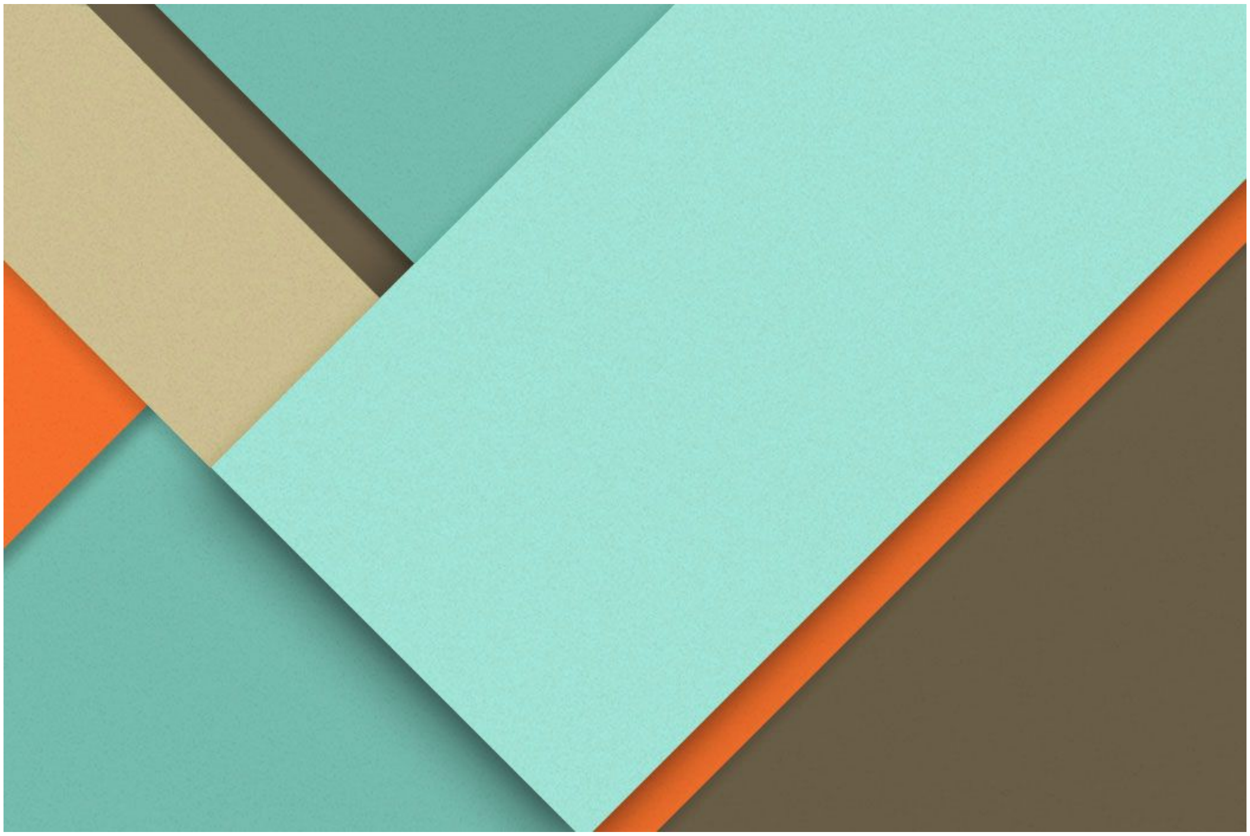


Employee Absenteeism

Project Report

06.02.2019

By Neha

Table of Content

Chapter 1 - Introduction

1.1 Project Description

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.

1.2 Problem Statement

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?

1.3 Data

Dataset Details:

Dataset Characteristics: Time Series Multivariate

Number of Attributes: 21

Missing Values : Yes

Attribute 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 (kilometers)

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)

Following is the glimpse of the actual data:

Table 1.1: Employee Absenteeism sample Data (Columns 1-6)

ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense
11	26	7	3	1	289
36	0	7	3	1	118
3	23	7	4	1	179
7	7	7	5	1	279
11	23	7	5	1	289

Table 1.2: Employee Absenteeism sample Data (Columns 7-12)

Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target	Disciplinary failure
36	13	33	239554	97	0
13	18	58	239554	97	1
51	18	38	239554	97	0
5	14	39	239554	97	0
36	13	33	239554	97	0

Table 1.3: Employee Absenteeism sample Data (Columns 13-18)

Education	Son	Social drinker	Social smoker	Pet	Weight
1	2	1	0	1	90
1	1	1	0	0	98
1	0	1	0	0	89
1	2	1	1	0	68
1	2	1	0	1	90

Table 1.4: Employee Absenteeism sample Data (Columns 19-21)

Height	Body mass index	Absenteeism time in hours
172	30	4
178	30	0
170	31	2
168	24	4
172	30	2

Chapter 2 - Methodology

2.1 Pre Processing

Any predictive modeling requires that we look at the data before we start modeling.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. However, in data mining terms looking at data refers to so much more than just looking. Looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is often called as Exploratory Data Analysis. For our data we apply preprocessing techniques that we necessary.

We can start looking by checking data types of imported data and then analysing it to check whether the data given is as per standards mentioned in the problem statement. After checking this, we see that all the variable are of continuous or numeric type. And some of the values in dataset are not following data standards mentioned in problem statement. For example, variables reason for absence and month of absence contains value of 0 for some observations, this can't be the case as it is clearly mentioned that reasons for absence are categorized from 1 to 28 types and from data it can be observed that month has values from 1 to 12. So we must change this values to proper type (only exception is where the target variable Absenteeism time in hours is also 0). Moreover, all the variable are of numeric type but when we check the actual unique values and range of the variable, it can be seen that many can be converted to factors like Reason for absence, Day of the week, Son, Social drinker, Education, Disciplinary failure etc. Check the code A.1. After making this exploratory data analysis changes, we can move ahead with data preprocessing techniques.

Summary of data is given below to know variables types and dimension of data.

```
> str(data)
'data.frame':	740 obs. of  21 variables:
 $ ID                             : num  11 36 3 7 11 3 10 20 14 1 ...
 $ Reason.for.absence             : num  26 0 23 7 23 23 22 23 19 22 ...
 $ Month.of.absence               : num  7 7 7 7 7 7 7 7 7 7 ...
 $ Day.of.the.week                : num  3 3 4 5 5 6 6 6 2 2 ...
 $ Seasons                        : num  1 1 1 1 1 1 1 1 1 1 ...
 $ Transportation.expense         : num  289 118 179 279 289 179 NA 260 155 235 ...
 $ Distance.from.Residence.to.Work: num  36 13 51 5 36 51 52 50 12 11 ...
 $ Service.time                   : num  13 18 18 14 13 18 3 11 14 14 ...
 $ Age                            : num  33 50 38 39 33 38 28 36 34 37 ...
 $ Work.load.Average.day.         : num  239554 239554 239554 239554 239554 ...
 $ Hit.target                     : num  97 97 97 97 97 97 97 97 97 97 ...
 $ Disciplinary.failure           : num  0 1 0 0 0 0 0 0 0 0 ...
 $ Education                      : num  1 1 1 1 1 1 1 1 1 3 ...
 $ Son                            : num  2 1 0 2 2 0 1 4 2 1 ...
 $ Social.drinker                 : num  1 1 1 1 1 1 1 1 1 0 ...
 $ Social.smoker                  : num  0 0 0 1 0 0 0 0 0 0 ...
 $ Pet                            : num  1 0 0 0 1 0 4 0 0 1 ...
 $ Weight                         : num  90 98 89 68 90 89 80 65 95 88 ...
 $ Height                         : num  172 178 170 168 172 170 172 168 196 172 ...
 $ Body.mass.index                : num  30 31 31 24 30 31 27 23 25 29 ...
 $ Absenteeism.time.in.hours      : num  4 0 2 4 2 NA 8 4 40 8 ...
```

Figure 2.1 Summary of data

```
ID                                 740 non-null int64
Reason for absence                 729 non-null float64
Month of absence                   736 non-null object
Day of the week                    740 non-null object
Seasons                            740 non-null object
Transportation expense             733 non-null float64
Distance from Residence to Work    737 non-null float64
Service time                       737 non-null float64
Age                                737 non-null float64
Work load Average/day              730 non-null float64
Hit target                         734 non-null float64
Disciplinary failure               734 non-null float64
Education                          730 non-null object
Son                                734 non-null object
Social drinker                     737 non-null object
Social smoker                      736 non-null object
Pet                                738 non-null object
Weight                             739 non-null float64
Height                             726 non-null float64
Body mass index                    709 non-null float64
Absenteeism time in hours          718 non-null float64
dtypes: float64(12), int64(1), object(8)
```

Figure 2.2 Data after conversion

2.1.1 Missing Value Analysis

Missing values in any variable can adversely affect the accuracy of model and hamper the prediction result. So treating missing values before model development is very important. As our data contains missing values, we must do missing value analysis for data. First, check percentage of missing values for each variable. If missing value percentage is greater than 30%, we have to drop that column from model development. By doing this, we may lose precious information but even after imputing missing values for this variable, it will be biased because we have imputed it manually. But it is not the case for our dataset, hence will go ahead and impute it.Of the 21 variables provides 18 variables had the missing values.We first checked which method to implement by using each method at a time.Out of mean, median and knn method, the accurate result was given by knn imputation. Hence we imputed the missing values using KNN method.

