Employee Absenteeism

Juhi Sahu

18 Aug 2018

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

The objective of this project is to study employee details and behavior. We are provided with a dataset that has employee work, absent reason and if the employee time of absence. Given below is the data we should analyse to answer company's question.

Table 1.1: Employee Absenteeism at work 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
3	23	7	6	1	179

Table 1.2: Employee Absenteeism at work data (Columns: 7-12)

Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target	Disciplinary failure
36	13	33	2,39,554	97	0
13	18	50	2,39,554	97	1
51	18	38	2,39,554	97	0
5	14	39	2,39,554	97	0
36	13	33	2,39,554	97	0
51	18	38	2,39,554	97	0

Table 1.3: Employee Absenteeism at work 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
1	0	1	0	0	89

Table 1.3: Employee Absenteeism at work data (Columns: 19-21)

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

The data provided is done with exploratory analysis. The data of variables and are as follows.

- Individual identification (ID)
- Reason for absence (ICD).
  Absences attested by the International Code of Diseases (ICD) stratified into 21 categories (I to XXI) as follows:

1. Individual identification (ID)
2. Reason for absence (ICD).

Absences attested by the International Code of Diseases (ICD) stratified into 21 categories (1 to 21) as follows:

(1) Certain infectious and parasitic diseases
(2) Neoplasms
(3) Diseases of the blood and blood-forming organs and certain disorders involving the immune mechanism
(4) Endocrine, nutritional and metabolic diseases
(5) Mental and behavioral disorders
(6) Diseases of the nervous system
(7) Diseases of the eye and adnexa
(8) Diseases of the ear and mastoid process
(9) Diseases of the circulatory system
(10) Diseases of the respiratory system
(11) Diseases of the digestive system
(12) Diseases of the skin and subcutaneous tissue
(13) Diseases of the musculoskeletal system and connective tissue
(14) Diseases of the genitourinary system
(15) Pregnancy, childbirth and the puerperium
(16) Certain conditions originating in the perinatal period
(17) Congenital malformations, deformations and chromosomal abnormalities
(18) Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified
(19) Injury, poisoning and certain other consequences of external causes
(20) External causes of morbidity and mortality
(21) Factors influencing health status and contact with health services.

And 7 categories without (CID)
(22) Patient follow-up
(23) Medical consultation
(24) Blood donation
(25) Laboratory examination
(26) Unjustified absence
(27) Physiotherapy
(28) Dental consultatio

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

Our target variable is Absenteeism time in hours of which we have to answer the company loss and measures to be taken.

Chapter 2

Methodology

2.1 Pre Processing

Any predictive modeling requires that we look at the data before we start modeling. 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.
Our preprocessing involves following steps:

1. Missing value analysis
2. Outlier Analysis
3. Feature selection
   3.1 Correlation Analysis
   3.2 ANOVA
4. Normalization

2.1.1 Missing value analysis

Missing values occur when no data value is stored for the variable in an observation. Missing values are a common occurrence, and you need to have a strategy for treating them. A missing value can signify a number of different things in your data. Perhaps the data was not available or not applicable or the event did not happen. It could be that the person who entered the data did not know the right value, or missed filling in. Typically, ignore the missing values, or exclude any records containing missing values, or replace missing values with the mean, or infer missing values from existing values. We check for missing values in our data. There are missing values in the data that was given. Of the 21 variables provides 18 variables had the missing values. We imputed the missing values using median method. The code for the missing value imputation is written in appendix

2.1.2 Outlier Analysis

An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. Outliers can drastically change the results of the data analysis and statistical modeling. There are numerous unfavorable impacts of outliers in the data set. It increases the error variance and reduces the power of statistical tests. If the outliers are non-randomly distributed, they can decrease normality. They can also impact the basic assumption

of Regression, ANOVA and other statistical model assumptions. The boxplot for our data could be seen as follows:

