copy of data
df = absent
######MISSING VALUE ANALYSIS
#Get number of missing values(use of sapply)
sapply(df,function(x){sum(is.na(x))})
missing_values = data.frame(sapply(df,function(x){sum(is.na(x))}))

#Get the rownames as new column
missing_values$Variables = row.names(missing_values)

#Reseting the row names
row.names(missing_values) = NULL

#Renaming the column
names(missing_values)[1] = "Miss_perc"

#Calculate missing percentage(so that we can see the highest percentage)
missing_values$Miss_perc = ((missing_values$Miss_perc/nrow(absent)) *100)

#Reorder the columns
missing_values = missing_values[,c(2,1)]
```

```
#Sort the rows according to decreasing missing percentage
missing_values = missing_values[order(-missing_values$Miss_perc),]

# we cannot drop any column because there isn't any column having missing value
greater than 30%
#Create missing value and impute using mean, median and knn
df1 =df
df2= df
df3= df


# we are using Body mass index becase it has max missing value ,

 df1[["Body.mass.index"]][10]
 df2[["Body.mass.index"]][10]
 df3[["Body.mass.index"]][10]


 # here the value we have choosen to remove is 29

 df1[["Body.mass.index"]][10] =NA
 df2[["Body.mass.index"]][10]= NA
 df3[["Body.mass.index"]][10] = NA

 # checking for different values

 #mean

 df1[["Body.mass.index"]][10] = mean(df1$Body.mass.index, na.rm = T)
 df1[["Body.mass.index"]][10] #the value we got is 26.68
```

```
 #median


 df2[["Body.mass.index"]][10] = median(df2$Body.mass.index, na.rm = T)
 df2[["Body.mass.index"]][10] #the value we got is 25


 #KNN
 df3 = knnImputation(data = df3, k = 5)
 df3[["Body.mass.index"]][10]  # we got the value 29.17163  , so we going ahead
with KNN



# implementing KNN on original dataframe
df = knnImputation(data = df, k = 5)

#Check if any missing values
sum(is.na(df))





######GRAPHS
# numerical data
numeric_index = sapply(df, is.numeric)
numeric_data = df[,numeric_index]

#Distribution of factor data using bar plot
bar1 = ggplot(data = df, aes(x = ID)) + geom_bar() + ggtitle("ID") + theme_dark()
bar2 = ggplot(data = df, aes(x = Reason.for.absence)) + geom_bar() +
ggtitle("Reason for absence") + theme_dark()
```

```
bar3 = ggplot(data = df, aes(x = Month.of.absence)) + geom_bar() +
ggtitle("Month") + theme_dark()
bar4 = ggplot(data = df, aes(x = Disciplinary.failure)) + geom_bar() +
ggtitle("Disciplinary failure") + theme_dark()
bar5 = ggplot(data = df, aes(x = Education)) + geom_bar() + ggtitle("Education") +
theme_dark()
bar6 = ggplot(data = df, aes(x = Son)) + geom_bar() + ggtitle("Son") + theme_dark()
bar7 = ggplot(data = df, aes(x = Social.smoker)) + geom_bar() + ggtitle("Social
smoker") + theme_dark()

#making a grid
gridExtra::grid.arrange(bar1,bar2,ncol=1)
gridExtra::grid.arrange(bar3,bar4,ncol=1)
gridExtra::grid.arrange(bar5,bar6,bar7,ncol=2)

#Check the distribution of numerical data using histogram
hist1 = ggplot(data = numeric_data, aes(x =Transportation.expense)) +
ggtitle("Distribution of : Transportation.expense") + geom_histogram(bins = 25)
hist2 = ggplot(data = numeric_data, aes(x =Height)) + ggtitle("Distribution of:
Height") + geom_histogram(bins = 25)
hist3 = ggplot(data = numeric_data, aes(x =Body.mass.index)) +
ggtitle("Distribution of: Body mass index") + geom_histogram(bins = 25)
hist4 = ggplot(data = numeric_data, aes(x =Absenteeism.time.in.hours)) +
ggtitle("Distribution of : Absenteeism time in hours") + geom_histogram(bins = 25)
hist5 = ggplot(data = numeric_data, aes(x =Age)) + ggtitle("Distribution of : Age") +
geom_histogram(bins = 25)
hist6 = ggplot(data = numeric_data, aes(x =Weight)) + ggtitle("Distribution of :
Weight") + geom_histogram(bins = 25)

#making a grid
gridExtra::grid.arrange(hist1,hist2,ncol=1)
gridExtra::grid.arrange(hist3,hist4,ncol=1)
gridExtra::grid.arrange(hist5,hist6,ncol=1)
```

```
#####OUTLIER ANALYSIS
# here we will replace the outliers with Knn method.
#Get the data with only numeric columns
numeric_index = sapply(df, is.numeric)
numeric_data = df[,numeric_index]
#Get the data with only factor columns
factor_data = df[,!numeric_index]

#Check for outliers using boxplots
for(i in 1:ncol(numeric_data)) {
  assign(paste0("box",i), ggplot(data = df, aes_string(y = numeric_data[,i]))
         +stat_boxplot(geom = "errorbar", width = 1)
         +geom_boxplot(outlier.colour = "red", fill = "grey", outlier.size = 1)
         +labs(y = colnames(numeric_data[i]))
         +ggtitle(paste("Boxplot: ",colnames(numeric_data[i]))))
}