	Variable	Missing_Percentage
0	Body mass index	4.189189
1	Absenteeism time in hours	2.972973
2	Height	1.891892
3	Reason for absence	1.486486
4	Work load Average/day	1.351351
5	Education	1.351351
6	Transportation expense	0.945946
7	Son	0.810811
8	Disciplinary failure	0.810811
9	Hit target	0.810811
10	Social smoker	0.540541
11	Month of absence	0.540541
12	Age	0.405405
13	Service time	0.405405
14	Distance from Residence to Work	0.405405
15	Social drinker	0.405405
16	Pet	0.270270
17	Weight	0.135135
18	Seasons	0.000000
19	Day of the week	0.000000
20	ID	0.000000

Figure 2.3 Missing Values

2.1.2 Outlier Analysis

The outliers are the values of variables which fall beyond the normal range of the variable values and considered as exception. So it is better to remove them to make data normally distributed. But it is not the case always, sometimes outliers are telling something about the target variable. So we must check this before processing of the outliers. Now in our case, some of the variables are containing outliers. The figure 2.1.1 shows the boxplots of all the numeric variables. But we will not process the outliers of variable depending on ID to preserve the data integrity. But others can be processed using boxplot method.

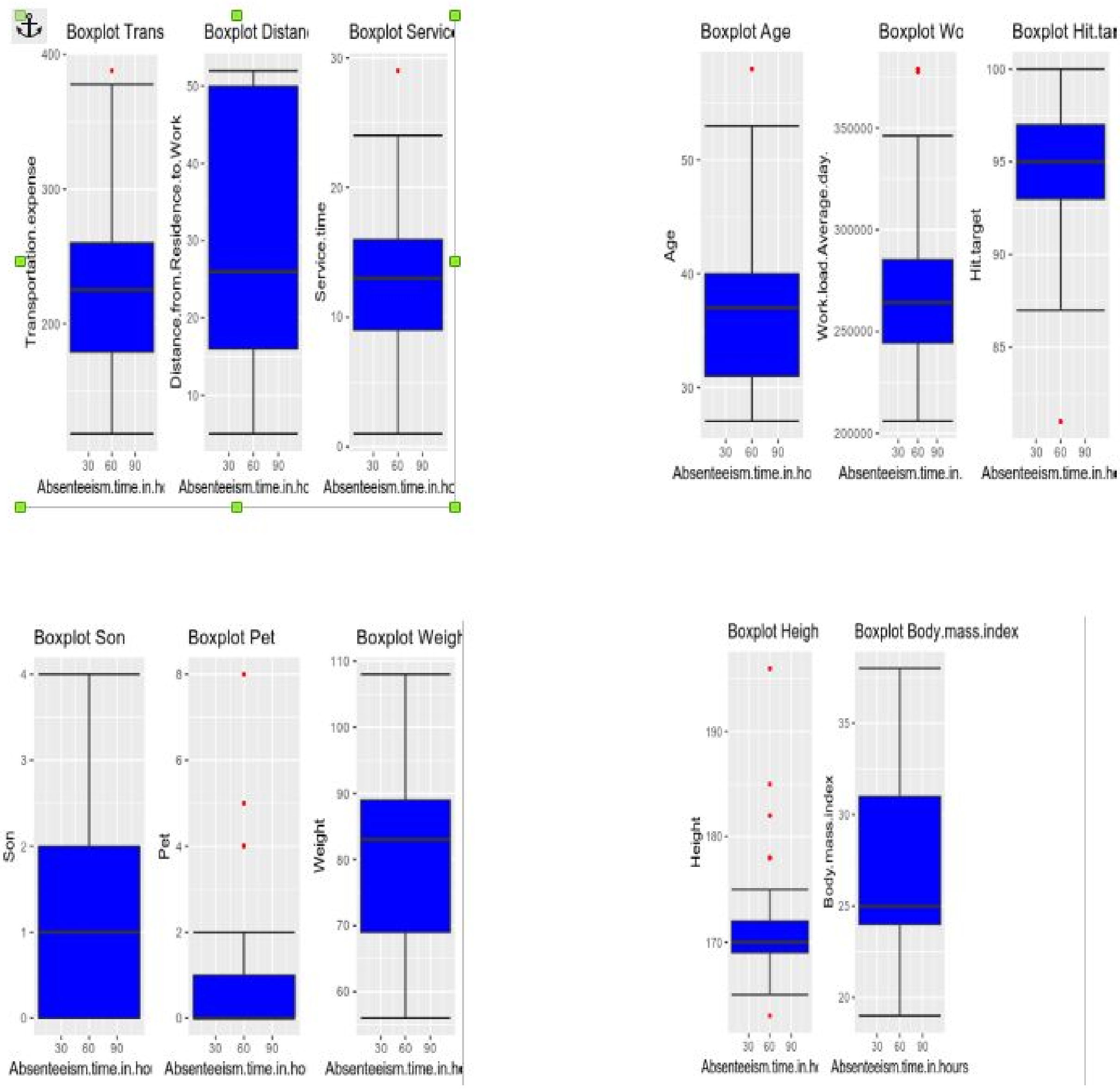


Figure 2.4 Boxplot of Variables with Outliers

2.1.3 Feature Selection

Feature Selection is one of the core concepts in machine learning which hugely impacts the performance of our model.Machine learning works on a simple rule – if we put garbage in, we will only get garbage to come out. By garbage here, I mean noise in data.The data features that you use to train your machine learning models have a huge influence on the performance you can achieve.Feature Selection is the process where we automatically or manually select those features which contribute most to our prediction variable or output in which we are interested in. Having irrelevant features in our data can decrease the accuracy of the models and make our model learn based on irrelevant features.There are several methods of doing that. We have used the correlation analysis to check collinearity between the variables and anova test to check dependence of target variable on the independent variables.

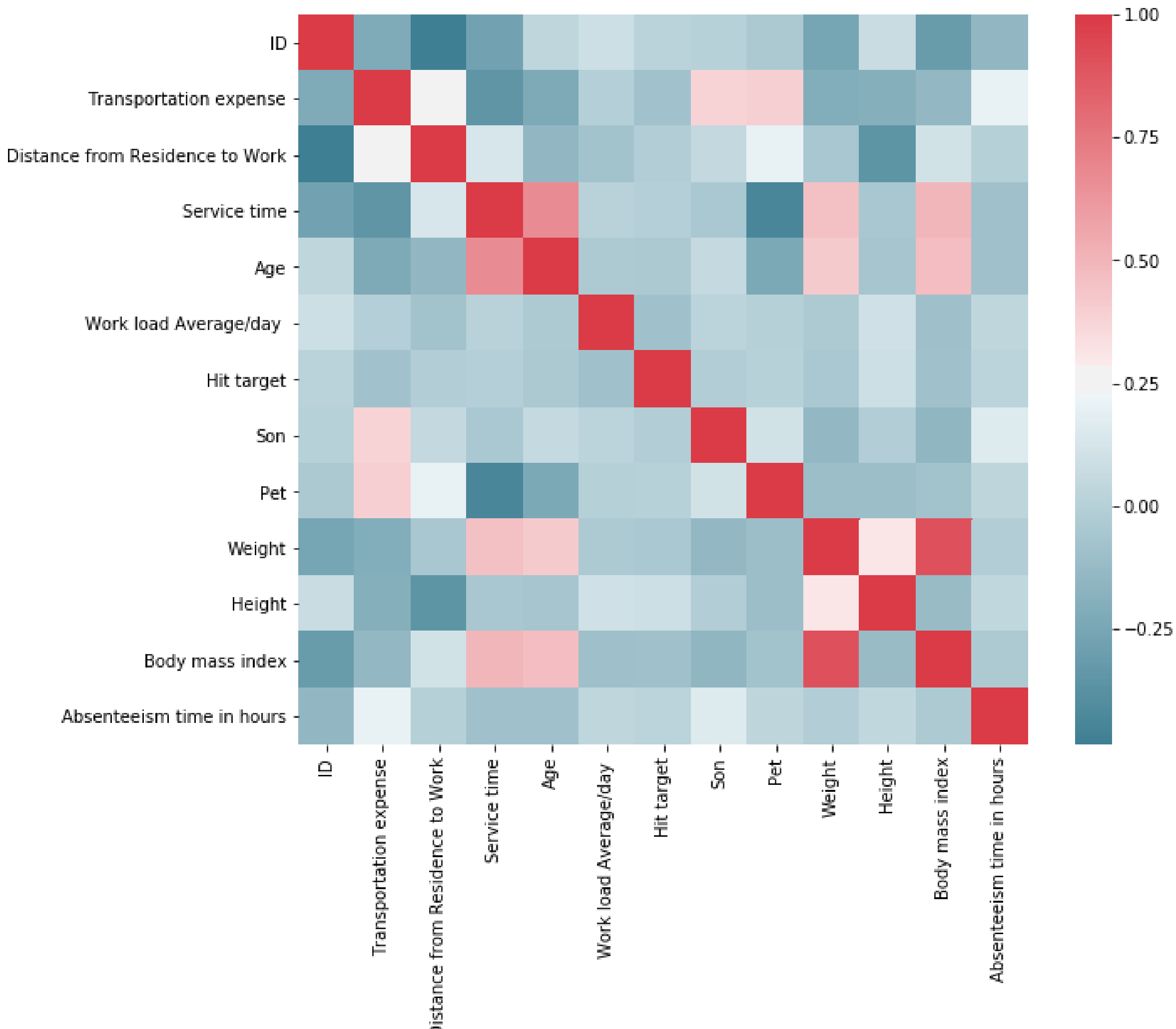


Figure 2.5 Correlation Plot for Employee Absenteeism Data

Analysis of variance (ANOVA) is a statistical technique that is used to check if the means of two or more groups are significantly different from each other. ANOVA checks the impact of one or more factors by comparing the means of different samples. As our target variable is numerical we will use ANOVA for feature selection technique to see whether any categorical variable is related to target variable. The result of anova is as follows.

```
> summary(anova_test)
                      Df Sum Sq Mean Sq F value   Pr(>F)    
ID                    35  12948   369.9   2.553 3.71e-06 ***
Day.of.the.week        4   1888   471.9   3.256   0.0117 *  
Education              1     57    56.9   0.393   0.5311    
Social.drinker         1     93    93.0   0.642   0.4234    
Reason.for.absence    27  22712   841.2   5.804  < 2e-16 ***
Seasons                3     73    24.4   0.168   0.9177    
Month.of.absence      12   1610   134.1   0.926   0.5206    
Disciplinary.failure   1      1     1.5   0.010   0.9193    
Residuals            655  94927   144.9                     
```