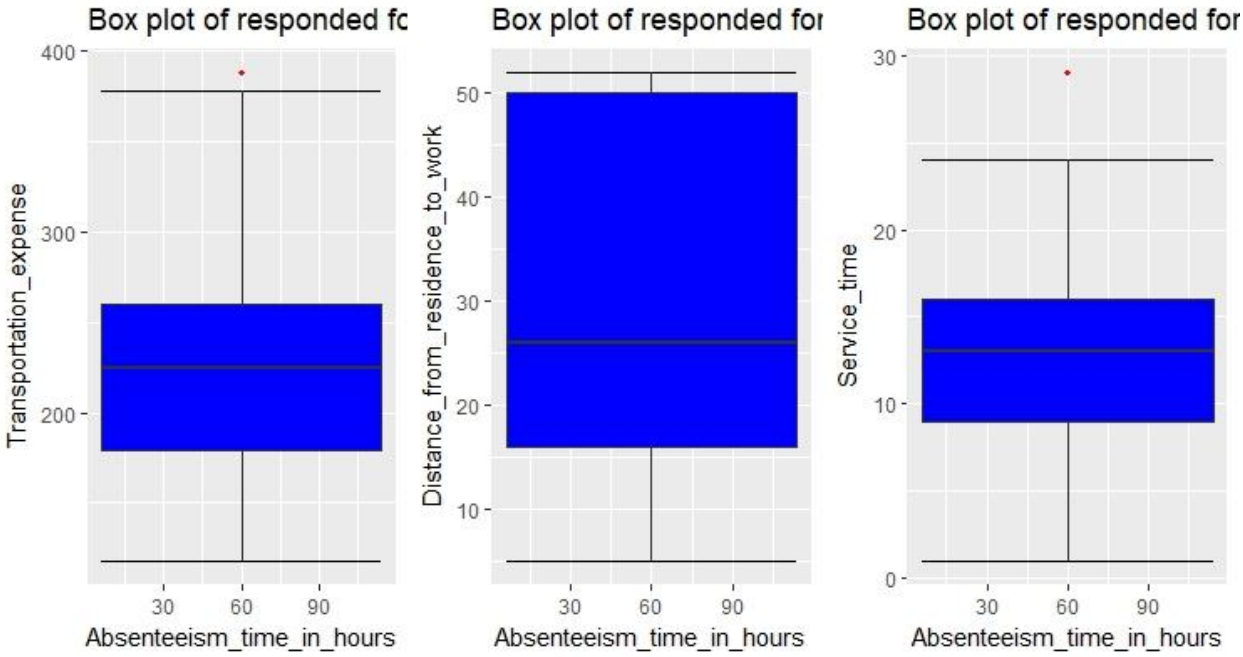


FIG 2.1

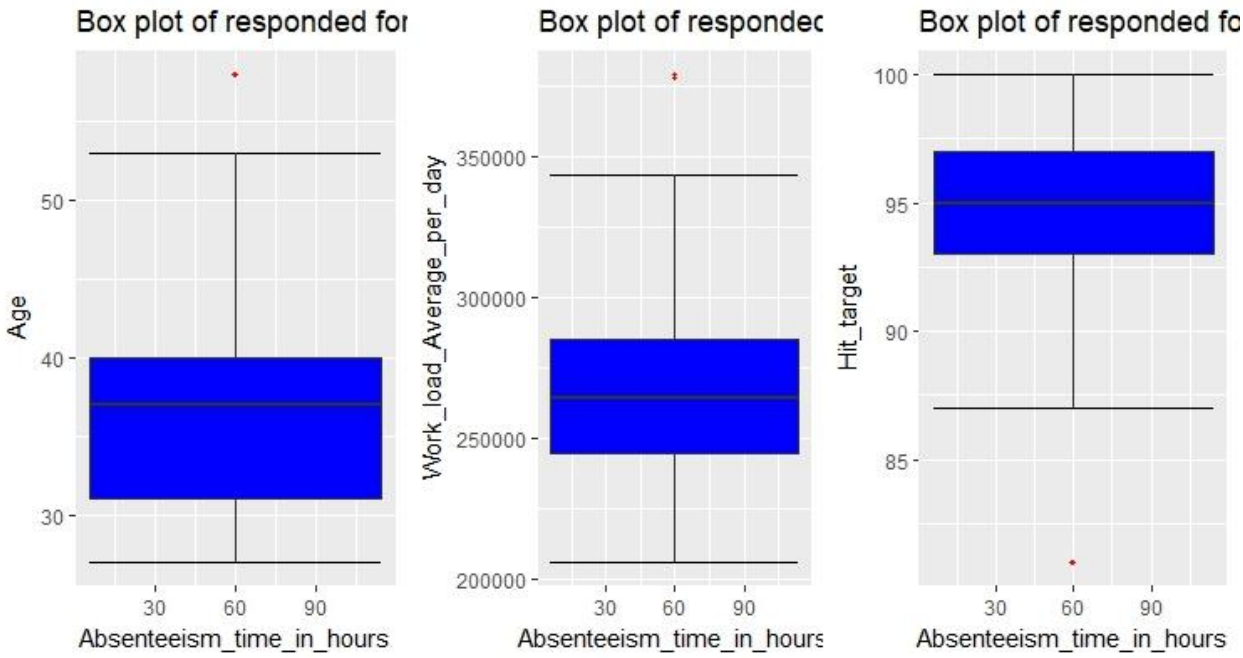


FIG 2.2

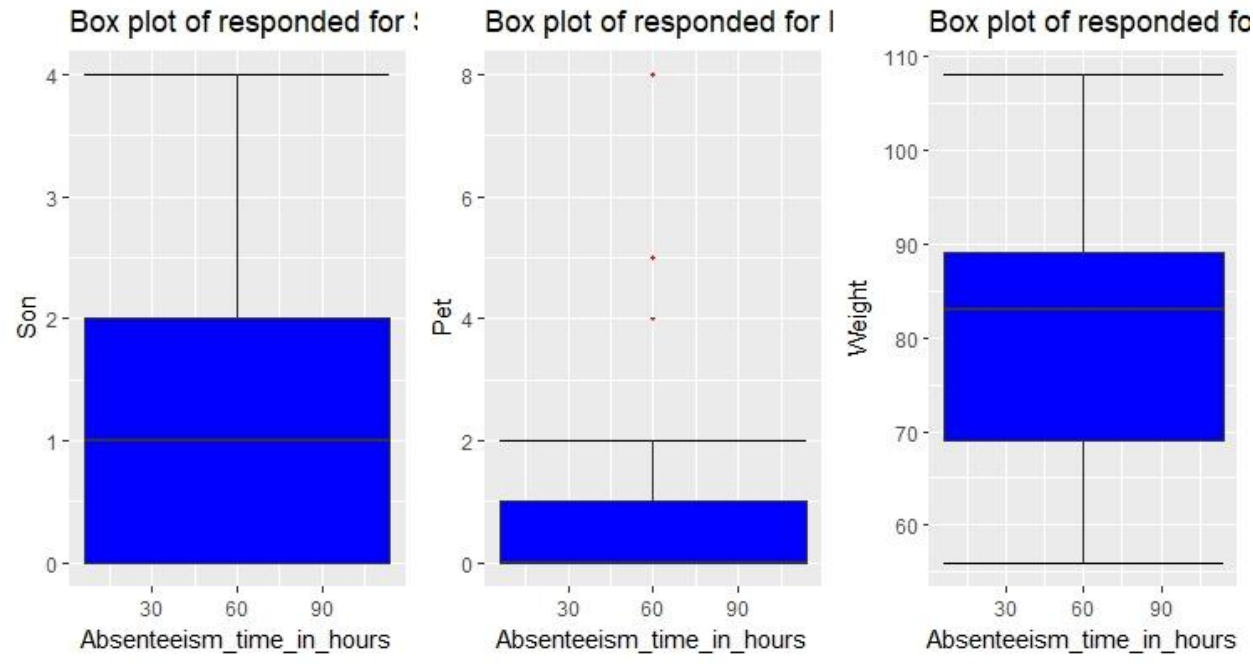

Box plot of responded for
Box plot of responded for
Box plot of responded fo
Son
Pet
Weight
Absenteeism_time_in_hours
Absenteeism_time_in_hours
Absenteeism_time_in_hours

FIG 2.3

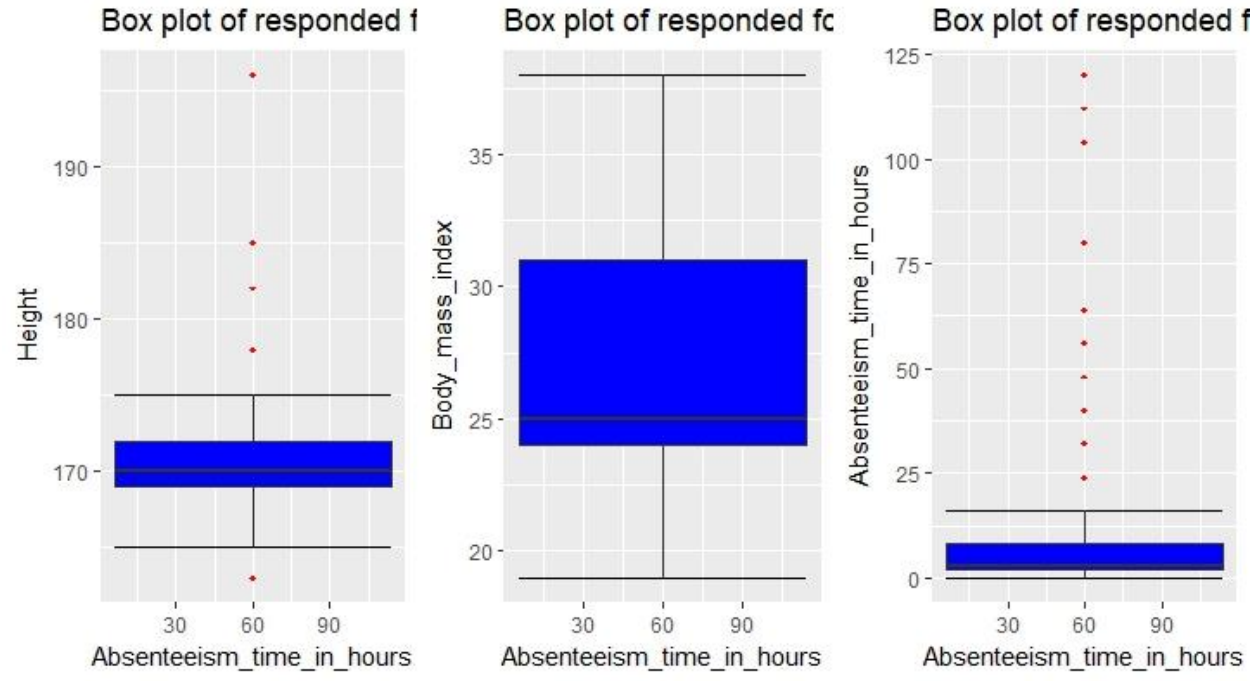

Box plot of responded f
Box plot of responded fo
Box plot of responded f
Height
Body_mass_index
Absenteeism_time_in_hours
Absenteeism_time_in_hours
Absenteeism_time_in_hours
Absenteeism_time_in_hours

FIG 2.4

The boxplot of the variables have given the outliers for Transportation expense, Service time, Age, Work load Average/day, Hit target, Pet, Height, Absenteeism time in hours. Since the outliers effects the ANOVA and linear regression models they either have to be deleted or imputed by using NAN method. We use the NAN method and impute the outliers removed using median method.

2.1.3 Feature Selection

Variable selection is an important aspect of model building. It helps in building predictive models free from correlated variables, biases and unwanted noise. It helps in selecting a subset of relevant features (variables, predictors) for use in model construction and subset of a learning algorithm's input variables upon which it should focus attention, while ignoring the rest.

Correlation Analysis

A correlation plot is a pie chart showing correlation coefficients between sets of variables. Each random variable (Xi) in the table is correlated with each of the other values in the table (Xj). This allows you to see which pairs have the highest correlation. Checking at the correlation of each numerical variable has given below result.

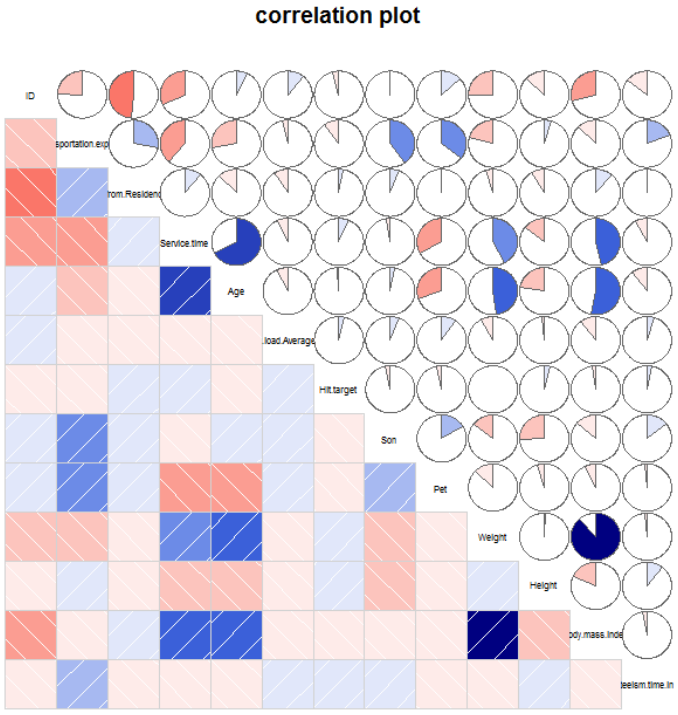


FIG 2.5

Our correlation matrix shows some interesting results as follows

1. Only weight and body mass index are correlated maximum.
2. Service time and age are slightly correlated

We can now remove one of the highly correlated variable so that our model can perform well with much accuracy.

ANOVA

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:

1. For Seasons:

```
 Coefficients:
            Estimate   Std.Error t value  Pr (>|t|)
(Intercept)   4.7649     0.2574  18.514   <2e-16 ***
Seasons2     -0.5577     0.3534  -1.578   0.1150
Seasons3     -0.1863     0.3574  -0.521   0.6024
Seasons4     -0.7636     0.3521  -2.169   0.0304 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.356 on 736 degrees of freedom
Multiple R-squared:  0.007908,  Adjusted R-squared:  0.003864
F-statistic: 1.956 on 3 and 736 DF,  p-value: 0.1193
```

2. For Reason:

```
 Coefficients:
                    Estimate   Std.Error  t value Pr(>|t|)
(Intercept)            7.8035     0.6866  11.365  < 2e-16 ***
Reason.for.absence10  -0.8465     0.8830  -0.959  0.33804
Reason.for.absence11  -2.2559     0.8765  -2.574  0.01026 *
Reason.for.absence12  -1.9082     1.2137  -1.572  0.11635
Reason.for.absence13  -1.8030     0.7856  -2.295  0.02201 *
Reason.for.absence14  -2.4767     0.9451  -2.621  0.00897 **
Reason.for.absence15   0.1965     2.1162   0.093  0.92606
Reason.for.absence16  -5.8035     1.7728  -3.274  0.00111 **
Reason.for.absence17   0.1965     2.9130   0.067  0.94625
Reason.for.absence18  -1.1049     0.9236  -1.196  0.23198
Reason.for.absence19  -0.5647     0.8196  -0.689  0.49103
Reason.for.absence2   -4.8476     2.9130  -1.664  0.09652 .
Reason.for.absence21  -1.9702     1.3443  -1.466  0.14320
Reason.for.absence22  -0.4668     0.8295  -0.563  0.57376
Reason.for.absence23  -4.9324     0.7247  -6.806 2.13e-11 ***
Reason.for.absence24   0.1965     1.7728   0.111  0.91179
Reason.for.absence25  -4.3197     0.8544  -5.056 5.45e-07 ***
Reason.for.absence26  -4.3873     0.7595  -5.776 1.14e-08 ***
```

```
Reason.for.absence27  -5.5282     0.7665  -7.212 1.41e-12 ***
Reason.for.absence28  -4.9775     0.7377  -6.747 3.13e-11 ***
Reason.for.absence3    0.1965     2.9130   0.067  0.94625
Reason.for.absence4   -3.3035     2.1162  -1.561  0.11896
Reason.for.absence5   -1.4702     1.7728  -0.829  0.40721
Reason.for.absence6   -1.1665     1.2137  -0.961  0.33684
Reason.for.absence7   -2.4685     1.0028  -2.462  0.01407 *
Reason.for.absence8   -2.4702     1.3443  -1.838  0.06654 .
Reason.for.absence9    1.2335     1.5732   0.784  0.43327
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.831 on 713 degrees of freedom
Multiple R-squared:  0.316,     Adjusted R-squared:  0.291
F-statistic: 12.67 on 26 and 713 DF,  p-value: < 2.2e-16
```

