PREDICTION OF LOSSES DUE TO EMPLOYEE ABSENTIEESM

TEJASH S. POPATE

July 2019

Table of Contents

Chapter 1 Introduction:

1.1 Problem Statement:

Given that XYZ is a courier company. It is appreciated 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?

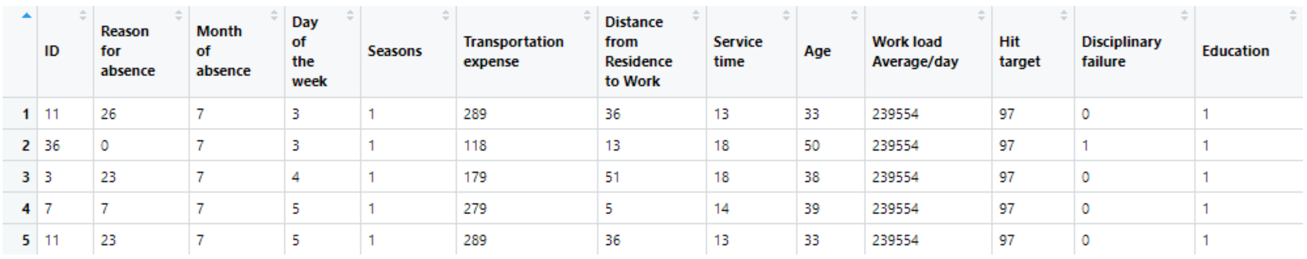
2. How much losses every month can be projected in 2011 if same trend of absenteeism continues?

The objective of the project is to find out reasons and patterns behind the issue of absenteeism and suggest some changes to the courier company accordingly to reduce the number of absenteeism. Further, based on the same trends in the dataset, human capital losses are to be predicted for every month for year 2011.

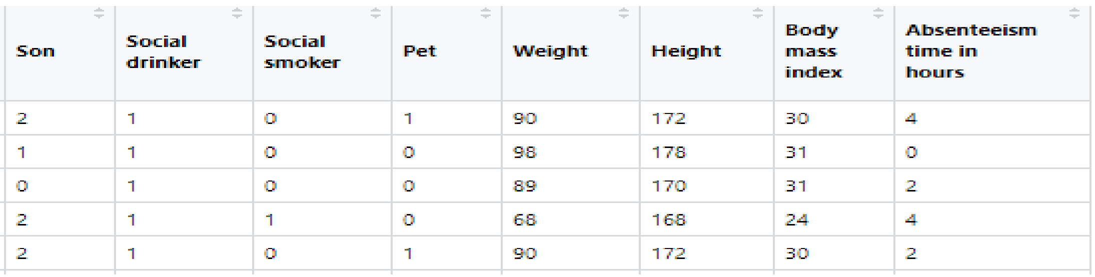
1.2 Data:

The aim is to predict the human capital losses that is absenteeism time in hours using given dataset and implementation of machine learning models. Given below is a sample of dataset that is for rest of this project.

Table 1.1: Employee Absenteeism Data (Columns 1-21)



	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target	Disciplinary failure	Education
1	11	26	7	3	1	289	36	13	33	239554	97	0	1
2	36	0	7	3	1	118	13	18	50	239554	97	1	1
3	3	23	7	4	1	179	51	18	38	239554	97	0	1
4	7	7	7	5	1	279	5	14	39	239554	97	0	1
5	11	23	7	5	1	289	36	13	33	239554	97	0	1



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

In the given dataset, there are **21 variables** of data type numeric and total **740 observations** including missing values in some of the rows. After giving a look to the dataset, the variables have been defined in different category:

Table 1.2: Employee Absenteeism Variables Category

Type of variable	Data Type	Variable Category
Predictor Variables: 1) ID 2) Reason for absence	**Character:**	**Categorical:** 1) ID 2) Reason for absence

3) Month of absence 4) Day of the week 5) Seasons 6) Transportation expense 7) Distance from Residence to Work 8) Service time 9) Age 10) Work load average/day 11) Hit target 12) Disciplinary failure 13) Education 14) Son 15) Social drinker 16) Social smoker 17) Pet 18) Weight 19) Height **20)** Body mass index	None of the variables are of this type.	3) Month of absence 4) Day of the week 5) Seasons 6) Disciplinary failure 7) Education 8) Social drinker 9) Social smoker 10) Son 11) Pet
Target Variable: Absenteeism time in hours	**Numeric:** All of them.	**Continuous:** 1) Transportation expense 2) Distance from Residence to Work 3) Service time 4) Age 5) Work load average/day 6) Hit Target 7) Weight 8) Height 9) Body mass index

Chapter 2 Methodology:

2.1 Data Pre-processing:

2.1.1 Exploratory Data Analysis: (EDA)

It is used to get the initial insights of data. It helps in understanding all the statistical parameters of both categorical and numeric variables/features from the data and also it helps in understanding presence of missing values and unique values in each feature. The imputation of these missing values and impact of imputation on the overall data is explained in the next step.