Figure 2.6 Summary of ANOVA test

H o = Categorical variable is Independent from the Target variable

H a = Categorical variable is Dependent on the Target variable

If the p value of the categorical variable is less than 0.05 then we will consider that the target variable is dependent on the categorical variable for which we reject the null hypothesis. From the above result we can see that only four variables are very much related to target variable hence we delete all the other variables.

Therefore from both the correlation analysis and ANOVA we got some variable which we shouldn't consider for further processing.

Therefore, following continuous variables can be removed after correlation

analysis :

1. Numeric Variables:
    a. Weight
2. Categorical Variables:
    a. Education
    b. Social.drinker
    c. Seasons
    d. Month.of.absence
    e. Disciplinary.failure

2.1.4 Feature Scaling

Some variables range of hundreds while other have range of thousands. We need to normalise this, so that model should not be more prone towards the high value variables. We can do this either by standardization or normalization. Feature scaling method limits the range of variable so they can perform on a common method. Standardization is more suited for the data which is normally distributed. As from Figure 2.6 we can see our Employee Absenteeism data is not normally distributed. So we can't use standardization technique, Normalization is more suitable for such data set. After normalization all numeric data values will be between 0 and 1.

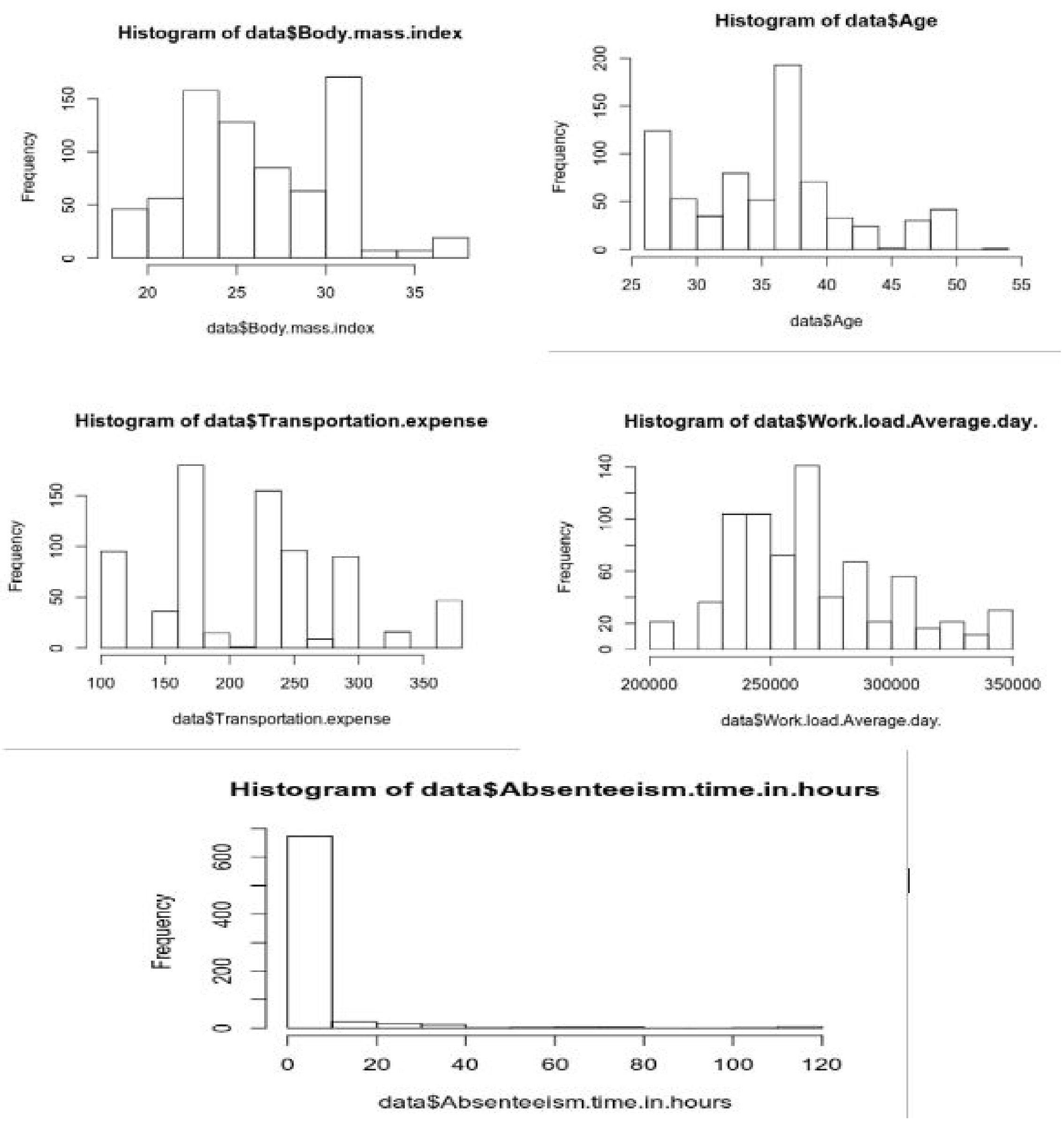


Figure 2.7 Histogram of Continuous Variables

2.2 Modeling

2.2.1 Model Selection

After preprocessing of data, we must proceed with model development. For Employee Absenteeism Project, we want to find what changes company should make to reduce the Absenteeism problem and also what are the expected loss in the year 2011 per month if same trend continues. So we need find the importance of each variable with respect to target variable to suggest the changes for company and predict the result of next year for same data to calculate the losses of company due to absenteeism. For doing this, we can use following regression models:

1. Decision Tree Regression
2. Random Forest Regression
3. Multiple Linear Regression

2.2.2 Decision Tree Regression

Decision tree is a predictive model based on a branching series of boolean tests. It is a rule. Each branch connects nodes with "and" and multiple branches are connected by "or". It can be used for classification and regression. It is a supervised machine learning algorithm. Accept continuous and categorical variables as independent variables. Extremely easy to understand by the business users. As with implementation of Decision Tree for regression, we get the importance for each variable for predicting target variable.

```
Variable importance
                           ID          Reason.for.absence                    Age  Transportation.expense
                           32                          25                     14                       7
Distance.from.Residence.to.Work              Service.time  Work.load.Average.day.         Body.mass.index
                            5                           5                      4                       3
                   Hit.target                         Pet         Day.of.the.week
                            2                           1                      1
```

Figure 2.8 Decision Trees Summary

2.2.3 Random Forest Regression

Random Forest is a supervised learning algorithm. Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. The idea behind the random forest is to build 'n' number of tree to have more accuracy on the data set. Forest because we build 'n' number of decision tree, random because we chose variables randomly. It is the powerful machine learning algorithm and reduces misclassification error. Random Forest is called 'ensemble' technique because it is a combination of multiple decision tree algorithm. This method combines Breiman's "bagging" idea and random selection of feature. The more tree the more robust random forest will. Here we will be using random forest regressor for our data. Glimpse of first tree of Random Forest Tree:

```
Call:
 randomForest(formula = Absenteeism.time.in.hours ~ ., data = train,      importance = TRUE, ntree = 100)
               Type of random forest: regression
                     Number of trees: 100
No. of variables tried at each split: 4

          Mean of squared residuals: 189.3559
                    % Var explained: 13.89
> getTree(RF_Reg, 1, labelVar = TRUE)
    left daughter right daughter                      split var  split point status  prediction
1               2              3                Day.of.the.week 2.400000e+01     -3   7.8787510
2               4              5                             ID 7.884517e+08     -3   4.3665979
3               6              7 Distance.from.Residence.to.Work 2.872340e-01     -3   9.7671034
4               8              9                            Age 8.653846e-01     -3   3.0166727
5              10             11                             ID 6.012948e+10     -3   8.6053630
6              12             13                   Service.time 6.086957e-01     -3  14.8940411
7              14             15         Transportation.expense 5.230769e-01     -3   7.7163283
8              16             17                            Son 7.500000e-01     -3   3.1122812
9              18             19             Reason.for.absence 3.276800e+04     -3   1.4444444
10             20             21             Reason.for.absence 2.684354e+08     -3   7.4714318
11             22             23                             ID 6.012954e+10     -3  11.2512025
12             24             25                         Height 7.234930e-01     -3  11.3255300
13             26             27                         Height 4.833890e-01     -3  26.4230769
```

Figure 2.9 Random Forest Tree

2.2.4 Multiple Linear Regression

Linear regression can only be performed for continuous target variable. It establishes a relationship between the dependent variable and one or more independent variable. For more than one explanatory variable, the process is called multiple linear regression. In the simple linear regression:

- One variable, denoted x, is regarded as the predictor, explanatory, or independent variable.
- The other variable, denoted y, is regarded as the response, outcome, or dependent variable. The equation expressing this relationship is the line:

$$y = b_o + b_1 x$$

Where bo = intercept, b_1 = coefficient of variable (predictor) x

VIF: The variance inflation factor (VIF) is the ratio of variance in a model with multiple terms, divided by the variance of a model with one term alone. It quantifies the severity of multicollinearity in an ordinary least squares regression analysis. It provides an index that measures how much the variance (the square of the estimate standard deviation) of an estimated regression coefficient is increased because of collinearity. VIF for our data can be seen as follows. If VIF is 0 then we can use that data for linear regression model.