3. For Month of Week

```
Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)           0.750      1.660   0.452  0.65160
Month.of.absence1     2.742      1.725   1.589  0.11244
Month.of.absence2     3.086      1.706   1.809  0.07088 .
Month.of.absence3     4.459      1.698   2.626  0.00882 **
Month.of.absence4     3.786      1.722   2.199  0.02820 *
Month.of.absence5     3.399      1.711   1.986  0.04740 *
Month.of.absence6     3.733      1.721   2.169  0.03039 *
Month.of.absence7     4.895      1.709   2.864  0.00430 **
Month.of.absence8     4.023      1.721   2.338  0.01964 *
Month.of.absence9     2.843      1.722   1.651  0.09909 .
Month.of.absence10    3.453      1.707   2.023  0.04345 *
Month.of.absence11    3.373      1.712   1.970  0.04920 *
Month.of.absence12    3.343      1.727   1.936  0.05328 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.321 on 727 degrees of freedom
Multiple R-squared:  0.04038,   Adjusted R-squared:  0.02454
F-statistic: 2.549 on 12 and 727 DF,  p-value: 0.002632
```

4. For Day of the Week

```
Coefficients:
                  Estimate   Std.Error t value Pr(>|t|)
(Intercept)         4.8842     0.2646  18.462   <2e-16 ***
Day.of.the.week3   -0.6521     0.3784  -1.724   0.0852 .
Day.of.the.week4   -0.4820     0.3771  -1.278   0.2016
Day.of.the.week5   -0.5465     0.4002  -1.366   0.1725
Day.of.the.week6   -0.9338     0.3850  -2.425   0.0155 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.357 on 735 degrees of freedom
Multiple R-squared:  0.008515, Adjusted R-squared:  0.00312
F-statistic: 1.578 on 4 and 735 DF,  p-value: 0.1782
```

5. For Disciplinary Failure

```
Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)             4.5893     0.1222   37.54  < 2e-16 ***
Disciplinary.failure1  -4.1055     0.5325   -7.71 4.07e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.237 on 738 degrees of freedom
Multiple R-squared:  0.07455,   Adjusted R-squared:  0.07329
F-statistic: 59.45 on 1 and 738 DF,  p-value: 4.069e-14
```

6. For Education

```
 Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.28924    0.13585  31.574   <2e-16 ***
Education2   1.09513    0.51340   2.133   0.0332 *
Education3   0.09722    0.40148   0.242   0.8087
Education4   0.96076    1.68445   0.570   0.5686
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.358 on 736 degrees of freedom
Multiple R-squared:  0.006516,  Adjusted R-squared:  0.002466
F-statistic: 1.609 on 3 and 736 DF,  p-value: 0.1859
```

7. For Social Drinker

```
Coefficients:

                Estimate Std. Error t value Pr(>|t|)
(Intercept)       4.0888     0.1876   21.80   <2e-16 ***
Social.drinker1   0.5005     0.2490    2.01   0.0448 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.355 on 738 degrees of freedom
Multiple R-squared:  0.005446, Adjusted R-squared:  0.004098
F-statistic: 4.041 on 1 and 738 DF,  p-value: 0.04476
```

8. For Social Smoker

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      4.3274     0.1283  33.728   <2e-16 ***
Social.smoker1   0.6232     0.4750   1.312     0.19
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 3.36 on 738 degrees of freedom
Multiple R-squared:  0.002327, Adjusted R-squared:  0.0009753
F-statistic: 1.721 on 1 and 738 DF,  p-value: 0.1899
```

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. The variables that could be deleted are as follows

Numerical: Weight, Height, and Distance from Residence
Categorical: ID, Seasons, Day of the Week, Education, Disciplinary failure

2.1.4 Feature Scaling

Feature scaling is a method used to standardize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing steps. If training an algorithm using different features and some of them are off the scale in their magnitude, then the results might be dominated by them. Therefore, the range of all features should be normalized so that each feature contributes approximately proportionately to the final distance. We use normalization here for feature scaling.

Normalization also called Min-Max scaling. It is the process of reducing unwanted variation either within or between variables. Normalization brings all of the variables into proportion with one another. It transforms data into a range between 0 and 1. We have to see the variables that are scattered highly and apply normalization. We normalize the following variables in our data so that we can process to the modeling phase. Normality check for variables is in appendix

Variables

Transportation expense, Workload Average/day, Distance from Residence to Work, Service time, Weight

Formulae used for normalization is:

$$Value_{new} = \frac{Value - minValue}{maxValue - minValue}$$

Now our data is ready for applying models. We will use several machine learning regression models that reads the employee behavior data and provide less error rate so that our prediction will be accurate

2.2 Modeling

2.2.1 Model Selection

After a thorough preprocessing we will be using some regression models on our processed data to predict the target variable.

Decision Tree: Decision tree 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. Split of decision tree is seen in the below tree. Decision tree regression is as follows

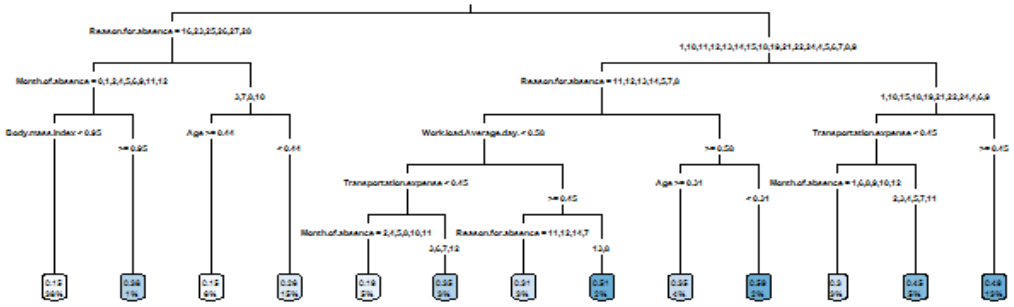


FIG 2.6

Random Forest: Random Forest or decision tree forests are an ensemble learning method for classification, regression and other tasks. It consists of an arbitrary number of simple trees, which are used to determine the final outcome. In the regression problem, their responses are averaged to obtain an estimate of the dependent variable. Using tree ensembles can lead to significant improvement in prediction accuracy (i.e., better ability to predict new data cases). The goal of using a large number of trees is to train enough that each feature has a chance to appear in several models. We can see certain rules of random forest in the R code.

```
Call:
 randomForest(formula = Absenteeism.time.in.hours ~ ., data = data1_train,
importance = TRUE, ntree = 500)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 4

          Mean of squared residuals: 0.03147593
                    % Var explained: 26.71
```

Error vs number of trees to be used graph is as follows:

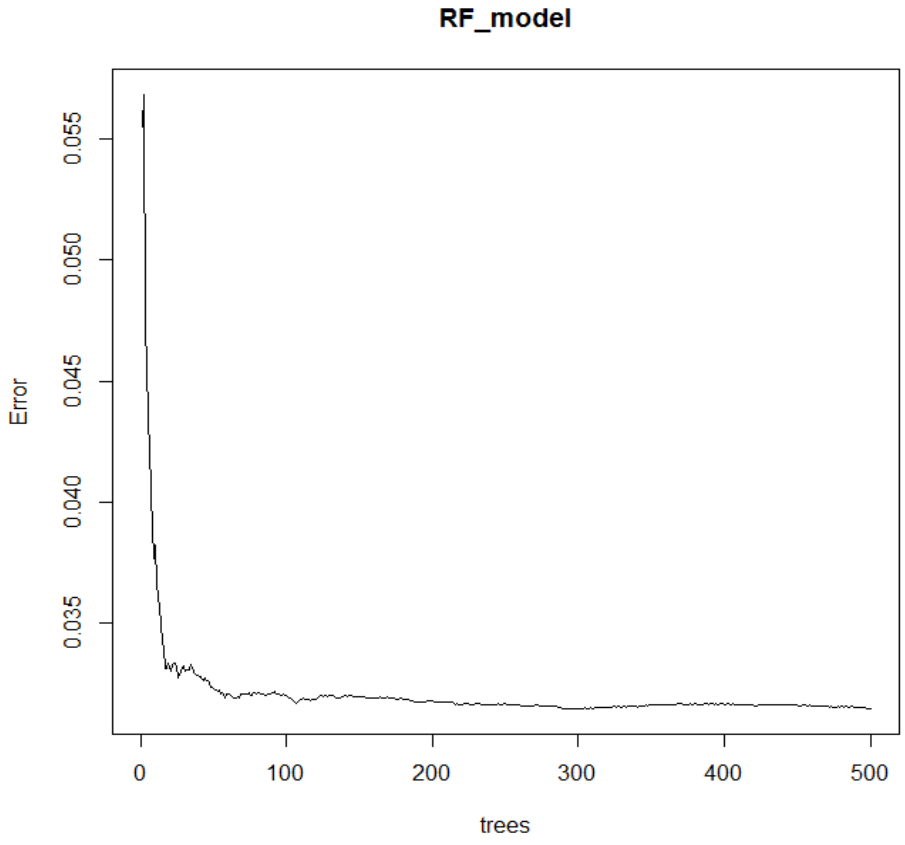


FIG 2.7

Linear Regression: Linear regression is the most basic type of regression and commonly used predictive analysis. Linear regression is an approach for modeling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables). The case of one explanatory variable is called simple linear regression. 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:

$$y = bo + b1x$$

Where bo = intercept, b1 = 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's 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

```
No variable from the 12 input variables has collinearity problem.