#Arrange the plots in grids
gridExtra::grid.arrange(box1,box2,ncol=2)
gridExtra::grid.arrange(box3,box4,ncol=2)
gridExtra::grid.arrange(box3,box4,ncol=2)
gridExtra::grid.arrange(box5,box6,ncol=2)
gridExtra::grid.arrange(box7,box8,ncol=2)
gridExtra::grid.arrange(box9,ncol=2)

#Get the names of numeric columns
numeric_columns = colnames(numeric_data)

#Replacing all outlier data with NA
for(i in numeric_columns){
  val = df[,i][df[,i] %in% boxplot.stats(df[,i])$out]
  print(paste(i,length(val)))
```

```
  df[,i][df[,i] %in% val] = NA
}

#Check number of missing values
sapply(df,function(x){sum(is.na(x))})

#Get number of missing values after replacing outliers as NA
missing_values_out = data.frame(sapply(df,function(x){sum(is.na(x))}))
missing_values_out$Columns = row.names(missing_values_out)
row.names(missing_values_out) = NULL
names(missing_values_out)[1] = "Miss_perc"
missing_values_out$Miss_perc = ((missing_values_out$Miss_perc/nrow(absent))
*100)
missing_values_out = missing_values_out[,c(2,1)]
missing_values_out = missing_values_out[order(-missing_values_out$Miss_perc),]
missing_values_out

#Compute the NA values using KNN imputation
df = knnImputation(df, k = 5)

#Check if any missing values
sum(is.na(df))







#######FEATURE SELECTION

#usdm library  # done for the checking correlation
#Check for multicollinearity
```

```
vifcor(numeric_data)

#Check for multicollinearity using corelation graph
corrdf = cor(numeric_data,numeric_data, method = "spearman" )

#As body mass index has high correlation with weight , so we are dropping body mass index
# and 'distance' because it has very low correaltion with target variable
to_drop <- c("Body.mass.index","Distance.from.Residence.to.Work")
df1=df[ ,  -which(names(df) %in% to_drop)]




######FEATURE SCALING

#hist(df$Absenteeism.time.in.hours)
df = df1
#Remove dependent variable
numeric_index = sapply(df,is.numeric)
numeric_data = df[,numeric_index]
numeric_columns = names(numeric_data)
numeric_columns = numeric_columns[-9]



#Normalization of continuous variables
for(i in numeric_columns){
  print(i)
  df[,i] = (df[,i] - min(df[,i]))/
    (max(df[,i]) - min(df[,i]))
}
```

```
#Get the names of factor variables
factor_columns = names(factor_data)


########DECISION TREE


#Splitting the  data (80-20 percent)
set.seed(1)
train_index = sample(1:nrow(df), 0.8*nrow(df))
train = df[train_index,]
test = df[-train_index,]


#Build decsion tree using rpart
dt_model = rpart(Absenteeism.time.in.hours ~ ., data = train, method = "anova")
# here we can try any method other than anova ,
#one of "anova", "poisson", "class" or "exp".
#If method is missing then the routine tries to make an intelligent guess.

#Ploting the tree
rpart.plot(dt_model)


#Perdict for test cases
dt_predictions = predict(dt_model, test[,-19])

df3= data.frame((dt_predictions))

#Create data frame for actual and predicted values
df_pred = data.frame("actual"= test[,19], "dt_pred"=dt_predictions)
```

```
# calculating error (mae)
mae_dt = regr.eval(test[,19], dt_predictions , stats = c ('mae'))
print(mae_dt)
print("accuracy ")
print((1-(mae_dt))*100)


#########RANDOM FOREST


#Training the model using training data
rf_model = randomForest(Absenteeism.time.in.hours~., data = train, ntree = 500)

#Predict the test cases
rf_predictions = predict(rf_model, test[,-19])

#Create dataframe for actual and predicted values
df_pred = cbind(df_pred,rf_predictions)


# calculating error (mae)
mae_rf = regr.eval(test[,19], rf_predictions , stats = c ('mae'))
print(mae_rf)
print("accuracy ")
print((1-(mae_rf))*100)




#########LINEAR REGRESSION

train_for_lr = train
```

```
test_for_lr = test

train_for_lr$ID = NULL
test_for_lr$ID = NULL
train_for_lr$Reason.for.absence = NULL
test_for_lr$Reason.for.absence = NULL

#we are removing these two columns because they are creating new levels
##Training the model using training data
lr_model = lm(formula = Absenteeism.time.in.hours~., data = train_for_lr)
# summary of lr_model
summary(lr_model)


#Predict the test cases
lr_predictions = predict(lr_model, test_for_lr[,-19])

#Creating a new dataframe for actual and predicted values
df_pred = cbind(df_pred,lr_predictions)


# calculating error (mae)
mae_lr = regr.eval(test[,19], lr_predictions , stats = c ('mae'))
print(mae_lr)
print("accuracy ")
print((1-(mae_lr))*100)
```

5.3 >> References

https://stackoverflow.com
https://towardsdatascience.com/