DUMMY VARIABLE: In regression analysis, a dummy variable (also known as an indicator variable, design variable, Boolean indicator, binary variable, or qualitative variable) is one that takes the value 0 or 1 to indicate the absence or presence of some categorical effect that may be expected to shift the outcome. Dummy variables are used as devices to sort data into mutually exclusive categories (such as smoker/non-smoker, etc.). For example, in econometric time series analysis, dummy variables may be used to indicate the occurrence of wars or major strikes. A dummy variable can thus be thought of as a truth value represented as a numerical value 0 or 1.

Following is the summary of the Linear model:

```
Call:
lm(formula = Absenteeism.time.in.hours ~ ., data = train[, !colnames(train) %in%
    c("ID")])

Residuals:
    Min      1Q  Median      3Q     Max
-37.545  -4.245  -1.024   2.186 108.794

Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                      -0.1796     4.2421  -0.042 0.966241
Reason.for.absence1               6.5030     4.0822   1.593 0.111731
Reason.for.absence10              7.4392     3.6709   2.027 0.043191 *
Reason.for.absence11              3.4954     3.6706   0.952 0.341369
Reason.for.absence12             19.7739     5.4564   3.624 0.000317 ***
Reason.for.absence13             13.5971     2.9975   4.536 7.04e-06 ***
Reason.for.absence14              5.5955     3.9547   1.415 0.157666
Reason.for.absence15             -0.1553     9.5738  -0.016 0.987066
Reason.for.absence16             -6.6546     7.9417  -0.838 0.402430
Reason.for.absence17             -1.1016    13.4278  -0.082 0.934649
Reason.for.absence18              9.1316     4.3926   2.079 0.038097 *
Reason.for.absence19             17.4959     3.2970   5.307 1.62e-07 ***
```

```
Reason.for.absence2              22.4497    13.3299   1.684 0.092719 .
Reason.for.absence21              1.6362     5.8204   0.281 0.778724
Reason.for.absence22              3.6914     3.3038   1.117 0.264355
Reason.for.absence23             -0.1194     2.5458  -0.047 0.962619
Reason.for.absence24              1.9895     7.9302   0.251 0.802004
Reason.for.absence25             -1.3032     3.7947  -0.343 0.731411
Reason.for.absence26              3.2012     3.3514   0.955 0.339902
Reason.for.absence27              1.1984     3.0989   0.387 0.699127
Reason.for.absence28             -0.4066     2.6617  -0.153 0.878656
Reason.for.absence3               4.0930    13.3522   0.307 0.759308
Reason.for.absence4               3.8418    13.4753   0.285 0.775677
Reason.for.absence5              -0.8360     9.6261  -0.087 0.930825
Reason.for.absence6              38.9699     6.2832   6.202 1.10e-09 ***
Reason.for.absence7               4.6052     4.5532   1.011 0.312256
Reason.for.absence8               0.3934     5.8124   0.068 0.946061
Reason.for.absence9              48.9229     7.9266   6.172 1.31e-09 ***
Day.of.the.week3                  1.1707     1.7129   0.683 0.494595
Day.of.the.week4                 -0.2470     1.6823  -0.147 0.883349
Day.of.the.week5                 -3.5792     1.8094  -1.978 0.048418 *
Day.of.the.week6                 -1.8798     1.7502  -1.074 0.283289
Transportation.expense            1.7955     2.9799   0.603 0.547077
Distance.from.Residence.to.Work  -4.9554     2.0690  -2.395 0.016951 *
Service.time                      6.6802     4.6754   1.429 0.153630
Age                               1.1435     3.9399   0.290 0.771751
Work.load.Average.day.           -3.9918     2.5128  -1.589 0.112731
Hit.target                        2.1621     2.3713   0.912 0.362280
Son                               6.6428     2.5194   2.637 0.008609 **
Social.smoker1                    1.3877     2.3748   0.584 0.559222
Pet                              -0.5275     1.8258  -0.289 0.772755
Height                            5.4467     3.2478   1.677 0.094104 .
Body.mass.index                  -3.7304     3.4718  -1.074 0.283077
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 12.97 on 549 degrees of freedom
Multiple R-squared:  0.291,	Adjusted R-squared:  0.2367
F-statistic: 5.364 on 42 and 549 DF,  p-value: < 2.2e-16
```

Figure 2.10 Multiple Linear Regression

Chapter 3 - Conclusion

3.1 Model Evaluation

After model development, it is important to check its accuracy. For time series data, the accuracy matrix used are MSE(Mean Square Error) or RMSE(Root Mean Square Error). Following are the results of all the implemented model with RMSE.

Table 1.5 Models with Their Results

SL.	Model Name	Accuracy in %	RMSE in %
1.	Decision Tree Regression	89.02	10.98
2.	Random Forest regression	91.47	8.53
3.	Multiple Linear regression	90.64	9.36

3.2 Model Selection

As it can be clearly seen from the MSE or RMSE result, the Random Forest Regression is performing best for this Employee Absenteeism Dataset. So we can freeze the Random Forest model for the predictions of this problem.

3.3 Solutions

3.3.1 Problem 1

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

Here we will use some visualizations to understand it more clearly.

1. Below is the distribution for reason of absenteeism of employees which shows that non ICDs have higher count especially 23 (medical consultation) and 28 (dental consultation). So the company can organize medical and dental checkups for its employees at certain intervals of time so as to keep a check at loss of absenteeism hours.