The linear correlation coefficients ranges between:
min correlation ( Pet ~ Work.load.Average.day. ):  -0.002690079
max correlation ( Body.mass.index ~ Weight ):  0.8805707

------- VIFs of the remained variables --------
                  Variables       VIF
1                                ID  1.733481
2           Transportation.expense  1.774494
3  Distance.from.Residence.to.Work  1.761654
4                     Service.time  3.442534
5                              Age  2.524362
6           Work.load.Average.day.  1.055865
7                       Hit.target  1.039801
8                              Son  1.266290
9                              Pet  1.541807
10                          Weight 21.601449
11                          Height  5.075401
12                 Body.mass.index 19.560069
```

DUMMY VARIABLE: In regression analysis, a dummy variable (also known as an indicat or 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 e

xample, in econometric time series analysis, dummy variables may be used to indicate t he 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 = df_train)

Residuals:
     Min       1Q   Median       3Q      Max
-0.42962 -0.08406 -0.00247  0.06084  0.78457

Coefficients: (8 not defined because of singularities)
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           0.283874   0.155708   1.823 0.068848 .
ID                   -0.002654   0.001092  -2.432 0.015360 *
Reason.for.absence.1  -0.149415   0.092332  -1.618 0.106207
Reason.for.absence.10 -0.151410   0.089015  -1.701 0.089540 .
Reason.for.absence.11 -0.259540   0.087861  -2.954 0.003276 **
Reason.for.absence.12 -0.138390   0.109304  -1.266 0.206031
Reason.for.absence.13 -0.213568   0.084588  -2.525 0.011866 *
Reason.for.absence.14 -0.266704   0.090429  -2.949 0.003325 **
Reason.for.absence.15 -0.077361   0.139055  -0.556 0.578215
Reason.for.absence.16 -0.516771   0.144084  -3.587 0.000366 ***
Reason.for.absence.17 -0.181170   0.185008  -0.979 0.327899
Reason.for.absence.18 -0.191716   0.090438  -2.120 0.034480 *
Reason.for.absence.19 -0.158322   0.085815  -1.845 0.065606 .
Reason.for.absence.2  -0.467436   0.180302  -2.593 0.009790 **
Reason.for.absence.21 -0.284272   0.113350  -2.508 0.012442 *
Reason.for.absence.22 -0.187142   0.087558  -2.137 0.033026 *
Reason.for.absence.23 -0.392736   0.081635  -4.811 1.96e-06 ***
Reason.for.absence.24 -0.110483   0.139084  -0.794 0.427342
Reason.for.absence.25 -0.354643   0.087291  -4.063 5.58e-05 ***
Reason.for.absence.26 -0.171606   0.087387  -1.964 0.050082 .
Reason.for.absence.27 -0.423231   0.086115  -4.915 1.19e-06 ***
Reason.for.absence.28 -0.401097   0.082668  -4.852 1.61e-06 ***
Reason.for.absence.3  -0.023639   0.180227  -0.131 0.895694
Reason.for.absence.4  -0.289419   0.139674  -2.072 0.038737 *
Reason.for.absence.5  -0.038830   0.139227  -0.279 0.780435
Reason.for.absence.6  -0.171548   0.103121  -1.664 0.096792 .
Reason.for.absence.7  -0.255737   0.092993  -2.750 0.006162 **
Reason.for.absence.8  -0.298331   0.144281  -2.068 0.039151 *
Reason.for.absence.9         NA         NA      NA       NA
Month.of.absence.0    -0.296115   0.088981  -3.328 0.000936 ***
Month.of.absence.1    -0.053284   0.057413  -0.928 0.353793
Month.of.absence.2    -0.031142   0.052811  -0.590 0.555644
Month.of.absence.3     0.021607   0.049982   0.432 0.665698
Month.of.absence.4    -0.084990   0.057310  -1.483 0.138670
Month.of.absence.5    -0.084955   0.058042  -1.464 0.143874
Month.of.absence.6    -0.056925   0.051685  -1.101 0.271226
Month.of.absence.7     0.032459   0.051158   0.634 0.526035
Month.of.absence.8     0.020421   0.053605   0.381 0.703388
Month.of.absence.9    -0.006040   0.043896  -0.138 0.890613
Month.of.absence.10   -0.032306   0.040659  -0.795 0.427221
```

```
Month.of.absence.11    -0.041440   0.036793  -1.126 0.260547
Month.of.absence.12           NA         NA      NA       NA
Day.of.the.week.2       0.011247   0.021535   0.522 0.601684
Day.of.the.week.3       0.020878   0.021696   0.962 0.336340
Day.of.the.week.4       0.008263   0.021432   0.386 0.699982
Day.of.the.week.5       0.030396   0.022817   1.332 0.183386
Day.of.the.week.6             NA         NA      NA       NA
Seasons.1              -0.025532   0.041072  -0.622 0.534440
Seasons.2              -0.011690   0.049026  -0.238 0.811624
Seasons.3               0.073548   0.051182   1.437 0.151306
Seasons.4                     NA         NA      NA       NA
Transportation.expense  0.113723   0.041518   2.739 0.006367 **
Service.time            0.052002   0.072971   0.713 0.476384
Age                    -0.061134   0.056661  -1.079 0.281096
Work.load.Average.day.  0.049583   0.037114   1.336 0.182126
Hit.target             -0.055412   0.043045  -1.287 0.198541
Disciplinary.failure.0  0.426651   0.044383   9.613  < 2e-16 ***
Disciplinary.failure.1        NA         NA      NA       NA
Education.1            -0.130547   0.086624  -1.507 0.132392
Education.2            -0.139301   0.098415  -1.415 0.157526
Education.3            -0.172036   0.096204  -1.788 0.074307 .
Education.4                   NA         NA      NA       NA
Son                     0.041945   0.030211   1.388 0.165602
Social.drinker.0        0.052360   0.024071   2.175 0.030053 *
Social.drinker.1              NA         NA      NA       NA
Social.smoker.0         0.013634   0.034858   0.391 0.695856
Social.smoker.1               NA         NA      NA       NA
Pet                    -0.090089   0.026766  -3.366 0.000819 ***
Body.mass.index         0.005473   0.050559   0.108 0.913845
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1562 on 531 degrees of freedom
Multiple R-squared:  0.4795,   Adjusted R-squared:  0.4207
F-statistic: 8.152 on 60 and 531 DF,  p-value: < 2.2e-16
```

Conclusion

3.1 Model Evaluation

Model evaluation is done on basis of evaluation metrics or error metrics. Evaluation metrics explain the performance of a model. An important aspect of evaluation metrics is their capability to discriminate among model results. Simply, building a predictive model is not our motive. But, creating and selecting a model which gives high accuracy on out of sample data. Hence, it is crucial to check accuracy or other metric of the model prior to computing predicted values. In our data as we applied regression models we have error metrics like Mean square error(MSE), MAPE, Root mean square error (RMSE), Mean absolute error (MAE) As out data is time variant data RMSE and MSE are the best error metrics to explain the accuracy of the models applied.

Decision Tree Regression:

```
MSE: 19.239333020892833
RMSE: 4.386266410159423
```

Random Forest Regression

```
MSE: 8.117834455989026
RMSE: 2.849181365934613
```

Linear Regression Model:

```
MSE: 10.156199121192424
RMSE: 3.1868792134614115
```

3.2 Model Selection

We can see that all models perform comparatively on average and therefore we select random forest classifier models for better prediction.

3.3 Answers

1. Company must bring the following changes for reducing absenteeism.

Let’s first see some of the visualizations to understand this more clearly.

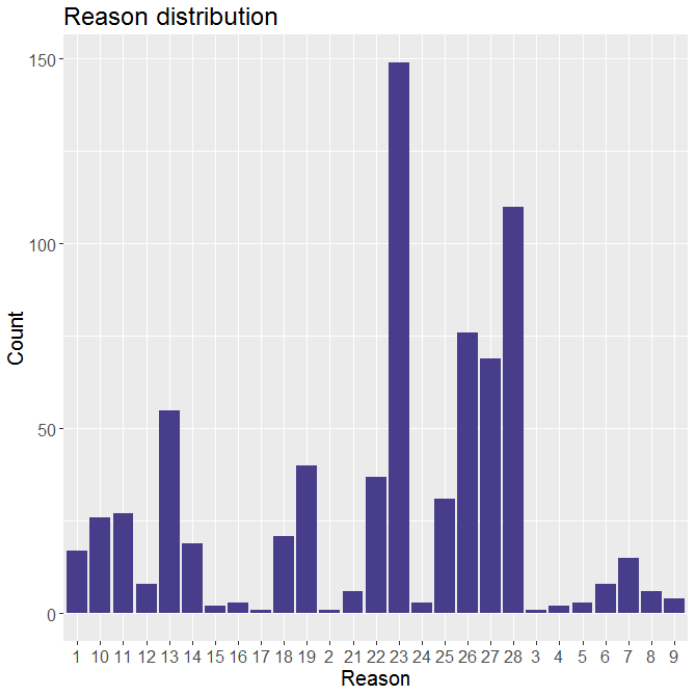


FIG 3.1

Above 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.

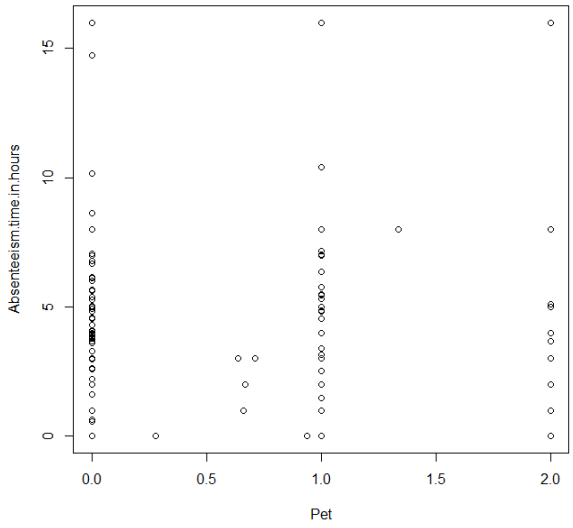


FIG 3.2

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.

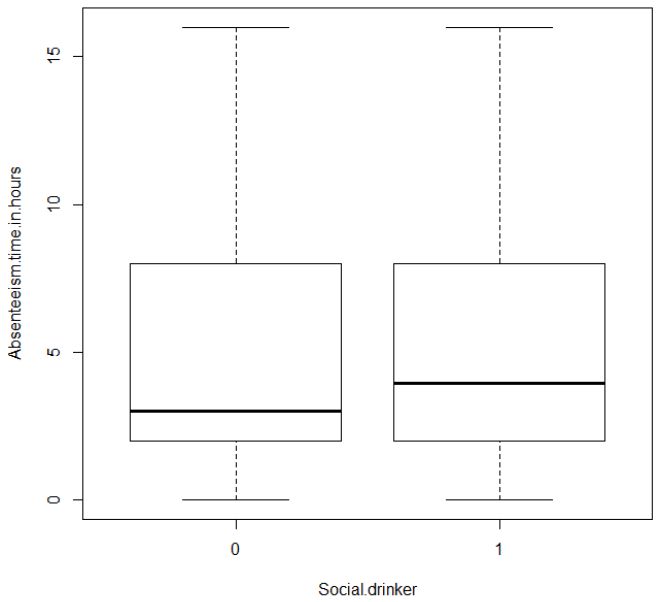


FIG 3.3

Above figure is a boxplot of social drinkers. The mean of people who are social drinkers are comparatively more, showing that drinkers tend to be more absent. Hence company should take measures to prevent its employees from drinking.