EDA includes two parts:

a) Univariate Analysis b) Bivariate Analysis

Univariate Analysis:This type of analysis helps in detecting anomaly in the data. Exploration depends on type of variable. If it is a continuous variable, the parameters such as central tendency, dispersion and distribution of variable (symmetric /right skewed/ left skewed)) are considered. If it is categorical variable, then frequency table, histogram and bar-plot are checked. Following tables and figures shows univariate analysis of each variable.

Table 2.1 EDA of categorical variables

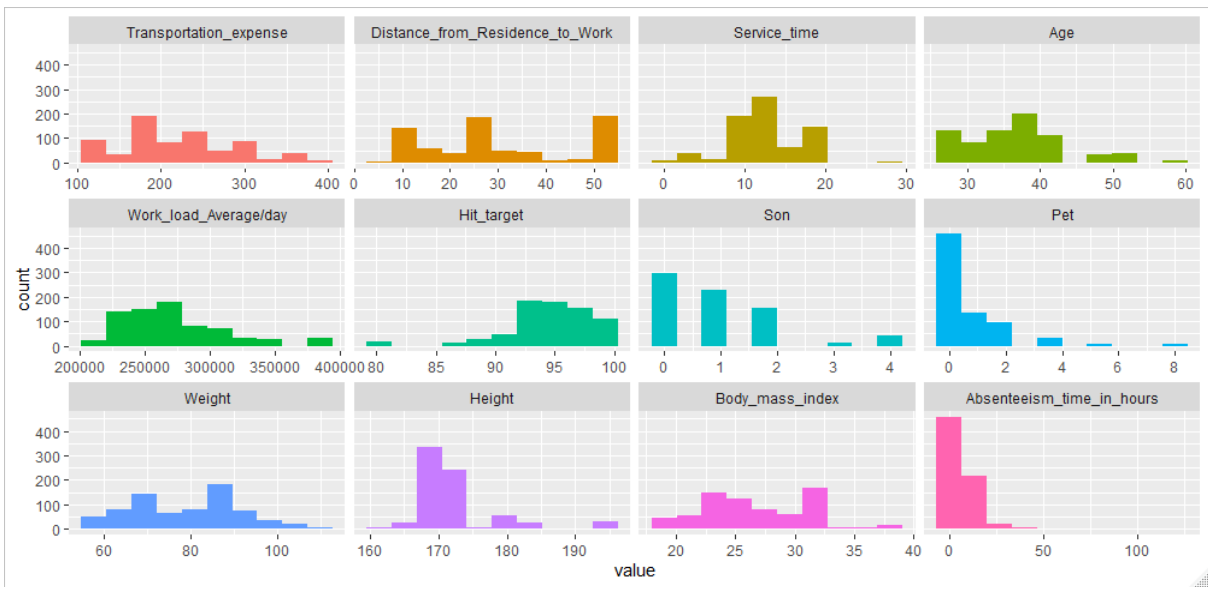
```
                 variable q_zeros p_zeros q_na p_na q_inf p_inf   type unique
1                      ID       0    0.00    0 0.00     0     0 factor     36
2      Reason_for_absence      43    5.81    3 0.41     0     0 factor     28
3       Month_of_absence        3    0.41    1 0.14     0     0 factor     13
4         Day_of_the_week       0    0.00    0 0.00     0     0 factor      5
5                 Seasons       0    0.00    0 0.00     0     0 factor      4
6 Disciplinary_failure        695   93.92    6 0.81     0     0 factor      2
7               Education       0    0.00   10 1.35     0     0 factor      4
8           Social_drinker    319   43.11    3 0.41     0     0 factor      2
9            Social_smoker    682   92.16    4 0.54     0     0 factor      2
```

Table 2.2 EDA of continuous variables

```
                         variable q_zeros p_zeros q_na p_na q_inf p_inf    type unique
1           Transportation_expense       0     0.0    7 0.95     0     0 numeric     24
2   Distance_from_Residence_to_Work      0     0.0    3 0.41     0     0 numeric     25
3                     Service_time       0     0.0    3 0.41     0     0 numeric     18
4                              Age       0     0.0    3 0.41     0     0 numeric     22
5            Work_load_Average/day       0     0.0   10 1.35     0     0 numeric     38
6                       Hit_target       0     0.0    6 0.81     0     0 numeric     13
7                              Son     295    39.9    6 0.81     0     0 numeric      5
8                              Pet     459    62.0    2 0.27     0     0 numeric      6
9                           Weight       0     0.0    1 0.14     0     0 numeric     26
10                          Height       0     0.0   14 1.89     0     0 numeric     14
11                 Body_mass_index       0     0.0   31 4.19     0     0 numeric     17
12         Absenteeism_time_in_hours    36     4.9   22 2.97     0     0 numeric     19
```