FigurF

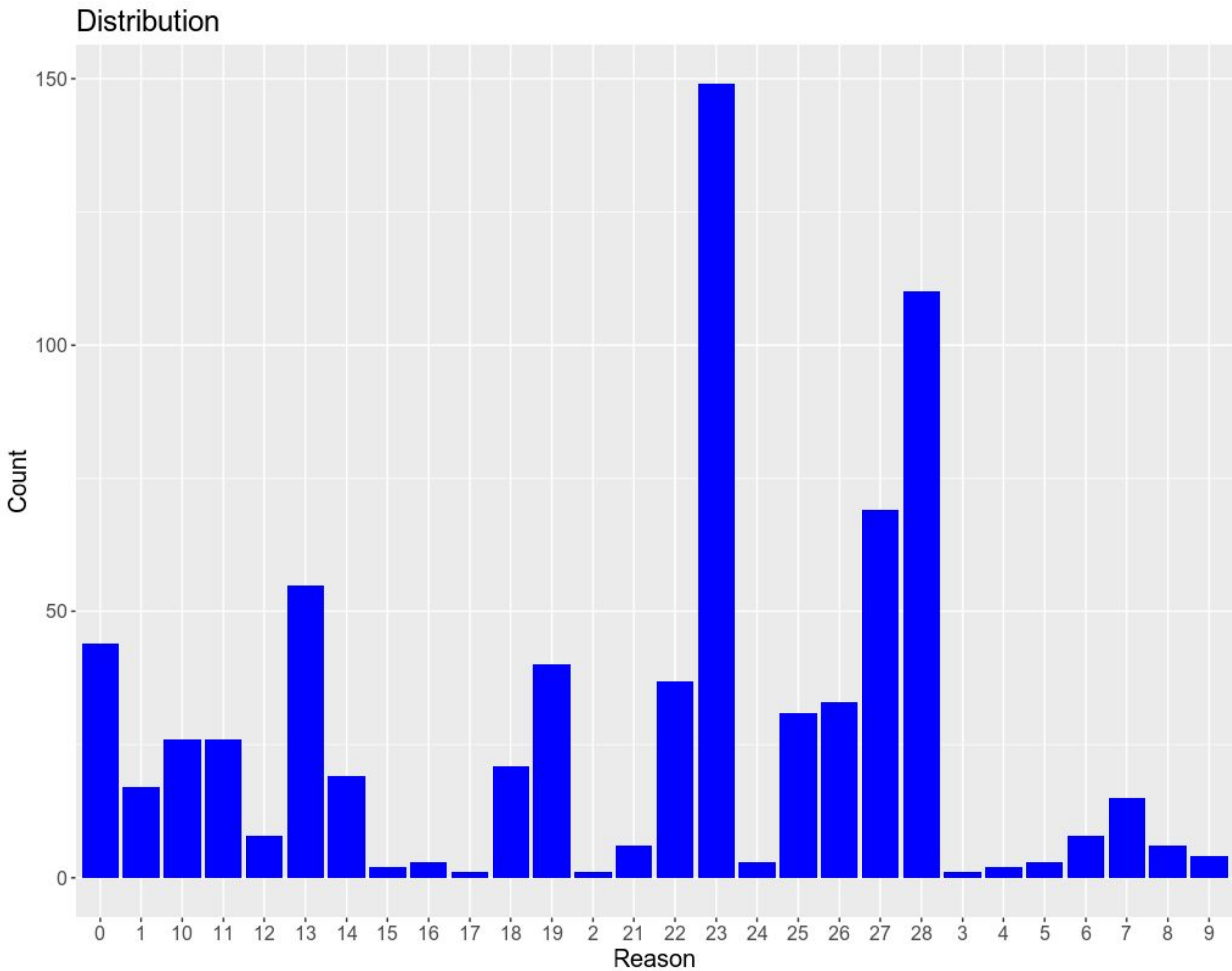
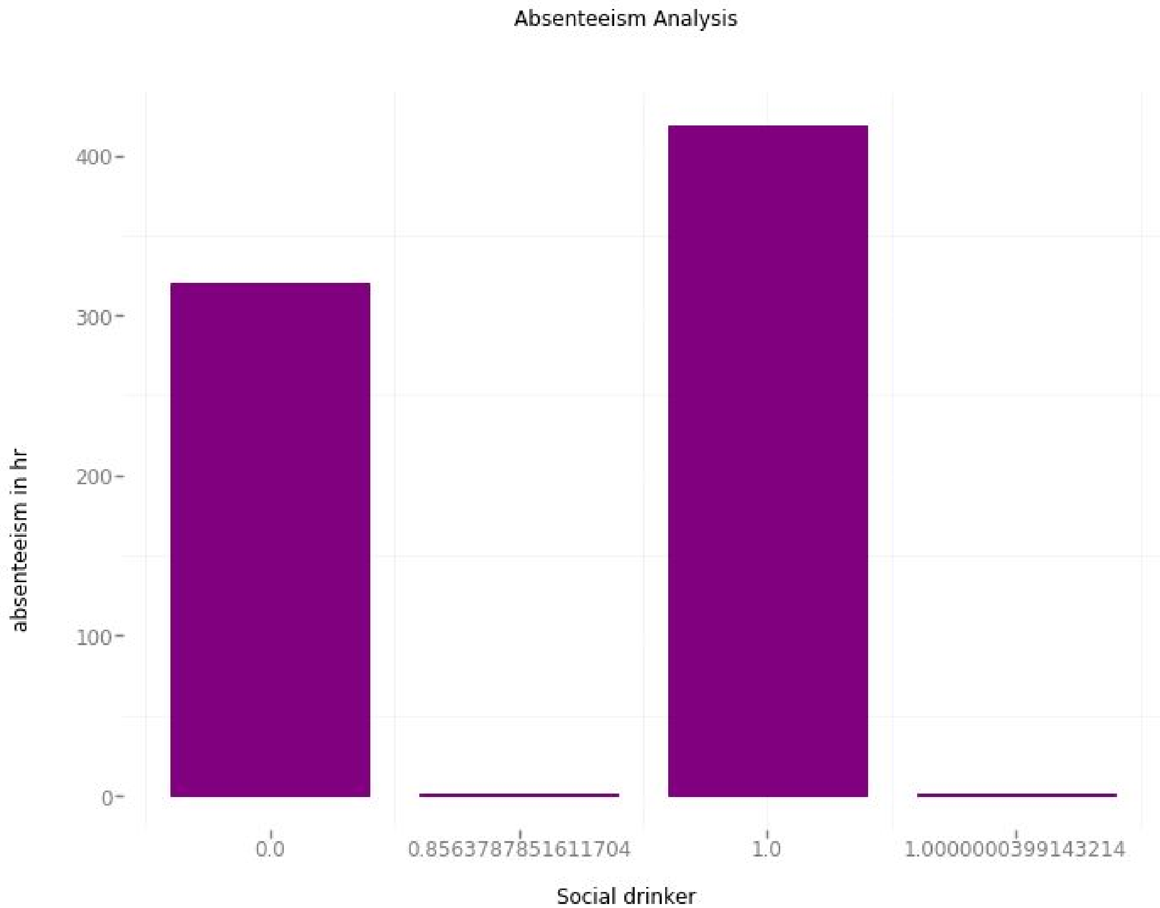


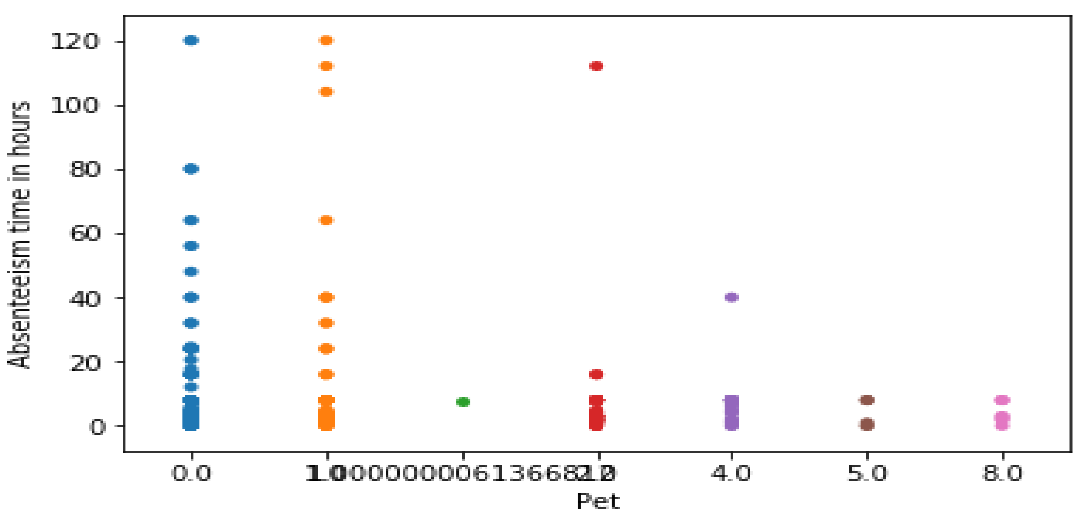
Figure 2

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

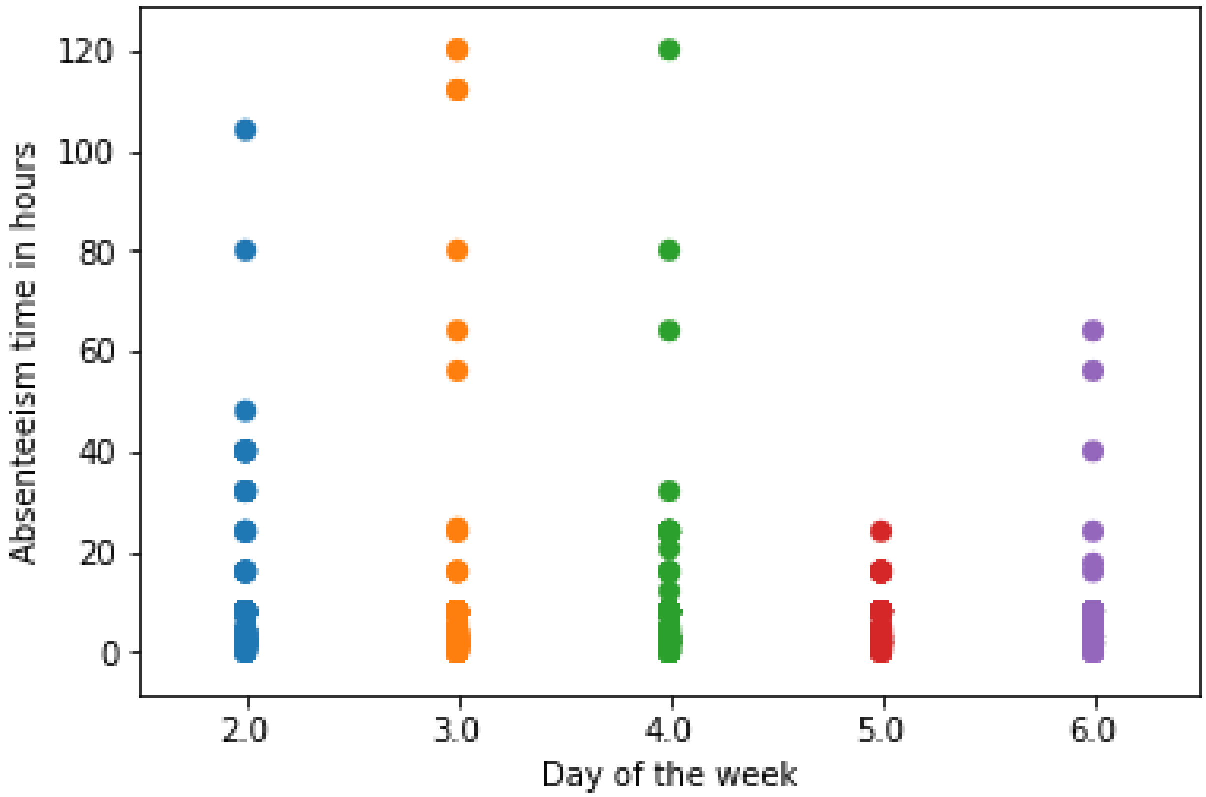
As a drinker is more prone to bad health condition so that causes a lot of absenteeism. So a firm should conduct health campaigns to educate employee about the harmful effects of drinking and smoking.