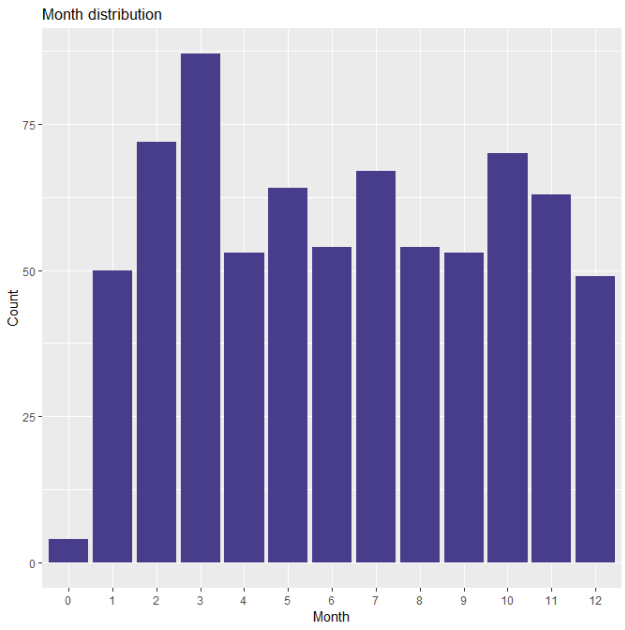


FIG 3.4

Above is the distribution of count of months in which employees were absent which shows that Month 3 has maximum absenteeism. For its counter effect, the company can give more stars to its employee having higher attendance in month 3. (Stars are directly proportional to the % salary hike an employee can have)

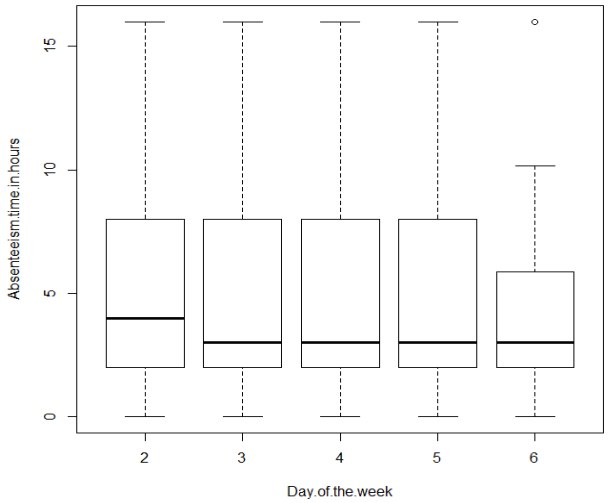


FIG 3.5

Above is the distribution of absenteeism over day of the week. It shows that Day6 (Friday) is the day of least absenteeism which can be due to the holidays following it i.e. Saturday, Sunday.
It can be inferred that absenteeism in hours is less seen when the following day is a holiday. So instead of giving 2 holidays back-to-back, the company can fix one holiday in a week day.

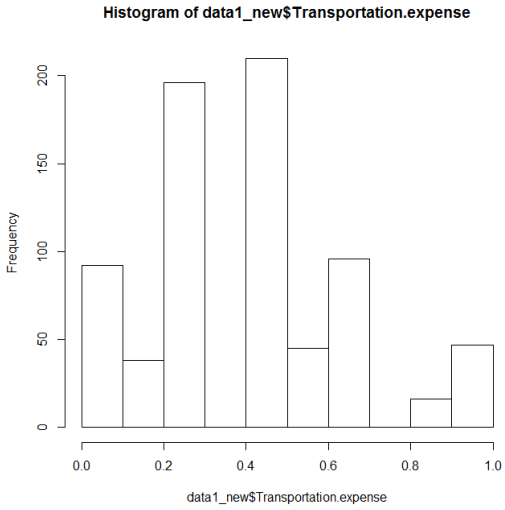
2. Loss for the compant in terms of work if same trend of absent continues.

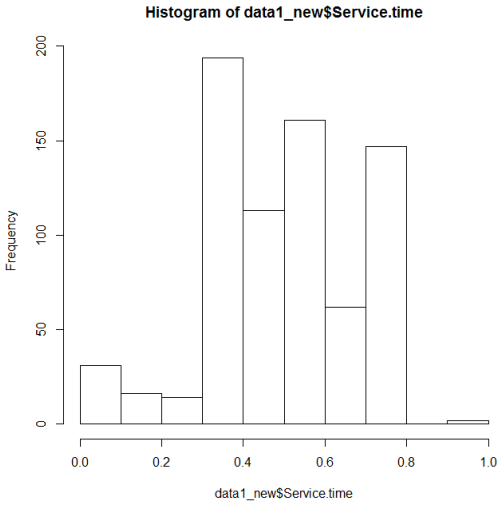
	Work Load Loss/Month
No Absent	0
Janaury	6312633
Febraury	8268542
March	16552699
April	10999489
May	10181350
June	14569137
July	19212717
August	9415534
September	6891606
October	9647002
November	12902373
December	12342048

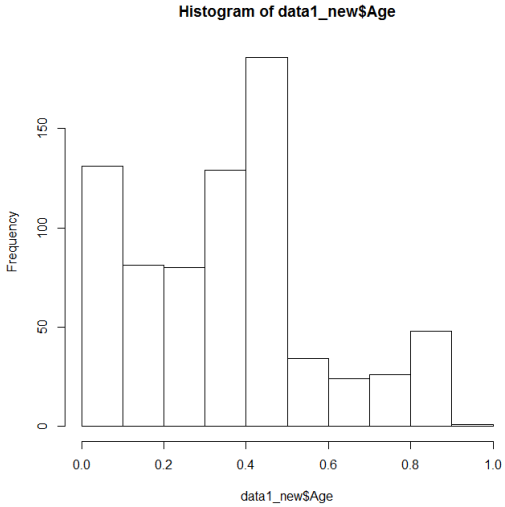
FIG 3.6

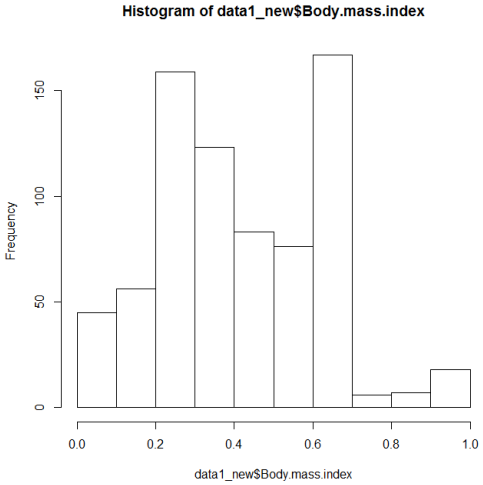
Appendix A – Extra Figures

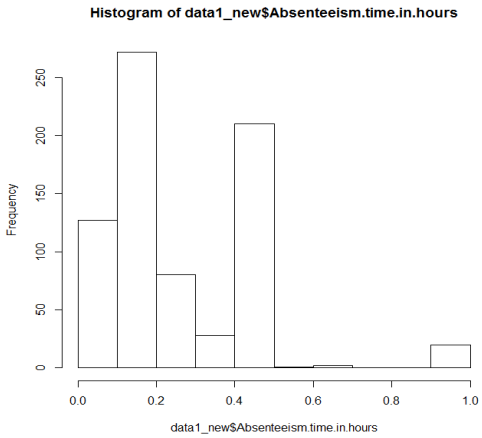
Normality check plots of various numerical variables:

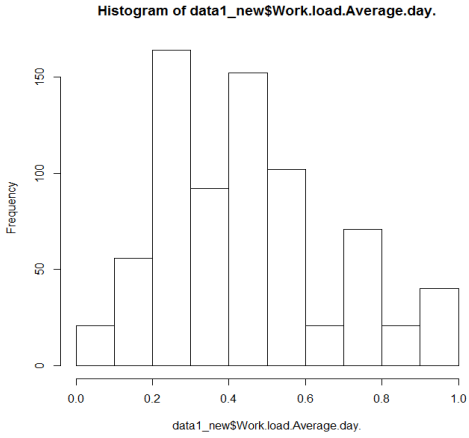












Appendix B - Code

R code

```
rm(list = ls())
setwd("C:/Users/dell/Desktop/edWisor Stuff/Project1")

#load the data
install.packages("xlsx")
library(xlsx)
data1 = read.xlsx("Absenteeism_at_work_Project.xls", sheetIndex = 1,
header = TRUE)
ds = data1

#data1=ds #copy
str(data1)

#PREPROCESSING
sapply(data1, function(x) sum(is.na(x)))
#converting data types
data1$Month.of.absence = factor(data1$Month.of.absence)
data1$Day.of.the.week= factor(data1$Day.of.the.week)
data1$Seasons = factor(data1$Seasons)
data1$Disciplinary.failure = factor(data1$Disciplinary.failure)
data1$Education = factor(data1$Education)
data1$Social.drinker = factor(data1$Social.drinker)
data1$Social.smoker = factor(data1$Social.smoker)

#col2
data1$Reason.for.absence[data1$Reason.for.absence == 0] = '26'
sum(is.na(data1$Reason.for.absence))
data1$Reason.for.absence = factor(data1$Reason.for.absence)

#col18
data1$Weight[is.na(data1$Weight)] = mean(data1$Weight, na.rm = T)
data1$Weight = as.integer(data1$Weight)

#col19
data1$Height[is.na(data1$Height)] = mean(data1$Height, na.rm = T)
data1$Height = as.integer(data1$Height)

#col20
data1$Body.mass.index[is.na(data1$Body.mass.index)] = (data1$Weight
*10000) /(data1$Height**2)
data1$Body.mass.index = as.integer(data1$Body.mass.index)

library(DMwR)
data1 = knnImputation(data1, k=3)
```

```
#-----------------------------------------VISVUALIZATION-------------
----------------------
library(ggplot2)
library(scales)
library(psych)

#?ggplot()
#ReasonsCount
ggplot(data1 , aes_string(x=data1$Reason.for.absence)) + geom_bar(stat
= "count", fill = "DarkslateBlue") +
xlab("Reason") + ylab("Count") + ggtitle("Reason distribution") +
theme(text = element_text(size = 15))
#non ICD reasons have higher count

#PetCount
ggplot(data1 , aes_string(x=data1$Pet)) + geom_bar(stat = "count",
fill = "DarkslateBlue") +
xlab("No Of Pets") + ylab("Count") + ggtitle("Pet distribution")
plot(Absenteeism.time.in.hours ~ Pet , data = data1)
#people with atleast one pet show less absentism in hours

#transportation expenses
ggplot(data1 , aes_string(x=data1$Transportation.expense)) +
geom_bar(stat = "count", fill = "DarkslateBlue") +
xlab("Transportation expense") + ylab("Count") +
ggtitle("Transportation expanse distribution")
plot(Absenteeism.time.in.hours ~ Transportation.expense, data = data1)

#Drinker
ggplot(data1 , aes_string(x=data1$Social.drinker)) + geom_bar(stat =
"count", fill = "DarkslateBlue") +
xlab("Drinker") + ylab("Count") + ggtitle("Drinker distribution")
plot(Absenteeism.time.in.hours ~ Social.drinker , data = data1)
#People who are social drinkers tend to be more absent

#Season
ggplot(data1 , aes_string(x=data1$Seasons)) + geom_bar(stat = "count",
fill = "DarkslateBlue") +
xlab("Seasons") + ylab("Count") + ggtitle("Season distribution")

#Month of absence
ggplot(data1 , aes_string(x= data1$Month.of.absence)) + geom_bar(stat
= "count", fill = "DarkslateBlue") +
xlab("Month") + ylab("Count") + ggtitle("Month distribution")
#month 3rd have higher absentism

#dayOfWeek
ggplot(data1 , aes_string(x=data1$Day.of.the.week)) + geom_bar(stat =
"count", fill = "DarkslateBlue") +
xlab("DayOfWeek") + ylab("Count") + ggtitle("Day distribution")
plot(Absenteeism.time.in.hours ~ Day.of.the.week , data = data1)
#people tend to be least absent on thursdays.
```

```
#disciplinary failure
ggplot(data1 , aes_string(x=data1$Disciplinary.failure)) +
geom_bar(stat = "count", fill = "DarkslateBlue") +
xlab("DayOfWeek") + ylab("Count") + ggtitle("Season distribution")
plot(Absenteeism.time.in.hours ~ Disciplinary.failure , data = data1)
#higher number of people do not have disciplinary failure


#---------------------------------------OUTLIER ANALYSIS--------------
----------------------
#Boxplotting
numeric_index = sapply(data1, is.numeric)
numeric_data = data1[,numeric_index]
cnames = colnames(numeric_data)
for (i in 1:length(cnames)){
assign(paste0("gn", i), ggplot(aes_string(y = cnames[i], x =
"Absenteeism.time.in.hours"), data = subset(data1)) +
stat_boxplot(geom = "errorbar", width = 0.5) +
geom_boxplot(outlier.colour = "red", fill = "grey", outlier.shape =
18, outlier.size = 1, notch = FALSE) +
theme(legend.position = "bottom") + labs(y = cnames[i], x="abs hours")
+ ggtitle(paste("Boxplot of abs
hours for", cnames[i])))
}

#plotting plots together
gridExtra::grid.arrange(gn2, gn3, ncol = 2)
gridExtra::grid.arrange(gn4, gn5, ncol = 2)
gridExtra::grid.arrange(gn6, gn7, ncol = 2)
gridExtra::grid.arrange(gn8, gn9, ncol = 2)
gridExtra::grid.arrange(gn10, gn11, ncol = 2)
gridExtra::grid.arrange(gn12, gn13, ncol = 2)

#replace outliers with NA and impute
for(i in cnames) {
print(i)
val = data1[,i][data1[,i] %in% boxplot.stats(data1[,i]) $out]
print(length(val))
data1[,i][data1[,i] %in% val] = NA
}
data1 = knnImputation(data1, k=3)
#sapply(data1, function(x) sum(is.na(x)))


#-----------------------------------FEATURE SELECTION--------------
-----------------
#correlation plot
library(corrgram)
round(cor(numeric_data),2)
```

```
corrgram(data1[, numeric_index], order = F, upper.panel = panel.pie,
text.panel = panel.txt, main = "correlation
plot")
data1_new = subset(data1, select=-c(Height, Weight,
Distance.from.Residence.to.Work))
#removing weight, Distance.from.Residence and height

#Anova test
#season of absence
AnovaModel_season =(lm(Absenteeism.time.in.hours ~ Seasons, data =
data1))
summary(AnovaModel_season) #remove

#Reason of absence
AnovaModel_reason=(lm(Absenteeism.time.in.hours ~ Reason.for.absence,
data = data1))
summary(AnovaModel_reason) #keep

#month of week
AnovaModel_month=(lm(Absenteeism.time.in.hours ~ Month.of.absence,
data = data1))
summary(AnovaModel_month) #keep

#day
AnovaModel_day=(lm(Absenteeism.time.in.hours ~ Day.of.the.week, data =
data1))
summary(AnovaModel_day) #remove

#Disciplinary failure
AnovaModel_disciplinary=(lm(Absenteeism.time.in.hours ~
Disciplinary.failure, data = data1))
summary(AnovaModel_disciplinary) #remove

#Education
AnovaModel_education=(lm(Absenteeism.time.in.hours ~ Education, data =
data1))
summary(AnovaModel_education) #remove

#drinker
AnovaModel_drinker=(lm(Absenteeism.time.in.hours ~ Social.drinker,
data = data1))
summary(AnovaModel_drinker) #keep

#smoker
AnovaModel_smoker=(lm(Absenteeism.time.in.hours ~ Social.smoker, data
= data1))
summary(AnovaModel_smoker) #keep
data1_new = subset(data1_new, select=-c(ID, Seasons, Day.of.the.week,
Education, Disciplinary.failure))


#-------------FEATURE SCALING---------------------
```

```
#normality check
hist(data1_new$Transportation.expense)
hist(data1_new$Service.time)
hist(data1_new$Age)
hist(data1_new$Work.load.Average.day.)
hist(data1_new$Body.mass.index)
hist(data1_new$Absenteeism.time.in.hours)
#since data is not normally distributed of any column we will use
normalization

cnames = cnames[-c(1,3,10,11)]
for (i in cnames){
print(i)
data1_new[,i] = (data1_new[,i]-min(data1_new[,i]))/
(max(data1_new[,i])-min(data1_new[,i]))
}


#-------------MODELING----------------------------
#sampling
train_index = sample(1:nrow(data1_new), 0.8*nrow(data1_new))
data1_train = data1_new[train_index,]
data1_test = data1_new[-train_index,]

#LINEAR REGRESSION
library(rpart)
library(MASS)

#check multicollinearity
install.packages("usdm")
library(usdm)
vif(numeric_data[,-13])
vifcor(numeric_data[,-13], th = 0.9)
#no variable from the 12 input variables has collinearity problem.

#creating dummies
install.packages("dummies")
library(dummies)
#?dummy.data.frame()
df_new = dummy.data.frame(data1_new, sep = '.')
dim(df_new)
colnames(df_new)

#sampling
train_index = sample(1:nrow(df_new), 0.8*nrow(df_new))
df_train = df_new[train_index,]
df_test = df_new[-train_index,]

#run regression model
lm_model11 = lm(Absenteeism.time.in.hours~. , data = df_train)
summary(lm_model11)
#R square= 0.47
```

```
#Adjusted R square = 0.42
#predict
predictions_LR = predict(lm_model11, df_test[,-69])
#Calculate MAPE
#mape = function(y, yhat){
# mean(abs((y-yhat)/y))*100
#}
#mape(df_test[,69], predictions_LR)
regr.eval(df_test[-69], predictions_LR, stats =
c('mse','rmse','mape','mae'))
#rmse = 30.35



#Decision Treet
install.packages("rpart.plot")
library(rpart.plot)
library(rpart)
fit = rpart(Absenteeism.time.in.hours~. , data = data1_train, method =
"anova")
plt = rpart.plot(fit, type = 3, digits = 2, fallen.leaves = TRUE)
Predict_DT = predict(fit, data1_test[,-20])

#accuracy
regr.eval(data1_test[,13], Predict_DT, stats = c('mae', 'mse', 'rmse',
'mape'))
#mae= 1.90
#mse= 8.08
#rmse= 2.84
#mape= Inf
#accuracy = 97.16


#Random Forest
install.packages("randomForest")
library(randomForest)
library(inTrees)
RF_model = randomForest(Absenteeism.time.in.hours~. , data1_train,
importance = TRUE, ntree = 500)
treeList = RF2List(RF_model)

#error plotting
plot(RF_model)
#extract rules
exec = extractRules(treeList, data1_train[,-13])
#visvualise rules
exec[1:2,]
#make rules more readable
readableRules = presentRules(exec, colnames(data1_train))
#Rule matrix
ruleMatrix = getRuleMetric(exec, data1_train[,-13],
data1_train$Absenteeism.time.in.hours)
```

```
#predict test data using RF model
RF_predict = predict(RF_model, data1_test[,-13])
#evaluate performance
postResample(RF_predict, data1_test$Absenteeism.time.in.hours)
#rmse = 2.87
#rsquare = 0.25
#mae = 2.05
#accuracy = 97.13


#-----------------------------MONTHLY LOSS
new = subset(data1, select = c(Month.of.absence, Service.time,
Absenteeism.time.in.hours, Work.load.Average.day. ))
#Work loss = ((Work load per day/ service time)* Absenteeism hours)
new["loss"]=with(new,((new[,4]*new[,3])/new[,2]))
for(i in 1:12)
{
d1=new[which(new["Month.of.absence"]==i),]
cat("\n month:",i, sum(d1$loss))
}
```