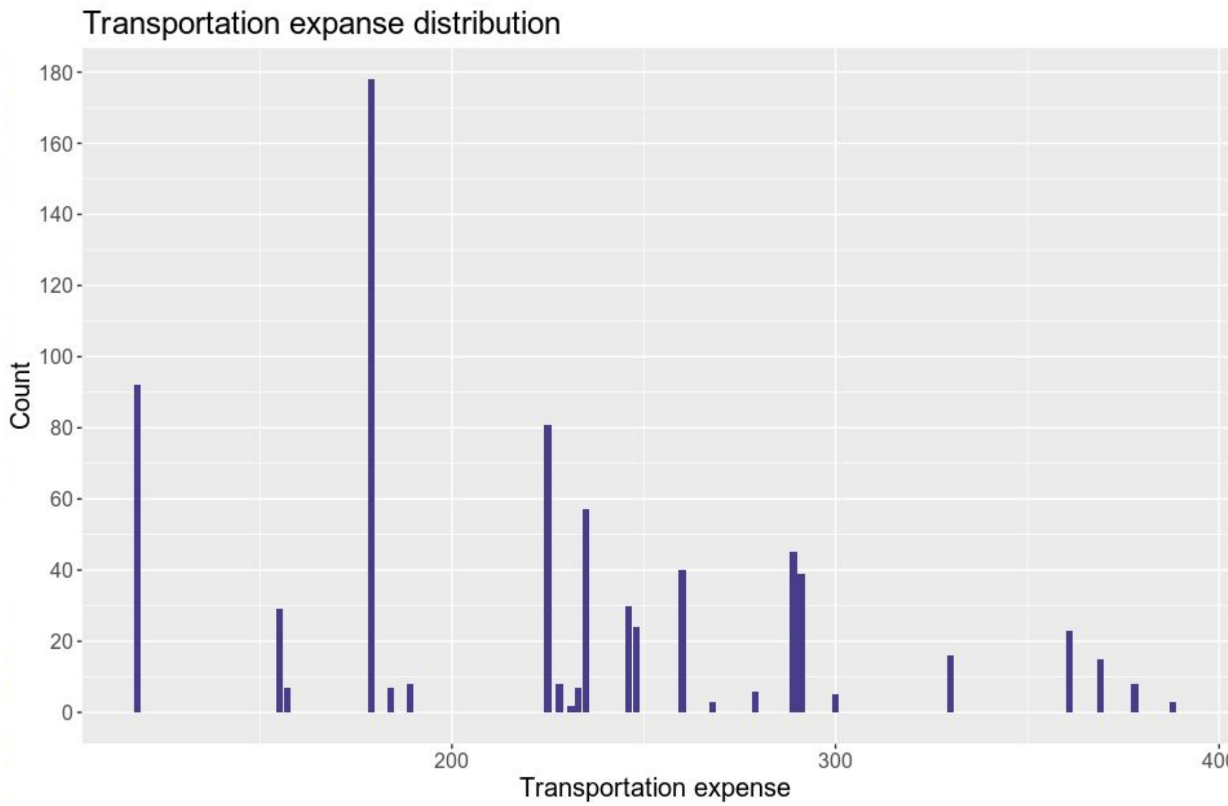
3. Above is the scatter plot of pets over absenteeism hours which show people having at least one pet shows less hours of absenteeism. So company should encourage its employees to keep pets at home.



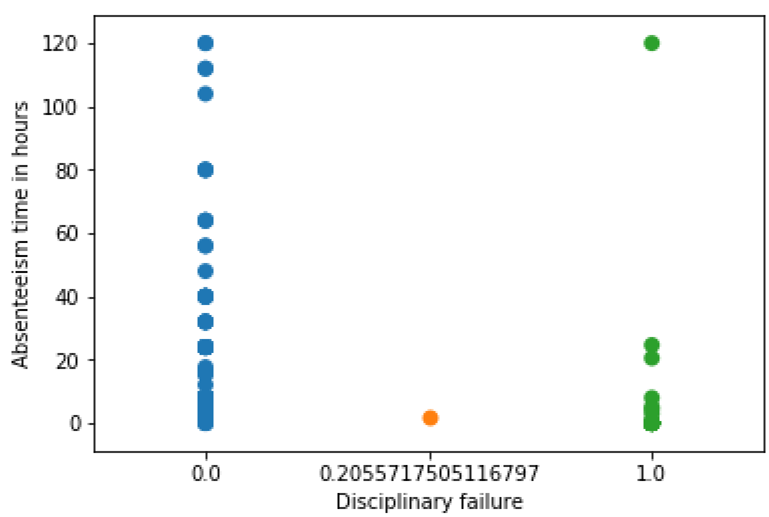
3. Most of the employees are absent on Monday. A company should motivate their employees for not being lazy.



4. Company should provide transportation expense to employees who are communicating from a considerable distance.



5. A company should provide proper teaching and training to their employees after interval of time as the discipline is the key to growth.



3.3.2 Problem 2

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

	Work Load Loss/Month
Janaury	6351550
Febraury	8268542
March	16584985
April	10999489
May	9985056
June	14779779
July	19106688
August	9482329
September	6709876
October	10437477
November	12682930
December	12299957

Appendix A

R Code :

```
#First clean the environment
rm(list = ls())
#set working directory
setwd("/home/neha/home/Project_2")
#Load required packages
x = c("xlsx", "DMwR","corrgram", "caret", "usdm", "rpart", "DataCombine",
"randomForest","e1071", "ggplot2", "inTrees", "lsr")
lapply(x, require, character.only=TRUE)
rm(x)
#Load the data
data = read.xlsx("Absenteeism_at_work_Project.xls", sheetIndex = 1)
```

########Explore the data#########

```
str(data)
dim(data)
summary(data)
class(data)
colnames(data)

#convert the variables into their respective types
data$ID = as.factor(as.character(data$ID))
data$Reason.for.absence = as.factor(as.character(data$Reason.for.absence))
data$Month.of.absence = as.factor(as.character(data$Month.of.absence))
data$Day.of.the.week = as.factor(as.character(data$Day.of.the.week))
data$Seasons = as.factor(as.character(data$Seasons))
data$Disciplinary.failure =
as.factor(as.character(data$Disciplinary.failure))
data$Education = as.factor(as.character(data$Education))
data$Social.drinker = as.factor(as.character(data$Social.drinker))
data$Social.smoker = as.factor(as.character(data$Social.smoker))

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

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

#store the missing value information
write.csv(missing_val, "Missing_Value_Percentage", row.names = F)

#Now let's impute missing value
#First we will create missing value in one cell to check

data$Body.mass.index[10] #value of this cell is 29
data$Body.mass.index[10] = NA

#Actual value : 29
#Mean value : 26.68
#Median Value : 25
#KNN Value : 29

#Now will apply mean method to impute this generated value and observe the
result
#Mean Method
#data$Body.mass.index[is.na(data$Body.mass.index)] =
mean(data$Body.mass.index, na.rm = T)

#Median Method
#data$Body.mass.index[is.na(data$Body.mass.index)] =
median(data$Body.mass.index, na.rm = T)

#KNN Method
data = knnImputation(data, k =3)

#after applying mean, median and KNN, we found KNN method is more accurate,
hence we freeze this
#check for missing value
sum(is.na(data)) #there is no missing value now

#########Outlier Analysis#############
```

```
numeric_index = sapply(data, is.numeric)
numeric_data = data[, numeric_index]

#Store all the column names excluding target variable name
cnames = colnames(numeric_data)[-12]

#Plotting boxplot to detect outliers
for (i in 1:length(cnames)) {
  assign(paste0("gn",i), ggplot(aes_string( y = (cnames[i]), x=
"Absenteeism.time.in.hours") , data = subset(data)) +
           stat_boxplot(geom = "errorbar" , width = 0.5) +
           geom_boxplot(outlier.color = "red", fill = "blue", outlier.shape
= 20, outlier.size = 1, notch = FALSE)+
           theme(legend.position = "bottom")+
           labs(y = cnames[i], x= "Absenteeism.time.in.hours")+
           ggtitle(paste("Boxplot" , cnames[i])))
  #print(i)
}

options(warn = 0)
#lets plot the boxplots
gridExtra::grid.arrange(gn1, gn2,gn3, ncol=3)
gridExtra::grid.arrange(gn4,gn5,gn6, ncol=3)
gridExtra::grid.arrange(gn7,gn8,gn9, ncol =3)
gridExtra::grid.arrange(gn10,gn11, ncol =3 )

#getting outliers using boxplot.stat method
for (i in cnames) {
  print(i)
  val = data[,i][data[,1] %in% boxplot.stats(data[,i])$out]
  print(length(val))
  print(val)
}

#Make each outlier as NA
for (i in cnames) {
  val = data[,i][data[,i] %in% boxplot.stats(data[,i])$out]
  data[,i][data[,i] %in% val] = NA
}

#checking the missing values
```

```
sum(is.na(data))

#Impute the values using KNN imputation method
data = knnImputation(data, k=3)

#Check again for missing value if present in case
sum(is.na(data))

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

#Correlation plot
corrgram(data[, cnames],order = F, upper.panel = panel.pie, text.panel =
panel.txt, main = "correlation plot" )

#ANOVA test

anova_test = aov(Absenteeism.time.in.hours ~ ID + Day.of.the.week +
Education + Social.smoker + Social.drinker + Reason.for.absence + Seasons +
Month.of.absence + Disciplinary.failure, data = data)
summary(anova_test)

#Dimensionality Reduction
data = subset(data,select = -c(Weight, Education, Social.drinker, Seasons,
Month.of.absence, Disciplinary.failure))

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

# Using histogram to check how if our data is normally distributed or not
hist(data$Body.mass.index)
hist(data$Age)
hist(data$Absenteeism.time.in.hours)
hist(data$Transportation.expense)
hist(data$Work.load.Average.day.)

#Hence we will choose normalisation instead of standardisation bcz
variables are not normally distributed

num_names = colnames(data[,sapply(data,is.numeric)])
num_names = num_names[-11]

for (i in num_names) {
  print(i)
```

```
  data[,i] = (data[,i] - min(data[,i]))/(max(data[,i]) - min(data[,i]))
}

###########Model Development#############

rmExcept("data")
#1.Decision Tree Regression

#divide the data into train and test
train_index = sample(1:nrow(data), 0.8*nrow(data))
train = data[train_index,]
test = data[-train_index,]

#Model
DT_Reg = rpart(Absenteeism.time.in.hours ~. , data = train, method =
"anova")

#Lets predict for test cases
Predictions_DT = predict(DT_Reg, test[-17])

#Evaluate the performance of model, using rmse as the data is time series
data
Rmse_DT = regr.eval(test[,15], Predictions_DT, stats = 'rmse')

#RMSE Value : 10.98
#Accuracy : 89.02

#2.Random Forest Regression

RF_Reg = randomForest(Absenteeism.time.in.hours ~. , train, importance =
TRUE, ntree = 100)

#Extract rules from random forest
#transform rf object to an inTrees' format
treeList = RF2List(RF_Reg)

#Extract rules
exec = extractRules(treeList, train[-15])
#Visualize some rules
exec[1:2,]
#Make rules more readable
ReadableRules = presentRules(exec, colnames(train))
```

```
ReadableRules[1:2,]
#Get rule metrics
RuleMetric = getRuleMetric(exec, train[-15],
train$Absenteeism.time.in.hours)
RuleMetric[1:2,]

#Predict test data using random forest model
Predictions_RF = predict(RF_Reg, test[-15])

##Evaluate the performance of model
Rmse_RF = regr.eval(test[,15], Predictions_RF, stats = 'rmse')

#RMSE Value : 8.53
#Hence Accuracy : 91.47

#3.Linear Regression

#First we need to check for multicollinearity

#Removing categorical data for checking multicollinearity
LR_data = subset(data, select = -c(ID, Reason.for.absence,
Month.of.absence, Day.of.the.week, Social.smoker, Social.drinker))
vif(LR_data[,-11])
vifcor(LR_data[,-11], th=0.9)

#Now will run the model
LR_Model = lm(Absenteeism.time.in.hours ~. , data = train[,
!colnames(train) %in% c("ID")])

#Summary of the model
summary(LR_Model)

#Predict
Predictions_LR = predict(LR_Model, test[,1:14])

#Calculate RMSE
RMSE(test[,15], Predictions_LR)

#RMSE Value : 9.42
#Accuracy : 90.58
```