Python code

```
# coding: utf-8

# In[5]:


import os
os.chdir("C:/Users/dell/Desktop/edWisor Stuff/Project1")


# In[6]:


import pandas as pd
import numpy as np
get_ipython().run_line_magic('matplotlib', '')
import matplotlib.pyplot as plt


# In[7]:


df = pd.read_excel("Absenteeism_at_work_Project.xls")
data1 = pd.DataFrame(df)


# In[8]:


data1.shape


# In[9]:
```

```
data1.head(10)


# In[10]:


#changingg data types
data1['Reason for absence'] = data1['Reason for
absence'].astype('object')
data1['Month of absence'] = data1['Month of absence'].astype('object')
data1['Day of the week'] = data1['Day of the week'].astype('object')
data1['Seasons'] = data1['Seasons'].astype('object')
data1['Disciplinary failure'] = data1['Disciplinary
failure'].astype('object')
data1['Education'] = data1['Education'].astype('object')
data1['Social drinker'] = data1['Social drinker'].astype('object')
data1['Social smoker'] = data1['Social smoker'].astype('object')


# In[11]:


#missing value analysis
data1.isnull().sum()


# In[13]:


sum(data1.isnull().sum()) / len(data1)


# In[12]:


#treating col2
data1.loc[data1['Reason for absence'] == 0, ['Reason for absence']] =
26
data1['Reason for absence'] = data1['Reason for
absence'].fillna(data1['Reason for absence'].median())


# In[13]:


#treating col18
data1['Weight'] = data1['Weight'].fillna(data1['Weight'].mean())


# In[14]:
```

```
#treating col19
data1['Height'] = data1['Height'].fillna(data1['Height'].mean())


# In[15]:


#treating col20
data1['Body mass index'] = data1['Body mass
index'].fillna((data1['Weight']*10000)/data1['Height']**2)


# In[16]:


#treating col6
data1['Distance from Residence to Work'] = data1['Distance from
Residence to Work'].fillna(data1['Distance from Residence to
Work'].median())


# In[17]:


#treating col7
data1['Transportation expense'] = data1['Transportation
expense'].fillna(data1['Transportation expense'].median())


# In[18]:



data1['Service time'] = data1['Service time'].fillna(data1['Service
time'].median())


# In[19]:


data1['Work load Average/day '] = data1['Work load Average/day
'].fillna(data1['Work load Average/day '].median())


# In[20]:


data1['Age'] = data1['Age'].fillna(data1['Age'].mean())


# In[21]:
```

```
data1['Hit target'] = data1['Hit target'].fillna(data1['Hit
target'].mean())


# In[22]:


data1['Absenteeism time in hours'] = data1['Absenteeism time in
hours'].fillna(data1['Absenteeism time in hours'].mean())


# In[23]:


data1['Month of absence'] = data1['Month of
absence'].fillna(data1['Month of absence'].median())


# In[24]:


data1['Disciplinary failure'] = data1['Disciplinary
failure'].fillna(data1['Disciplinary failure'].median())


# In[25]:


data1['Education'] =
data1['Education'].fillna(data1['Education'].median())


# In[26]:


data1['Son'] = data1['Son'].fillna(data1['Son'].median())


# In[27]:


data1['Social drinker'] = data1['Social drinker'].fillna(data1['Social
drinker'].median())


# In[28]:


data1['Social smoker'] = data1['Social smoker'].fillna(data1['Social
smoker'].median())
```

```
# In[29]:


data1['Pet'] = data1['Pet'].fillna(data1['Pet'].median())


# In[ ]:


#knnImputation
#from fancyimpute import KNN ni chaling


# In[ ]:


#VISVUALIZATION


# In[30]:


#OUTLIERS

#create copy
copy1 = data1.copy()


# In[30]:


copy1


# In[46]:


numeric = data1[['Transportation expense', 'Distance from Residence to
Work',
                 'Service time', 'Age', 'Work load Average/day ',
'Hit target','Son','Pet',
                 'Weight', 'Height', 'Body mass index','Absenteeism
time in hours']]


# In[32]:


#plot boxplot
get_ipython().run_line_magic('matplotlib', 'inline')
```

```
# In[33]:


plt.boxplot(data1['Transportation expense'])


# In[36]:


#Replace Transportation expense outliers with NA
#Extract quartiles
q75, q25 = np.percentile(data1['Transportation expense'], [75 ,25])

#Calculate IQR
iqr = q75 - q25

#Calculate inner and outer fence
minimum = q25 - (iqr*1.5)
maximum = q75 + (iqr*1.5)

#Replace with NA
data1.loc[data1['Transportation expense'] < minimum,:'Transportation
expense'] = np.nan
data1.loc[data1['Transportation expense'] > maximum,:'Transportation
expense'] = np.nan


# In[37]:


plt.boxplot(data1['Distance from Residence to Work'])


# In[38]:


plt.boxplot(data1['Service time'])


# In[34]:


#Detect Service time outliers with NA
#Extract quartiles
q75, q25 = np.percentile(data1['Service time'], [75 ,25])

#Calculate IQR
iqr = q75 - q25

#Calculate inner and outer fence
```

```
minimum = q25 - (iqr*1.5)
maximum = q75 + (iqr*1.5)

#Replace with NA
data1.loc[data1['Service time'] < minimum,:'Service time'] = np.nan
data1.loc[data1['Service time'] > maximum,:'Service time'] = np.nan


# In[40]:


plt.boxplot(data1['Age'])


# In[35]:


#Detect and replace with NA
#Extract quartiles
q75, q25 = np.percentile(data1['Age'], [75 ,25])

#Calculate IQR
iqr = q75 - q25

#Calculate inner and outer fence
minimum = q25 - (iqr*1.5)
maximum = q75 + (iqr*1.5)

#Replace with NA
data1.loc[data1['Age'] < minimum,:'Age'] = np.nan
data1.loc[data1['Age'] > maximum,:'Age'] = np.nan


# In[42]:


plt.boxplot(data1['Work load Average/day '])


# In[36]:


#Detect and replace with NA
#Extract quartiles
q75, q25 = np.percentile(data1['Work load Average/day '], [75 ,25])

#Calculate IQR
iqr = q75 - q25

#Calculate inner and outer fence
minimum = q25 - (iqr*1.5)
maximum = q75 + (iqr*1.5)
```

```
#Replace with NA
data1.loc[data1['Work load Average/day '] < minimum,:'Work load
Average/day '] = np.nan
data1.loc[data1['Work load Average/day '] > maximum,:'Work load
Average/day '] = np.nan


# In[45]:


plt.boxplot(data1['Body mass index'])


# In[46]:


plt.boxplot(data1['Hit target'])


# In[37]:


#Detect and replace with NA
#Extract quartiles
q75, q25 = np.percentile(data1['Hit target'], [75 ,25])

#Calculate IQR
iqr = q75 - q25

#Calculate inner and outer fence
minimum = q25 - (iqr*1.5)
maximum = q75 + (iqr*1.5)

#Replace with NA
data1.loc[data1['Hit target'] < minimum,:'Hit target'] = np.nan
data1.loc[data1['Hit target'] > maximum,:'Hit target'] = np.nan


# In[48]:


plt.boxplot(data1['Absenteeism time in hours'])


# In[38]:


#Detect and replace with NA
#Extract quartiles
q75, q25 = np.percentile(data1['Absenteeism time in hours'], [75 ,25])
```

```
#Calculate IQR
iqr = q75 - q25

#Calculate inner and outer fence
minimum = q25 - (iqr*1.5)
maximum = q75 + (iqr*1.5)

#Replace with NA
data1.loc[data1['Absenteeism time in hours'] < minimum,:'Absenteeism
time in hours'] = np.nan
data1.loc[data1['Absenteeism time in hours'] > maximum,:'Absenteeism
time in hours'] = np.nan


# In[39]:


#Impute replaced outliers
data1['Transportation expense'] = data1['Transportation
expense'].fillna(data1['Transportation expense'].median())
data1['Service time']= data1['Service time'].fillna(data1['Service
time'].median())
data1['Height']= data1['Height'].fillna(data1['Height'].median())
data1['Pet']= data1['Pet'].fillna(data1['Pet'].median())
data1['Hit target']= data1['Hit target'].fillna(data1['Hit
target'].median())
data1['Age'] = data1['Age'].fillna(data1['Age'].median())
data1['Work load Average/day ']= data1['Work load Average/day
'].fillna(data1['Work load Average/day '].median())
data1['Absenteeism time in hours']= data1['Absenteeism time in
hours'].fillna(data1['Absenteeism time in hours'].median())


# In[40]:


data1['ID'] = copy1['ID']
data1['Reason for absence'] = copy1['Reason for absence']
data1['Month of absence'] = copy1['Month of absence']
data1['Day of the week'] = copy1['Day of the week']
data1['Seasons'] = copy1['Seasons']
data1['Distance from Residence to Work'] = copy1['Distance from
Residence to Work']
data1['Disciplinary failure'] = copy1['Disciplinary failure']
data1['Education'] = copy1['Education']
data1['Son'] = copy1['Son']
data1['Social drinker'] = copy1['Social drinker']
data1['Social smoker'] = copy1['Social smoker']
data1['Weight'] = copy1['Weight']
data1['Body mass index'] =  copy1['Body mass index']
```

```
# In[41]:


#converting datatypes
data1['ID'] = data1['ID'].astype('category')
data1['Reason for absence'] = data1['Reason for
absence'].astype('category')
data1['Month of absence'] = data1['Month of
absence'].astype('category')
data1['Day of the week'] = data1['Day of the week'].astype('category')
data1['Seasons'] = data1['Seasons'].astype('category')
data1['Disciplinary failure'] = data1['Disciplinary
failure'].astype('category')
data1['Education'] = data1['Education'].astype('category')
data1['Social drinker'] = data1['Social drinker'].astype('category')
data1['Social smoker'] = data1['Social smoker'].astype('category')
data1['Age'] = data1['Age'].astype('int')
data1['Son'] = data1['Son'].astype('int')
data1['Pet'] = data1['Pet'].astype('int')


# In[42]:


data1.dtypes


# In[68]:


data1.head(5)


# In[47]:


#FEATURE SELECTION

numeric = data1[['Transportation expense', 'Distance from Residence to
Work',
                 'Service time', 'Age', 'Work load Average/day ',
'Hit target','Son','Pet',
                 'Weight', 'Height', 'Body mass index','Absenteeism
time in hours']]


# In[43]:


numeric.shape
```

```
# In[48]:


#Set the width and hieght of the plot
f, ax = plt.subplots(figsize=(10,8))


# In[49]:


#correlation plot
corr = numeric.corr()


# In[52]:


#Plot using seaborn library
import seaborn as sns
get_ipython().run_line_magic('matplotlib', 'inline')
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool), cmap=
sns.diverging_palette(220,10,as_cmap=True),annot=True,
            square=True, ax=ax)


# In[112]:


corr


# In[51]:


sns.pairplot(data1)


# In[98]:


data1.groupby(["ID", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))


# In[101]:


data1.groupby(["Reason for absence", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))
```

```
# In[102]:


data1.groupby(["Month of absence", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))


# In[103]:


data1.groupby(["Day of the week", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))


# In[104]:


data1.groupby(["Education", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))
#company shoud employ


# In[105]:


data1.groupby(["Disciplinary failure", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))


# In[106]:


data1.groupby(["Social drinker", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))


# In[107]:


data1.groupby(["Social smoker", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))


# In[113]:
```

```
data1.groupby(["Transportation expense", "Absenteeism time in
hours"]).size().unstack().plot(kind='bar', stacked=True,
figsize=(10,10))


# In[122]:


data1.head(2)


# In[127]:


data1.dtypes


# In[121]:


#Droppong variables
data1 = data1.drop(['ID','Height','Weight','Distance from Residence to
Work'], axis=1)


# In[135]:


#ANOVA TEST
from scipy import stats


# In[163]:


#Reason for absence
grps2 = pd.unique(data1['Reason for absence'].values)
d_data = {grp:data1['Absenteeism time in hours'][data1['Reason for
absence'] == grp] for grp in grps2}
k= len(pd.unique(data1['Reason for absence']))
N = len(data1.values)
n = data1.groupby('Reason for absence').size()[0]

f,p = stats.f_oneway(d_data[1.0], d_data[2.0], d_data[3.0],
d_data[4.0], d_data[5.0], d_data[6.0], d_data[7.0], d_data[8.0],
d_data[9.0], d_data[10.0],
                     d_data[11.0], d_data[12.0], d_data[13.0],
d_data[14.0], d_data[15.0], d_data[16.0], d_data[17.0], d_data[18.0],
d_data[19.0], d_data[21.0],
                     d_data[22.0], d_data[23.0], d_data[24.0],
d_data[25.0], d_data[26.0], d_data[27.0], d_data[28.0])
```

```
f,p

#f = 10.14
#p = 3.21


# In[165]:


#Month of absence
grps2 = pd.unique(data1['Month of absence'].values)
d_data = {grp:data1['Absenteeism time in hours'][data1['Month of
absence'] == grp] for grp in grps2}
k= len(pd.unique(data1['Month of absence']))
N = len(data1.values)
n = data1.groupby('Month of absence').size()[0]

f,p = stats.f_oneway(d_data[0.0], d_data[1.0], d_data[2.0],
d_data[3.0], d_data[4.0], d_data[5.0], d_data[6.0], d_data[7.0],
d_data[8.0], d_data[9.0], d_data[10.0],
                     d_data[11.0], d_data[12.0])

f,p

#f = 2.35
#p = 0.005


# In[167]:


#Day of the week
grps2 = pd.unique(data1['Day of the week'].values)
d_data = {grp:data1['Absenteeism time in hours'][data1['Day of the
week'] == grp] for grp in grps2}
k= len(pd.unique(data1['Day of the week']))
N = len(data1.values)
n = data1.groupby('Day of the week').size()[0]

f,p = stats.f_oneway(d_data[2], d_data[3], d_data[4], d_data[5],
d_data[6])

f,p

#f = 0.99
#p = 0.41


# In[ ]:
```

```
#Seasons
grps = pd.unique(data1.Seasons.values)
d_data = {grp:data1['Absenteeism time in hours'][data1.Seasons == grp]
for grp in grps}
k= len(pd.unique(data1.Seasons))
N = len(data1.values)
n = data1.groupby('Seasons').size()[0]

f,p = stats.f_oneway(d_data[1], d_data[4], d_data[2], d_data[3])

f,p

#f= 0.89
#p = 0.44


# In[160]:


#Disciplinary failure
grps2 = pd.unique(data1['Disciplinary failure'].values)
d_data = {grp:data1['Absenteeism time in hours'][data1['Disciplinary
failure'] == grp] for grp in grps2}
k= len(pd.unique(data1['Disciplinary failure']))
N = len(data1.values)
n = data1.groupby('Disciplinary failure').size()[0]

f,p = stats.f_oneway(d_data[0.0], d_data[1.0])

f,p

#f = 38.90
#p = 7.49


# In[146]:


#Education
grps = pd.unique(data1.Education.values)
d_data = {grp:data1['Absenteeism time in hours'][data1.Education ==
grp] for grp in grps}
k= len(pd.unique(data1.Education))
N = len(data1.values)
n = data1.groupby('Education').size()[0]

f,p = stats.f_oneway(d_data[1.0], d_data[2.0], d_data[3.0],
d_data[4.0])

f,p

#f = 1.07
```

```
#p = 0.35


# In[159]:


#Social drinker
grps = pd.unique(data1['Social drinker'].values)
d_data = {grp:data1['Absenteeism time in hours'][data1['Social
drinker'] == grp] for grp in grps}
k= len(pd.unique(data1['Social drinker']))
N = len(data1.values)
n = data1.groupby('Social drinker').size()[0]

f,p = stats.f_oneway(d_data[1.0], d_data[0.0])

f,p

#f = 3.40
#p = 0.05


# In[169]:


#Social smoker
grps = pd.unique(data1['Social smoker'].values)
d_data = {grp:data1['Absenteeism time in hours'][data1['Social
smoker'] == grp] for grp in grps}
k= len(pd.unique(data1['Social smoker']))
N = len(data1.values)
n = data1.groupby('Social smoker').size()[0]

f,p = stats.f_oneway(d_data[1.0], d_data[0.0])

f,p

#f = 2.37
#p = 0.12


# In[170]:


#Droppong variables
data1 = data1.drop(['Day of the week', 'Seasons', 'Disciplinary
failure', 'Education'], axis=1)


# In[205]:
```

```
train.shape


# In[173]:


#NORMALIZATION

cnames = ['Transportation expense','Service time', 'Age','Work load
Average/day ','Hit target', 'Son','Pet','Body mass index']

for i in cnames:
    print(i)
    data1[i] = (data1[i] - min(data1[i]))/(max(data1[i]) -
min(data1[i]))


# In[176]:


# MODELING

#Sampling
train, test = train_test_split(data1, test_size = 0.2)


# In[182]:


#DECISION TREE REGRESSION
from sklearn.cross_validation import train_test_split
from sklearn.tree import DecisionTreeRegressor


# In[206]:


fit = DecisionTreeRegressor(max_depth =
2).fit(train.iloc[:,0:12],train.iloc[:,12])


# In[207]:


prediction_dt = fit.predict(test.iloc[:,0:12])


# In[208]:


#error matrix
from sklearn import metrics
```

```
# In[209]:


print('MSE:', metrics.mean_squared_error(test.iloc[:,10],
prediction_dt))
print('RMSE:', np.sqrt(metrics.mean_squared_error(test.iloc[:,10],
prediction_dt)))

#mse = 19.23
#rmse = 4.38

#accuracy = 100-rmsse = 95.62


# In[210]:


#RANDOM FOREST REGRESSION
from sklearn.ensemble import RandomForestRegressor


# In[216]:


rf = RandomForestRegressor(n_estimators = 500, random_state =
42).fit(train.iloc[:,0:12],train.iloc[:,12])


# In[217]:


prediction_rf = rf.predict(test.iloc[:,0:12])


# In[218]:


#error matrix
print('MSE:', metrics.mean_squared_error(test.iloc[:,12],
prediction_rf))
print('RMSE:', np.sqrt(metrics.mean_squared_error(test.iloc[:,12],
prediction_rf)))

#mse = 8.11
#rmse = 2.84

#accuracy = 97.16


# In[226]:
```

```
#LINEAR REGRESSION
import statsmodels.api as sm


# In[230]:


data1.dtypes


# In[232]:


pd.get_dummies(data1, columns = ['Reason for absence', 'Month of
absence', 'Social drinker', 'Social smoker'], drop_first = True)


# In[234]:


#sampling
train, test = train_test_split(data1, test_size = 0.2)


# In[236]:


model =
sm.OLS(train.iloc[:,12],train.iloc[:,0:12].astype(float)).fit()


# In[238]:


model.summary()

#r-square = 0.66
#adj r-sq = 0.65


# In[239]:


prediction_lr = model.predict(test.iloc[:,0:12])


# In[240]:


#ERROR METRICS
```

```
print('MSE:', metrics.mean_squared_error(test.iloc[:,12],
prediction_lr))
print('RMSE:', np.sqrt(metrics.mean_squared_error(test.iloc[:,12],
prediction_lr)))

#mse = 10.15
#rmse = 3.18

#accuracy = 100-rmse = 96.82


# In[255]:


#we fix Random Forest as our model (accuracy = 97.16)


# In[242]:


#LOSSES EVERY MONTH
new = copy1[['Month of absence','Service time','Work load Average/day
','Absenteeism time in hours']]


# In[243]:


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


# In[245]:


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


# In[246]:


new.head()


# In[247]:


No_absent = new[new['Month of absence'] == 0]['Loss'].sum()
January = new[new['Month of absence'] == 1]['Loss'].sum()
February = new[new['Month of absence'] == 2]['Loss'].sum()
March = new[new['Month of absence'] == 3]['Loss'].sum()
April = new[new['Month of absence'] == 4]['Loss'].sum()
```

```
May = new[new['Month of absence'] == 5]['Loss'].sum()
June = new[new['Month of absence'] == 6]['Loss'].sum()
July = new[new['Month of absence'] == 7]['Loss'].sum()
August = new[new['Month of absence'] == 8]['Loss'].sum()
September = new[new['Month of absence'] == 9]['Loss'].sum()
October = new[new['Month of absence'] == 10]['Loss'].sum()
November = new[new['Month of absence'] == 11]['Loss'].sum()
December = new[new['Month of absence'] == 12]['Loss'].sum()


# In[248]:


record = {'No Absent': No_absent, 'Janaury': January,'Febraury':
February,'March': March,
       'April': April, 'May': May,'June': June,'July': July,
       'August': August,'September': September,'October':
October,'November': November,
       'December': December}


# In[249]:


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


# In[252]:


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