Python Code :

#import Libraries

```
import os
import pandas as pd
import numpy as np
from fancyimpute import KNN
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import chi2_contingency
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from scipy import stats
import statsmodels.api as sm
import seaborn as sns
import matplotlib.gridspec as gridspec
from sklearn.cross_validation import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from ggplot import *

# In[3]:
#set working directory
os.chdir("/home/neha/home/Project_2/Python_Project2")

# In[4]:
#Load the data
employee_data = pd.read_excel("Absenteeism_at_work_Project.xls")

# In[5]:
employee_data.head()

# In[6]:
employee_data.info()

# # Exploratory Data Analysis
```

```
# In[7]:
employee_data['Month of absence'] = employee_data['Month of
absence'].astype(object)
employee_data['Day of the week'] = employee_data['Day of the
week'].astype(object)
employee_data['Seasons'] = employee_data['Seasons'].astype(object)
employee_data['Education'] = employee_data['Education'].astype(object)
employee_data['Son'] = employee_data['Son'].astype(object)
employee_data['Social drinker'] = employee_data['Social
drinker'].astype(object)
employee_data['Social smoker'] = employee_data['Social
smoker'].astype(object)
employee_data['Pet'] = employee_data['Pet'].astype(object)

# In[8]:
def get_cname(data):
    all_cnames = []
    num_cnames = []
    cat_cnames = []
    for i in data.columns:
        all_cnames.append(str(i))
        if(data[i].dtype == "object"):
            cat_cnames.append(str(i))
        else:
            num_cnames.append(str(i))
    cnames = [all_cnames, num_cnames, cat_cnames]
    return(cnames)

# In[9]:
cnames = get_cname(employee_data)


# In[10]:
rows = employee_data.shape[0] #gives number of row count
cols = employee_data.shape[1] #gives number of col cout


# In[11]:
# Change Data as per problem requirement

for i in range(0,rows):
    if employee_data["Absenteeism time in hours"][i] != 0:
```

```
        if employee_data["Reason for absence"][i] == 0:
            employee_data["Reason for absence"][i] = np.nan
    if employee_data["Month of absence"][i] == 0:
                employee_data["Month of absence"][i] = np.nan

def preprocessing(employee_data):

    # Change into require Data Types
    employee_data["ID"] = employee["ID"].astype(str)
    employee_data["Reason for absence"] = employee_data["Reason for
absence"].astype(str)
    employee_data["Month of absence"] = employee_data["Month of
absence"].astype(str)
    employee_data["Day of the week"] = employee_data["Day of the
week"].astype(str)
    employee_data["Seasons"] = employee_data["Seasons"].astype(str)
    employee_data["Disciplinary failure"] = employee_data["Disciplinary
failure"].astype(str)
    employee_data["Education"] = employee_data["Education"].astype(str)
    employee_data["Son"] = employee_data["Son"].astype(str)
    employee_data["Social drinker"] = employee_data["Social
drinker"].astype(str)
    employee_data["Social smoker"] = employee_data["Social
smoker"].astype(str)
    employee_data["Pet"] = employee_data["Pet"].astype(str)

    # Change NaN string values back to NaN
    employee_data["ID"] = employee_data["ID"].replace("nan",np.nan)
    employee_data["Reason for absence"] = employee_data["Reason for
absence"].replace("nan",np.nan)


  employee_data["Month of absence"] = employee_data["Month of
absence"].replace("nan",np.nan)
    employee_data["Day of the week"] = employee_data["Day of the
week"].replace("nan",np.nan)
    employee_data["Seasons"] =
employee_data["Seasons"].replace("nan",np.nan)
    employee_data["Disciplinary failure"] = employee_data["Disciplinary
failure"].replace("nan",np.nan)
    employee_data["Education"] =
employee_data["Education"].replace("nan",np.nan)
```

```
    employee_data["Son"] = employee_data["Son"].replace("nan",np.nan)
    employee_data["Social drinker"] = employee_data["Social
drinker"].replace("nan",np.nan)
    employee_data["Social smoker"] = employee_data["Social
smoker"].replace("nan",np.nan)
    employee_data["Pet"] = employee_data["Pet"].replace("nan",np.nan)

#Covert factor varaible values to labels
    for i in range(0, len(employee_data.columns)):
        if(employee_data.iloc[:,i].dtypes == 'object'):
            employee_data.iloc[:,i] =
pd.Categorical(employee_data.iloc[:,i])
            employee_data.iloc[:,i] = employee_data.iloc[:,i].cat.codes
            employee_data.iloc[:,i] =
employee_data.iloc[:,i].astype('object')

#Convert -1 values back to NaN
    for i in employee_data[0]:
        for j in range(0,rows):
            if Data.loc[j,i] == -1:
               Data.loc[j,i] = np.nan
    return employee_data


# In[12]:
employee_data.info()
# # Missing Value Analysis

# In[13]:
missing_val = pd.DataFrame(employee_data.isnull().sum())

# In[14]:
missing_val = missing_val.reset_index()

# In[15]:
missing_val = missing_val.rename(columns = {'index':'Variable',
0:'Missing_Percentage'})

# In[16]:
missing_val['Missing_Percentage'] =
(missing_val['Missing_Percentage']/len(employee_data))*100
```

```
# In[17]:
missing_val = missing_val.sort_values('Missing_Percentage', ascending = 
False).reset_index(drop = True)

# In[18]:
#now will go ahead and check which method works better for imputing missinf 
value out of mean, median & KNN
#will select one cell and put NA and try these method
#Actual Value : 23.0
#Mean Value : 26.68
#Median Value : 25
#KNN Value : 23.2

#employee_data['Body mass index'].iloc[12]


# In[19]:
#employee_data['Body mass index'].iloc[12] = np.nan


# In[20]:
missing_val

# In[21]:
#Mean
#employee_data['Body mass index'] = employee_data['Body mass 
index'].fillna(employee_data['Body mass index'].mean())

#Median
#employee_data['Body mass index'] = employee_data['Body mass 
index'].fillna(employee_data['Body mass index'].median())

#KNN
employee_data = pd.DataFrame(KNN(k = 3).complete(employee_data), columns = 
employee_data.columns)


# # Outlier Analysis

# In[22]:
employee_data.isnull().sum()
#there is no missing value left
```

```
# In[23]:
Data1 = employee_data.copy()


# In[24]:

#Plotting boxplot of all the continuous variable
get_ipython().run_line_magic('matplotlib', 'inline')
plt.boxplot(employee_data['Transportation expense'])
plt.xlabel('Transportation expense')
plt.title("BoxPlot of 'Transportation expense' ")
plt.ylabel('Values')

# In[25]:
plt.figure(figsize = [10.0, 7.0])
plt.boxplot([employee_data['Distance from Residence to Work'],
employee_data['Service time'], employee_data['Age'], employee_data['Hit
target'], employee_data['Son'], employee_data['Pet'],
employee_data['Weight'], employee_data['Height'], employee_data['Body mass
index']])
plt.xlabel(['1. Distance from Residence to Work', '2. Service time', '3.
Age', '4. Hit target', '5. Son', '6. Pet','7. Weight', '8. Height','9. Body
mass index'])
plt.title("BoxPlot of rest of the Variables")
plt.ylabel('Values')

# In[26]:
numeric_cnames = ['Transportation expense', 'Service time', 'Age', 'Work
load Average/day ', 'Pet', 'Height' ]

# In[27]:


#Detect and impute outliers with NA
for i in numeric_cnames:

    q75, q25 = np.percentile(employee_data[i], [75,25])

    iqr = q75 - q25
```

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

# In[28]:
employee_data.loc[employee_data[i] < min,i] = np.nan
employee_data.loc[employee_data[i] > max,i] = np.nan

# In[29]:

#check for missing value
employee_data.isnull().sum()

# In[30]:

#Imputing missing values with KNN
employee_data = pd.DataFrame(KNN(k=3).complete(employee_data), columns =
employee_data.columns)

# In[31]:

employee_data.isnull().sum()


# # Feature Selection
# In[32]:

numeric_cnames = ['Reason for absence', 'Transportation expense', 'Distance
from Residence to Work',
                  'Service time','Age', 'Hit target', 'Disciplinary
failure', 'Weight',
                  'Height', 'Body mass index']

# In[33]:
#Correlation plot
employee_corr = employee_data.loc[:, numeric_cnames]

# In[34]:

f, ax = plt.subplots(figsize=(10, 8))
corr = employee_corr.corr()
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool),
cmap=sns.diverging_palette(220, 10, as_cmap=True),
```

```
            square=True, ax=ax)

# In[35]:

object_cnames = ['Month of absence', 'Day of the week', 'Seasons',
'Education', 'Son','Social      drinker', Social smoker', 'Pet' ]

# In[36]:

#Chi-square test of independence
for i in object_cnames:
    print(i)
    chi2, p, dof, ex =
chi2_contingency(pd.crosstab(employee_data['Absenteeism time in hours'],
employee_data[i]))
    print(p)

# In[37]:

employee_data = employee_data.drop(['Weight', 'Month of absence', 'Day of
the week', 'Seasons','Education',  'Social smoker', 'Pet'], axis =1)

# In[38]:

numeric_cnames = ["Reason for absence", "Transportation expense", "Distance
from Residence to Work", "Service time", "Age",
                 "Disciplinary failure", "Hit target", "Height", "Body mass
index"]

# In[39]:

#Normalization

for i in numeric_cnames:
    print(i)
    employee_data[i] = (employee_data[i] -
np.min(employee_data[i]))/(np.max(employee_data[i]) -
np.min(employee_data[i]))

# # Model Development

# In[40]:
```

```
#Divide the data in train and test
X = employee_data.values[:,0:13]
y = employee_data.values[:,13]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

# In[41]:

from sklearn.metrics import mean_squared_error
from math import sqrt

def RMSE(y, pred):
    print(sqrt(mean_squared_error(y, pred)))

# In[42]:

#1.Decision Tree Regression
DT_regressor = DecisionTreeRegressor()
DT_regressor.fit(X_train, y_train)

# In[43]:

DT_predict = DT_regressor.predict(X_test)

# In[44]:

RMSE(y_test, DT_predict)
#RMSE Value : 16.79
#Accuracy : 83.21


# In[45]:
#2.Random Forest Regression
RF_regressor = RandomForestRegressor()
RF_regressor.fit(X_train, y_train)

# In[46]:

RF_predict = RF_regressor.predict(X_test)

# In[47]:
```

```
RMSE(y_test, RF_predict)
#RMSE Value : 9.97
#Accuracy : 90.03

# In[48]:

#3.Multiple Linear Regression
LR_regressor = LinearRegression()
LR_regressor.fit(X_train, y_train)

# In[49]:

LR_predict = LR_regressor.predict(X_test)

# In[50]:

RMSE(y_test, LR_predict)
#RMSE Value : 8.09
#Accuracy : 91.91


# In[51]:
#How much losses every month can we project in 2011 if same trend of
absenteeism continues

# In[52]:
loss_per_month = Data1[['Month of absence','Service time','Work load
Average/day ','Absenteeism time in hours']]

# In[53]:

loss_per_month["Loss"]=(loss_per_month['Work load Average/day
']*loss_per_month['Absenteeism time in hours'])/loss_per_month['Service
time']


# In[54]:

loss_per_month["Loss"] = np.round(loss_per_month["Loss"]).astype('int64')

# In[55]:
```

```
loss_per_month.head()

# In[56]:

#No_absent = loss_per_month[loss_per_month['Month of absence'] ==
0]['Loss'].sum()
January = loss_per_month[loss_per_month['Month of absence'] ==
1]['Loss'].sum()
February = loss_per_month[loss_per_month['Month of absence'] ==
2]['Loss'].sum()
March = loss_per_month[loss_per_month['Month of absence'] ==
3]['Loss'].sum()
April = loss_per_month[loss_per_month['Month of absence'] ==
4]['Loss'].sum()
May = loss_per_month[loss_per_month['Month of absence'] == 5]['Loss'].sum()
June = loss_per_month[loss_per_month['Month of absence'] ==
6]['Loss'].sum()
July = loss_per_month[loss_per_month['Month of absence'] ==
7]['Loss'].sum()
August = loss_per_month[loss_per_month['Month of absence'] ==
8]['Loss'].sum()
September = loss_per_month[loss_per_month['Month of absence'] ==
9]['Loss'].sum()
October = loss_per_month[loss_per_month['Month of absence'] ==
10]['Loss'].sum()
November = loss_per_month[loss_per_month['Month of absence'] ==
11]['Loss'].sum()
December = loss_per_month[loss_per_month['Month of absence'] ==
12]['Loss'].sum()

# In[57]:

record = {'January': January,'February': February,'March': March,
       'April': April, 'May': May,'June': June,'July': July,
       'August': August,'September': September,'October':
October,'November': November,
       'December': December}

# In[58]:

WorkLoss_permonth = pd.DataFrame.from_dict(record, orient='index')
```

```
# In[59]:

WorkLoss_permonth.rename(index=str, columns={0: "Work Load Loss/Month"})

## Data Visualizations

# In[72]:


ggplot(Data1, aes(x='Son', y='Absenteeism time in hours')) +
geom_bar(fill= "Purple") +    scale_color_brewer(type='diverging',
palette=5) +    xlab("Son") + ylab("Absenteeism in hr") +
ggtitle("Absenteeism Analysis") + theme_bw()

# In[73]:

ggplot(Data1, aes(x='Distance from Residence to Work', y='Absenteeism time
in hours')) +    geom_bar(fill= "Yellow") +
scale_color_brewer(type='diverging', palette=5) +    xlab("Distance from
Residence to Work") + ylab("Absenteeism in hr") + ggtitle("Absenteeism
Analysis") + theme_bw()

# In[79]:

ggplot(Data1, aes(x='Transportation expense', y='Absenteeism time in
hours')) +    geom_bar(fill= "Green") +
scale_color_brewer(type='diverging', palette=5) +    xlab("Transportation
expense") + ylab("Absenteeism in hr") + ggtitle("Absenteeism Analysis") +
theme_bw()

# In[62]:

ggplot(Data1, aes(x='Social drinker', y='Absenteeism time in hours')) +
geom_bar(fill= "Purple") +    scale_color_brewer(type='diverging',
palette=5) +    xlab("Social drinker") + ylab("absenteeism in hr") +
ggtitle("Absenteeism Analysis") + theme_bw()

# In[83]:

ggplot(Data1, aes(x='Body mass index', y='Absenteeism time in hours')) +
geom_bar(fill= "Grey") +    scale_color_brewer(type='diverging', palette=5)
+    xlab("Body mass index") + ylab("absenteeism in hr") +
```

```
ggtitle("Absenteeism Analysis") + theme_bw()

# In[64]:

sns.stripplot(x="Pet", y="Absenteeism time in hours", data=Data1, size =
8);
plt.savefig('Month of absence.png')

# In[92]:

sns.stripplot(x="Day of the week", y="Absenteeism time in hours",
data=Data1, size = 7);
plt.savefig('Day of the week.png')

# In[94]:

sns.stripplot(x="Disciplinary failure", y="Absenteeism time in hours",
data=Data1, size = 8);
plt.savefig('Disciplinary failure.png')

# In[61]:

ggplot(Data1, aes(x='Reason for absence', y='Absenteeism time in hours')) +
geom_bar(fill= "Purple") +    scale_color_brewer(type='diverging',
palette=5) +    xlab("Reasons") + ylab("Absenteeism in hr") +
ggtitle("Absenteeism Analysis") + theme_bw()
```