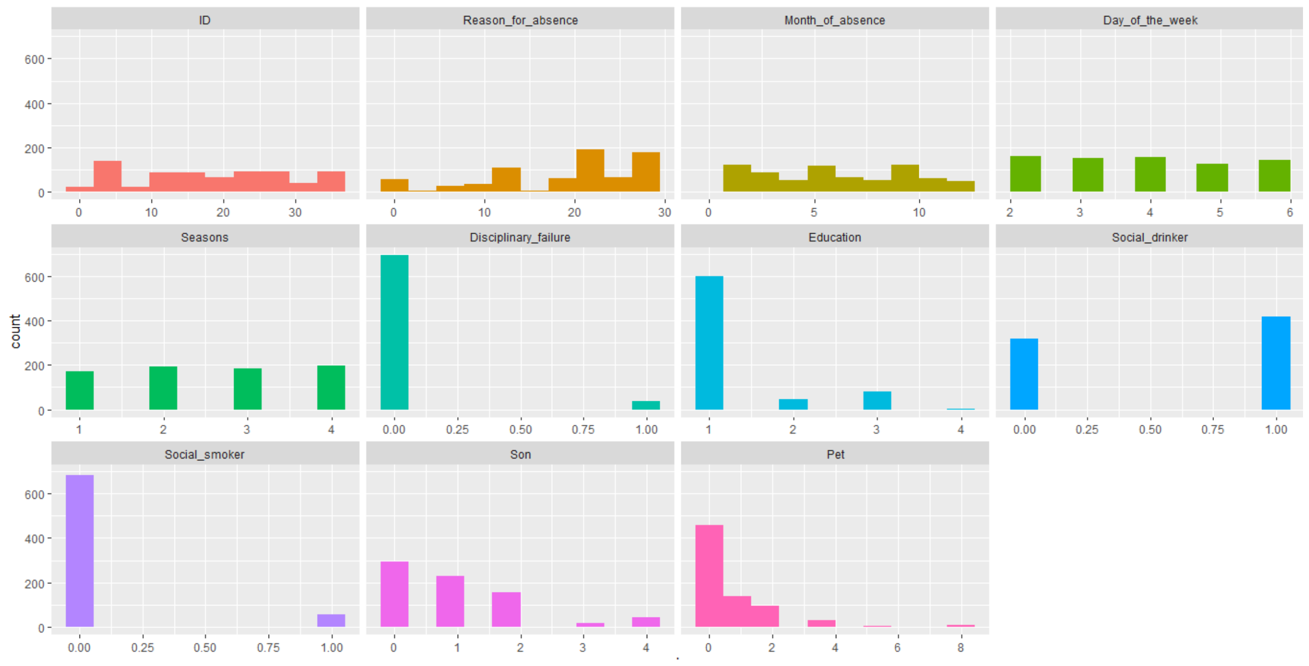
{ q_zeros: quantity of zeros, p_zeros: percentages of zeros, q_na: quantity of NA values (missing values)
p_na: percentages of missing values, type: type of variable, unique: number of unique values}

Figure 2.1.a Histogram plots of continuous variables



(Note: Here NA values are not considered for histogram plots)

Figure 2.1.b Bar-plots of Categorical Variables



<u>Inferences:</u>

- Most of the employees are in the age range of 25 to 35 years.
- Half of the employee are having BMI greater than 25 which is considered as overweight.
- More than half of the employees have zero hour of absenteeism and have no pets.
- Target variable ('Absenteeism time in hours') contains missing values which are to be removed before imputation and further analysis.
- None of the distributions of continuous variable is normal distribution.

<u>Bivariate Analysis</u>: Here, two variables are studied together for their empirical relationship and association of two variables is studied. It helps in feature selection and predictions and also helps in detecting anomalies in the data. This type of analysis can be categorized based on relation between variables as follows:

- Continuous -continuous variable: Both considered variables for analysis are of continuous type. Scatter plot and correlation plot are used for this kind of analysis
- Categorical-Continuous variable: Here one variable is categorical in nature and another is continuous in nature. Bar plot and two sample t-test are used for such analysis
- Categorical-Categorical variable: Both variables are of categorical in nature. Two way table and chi-squared test are used for such type of analysis. This also helps in feature selection part.

2.1.2 <u>Missing Value Analysis:</u>

All the variables from the dataset except 'ID', 'Seasons', 'Day of week' have missing values. Many machine learning algorithms fail to work properly if data contains missing values. Thus this step is important part of pre-processing.

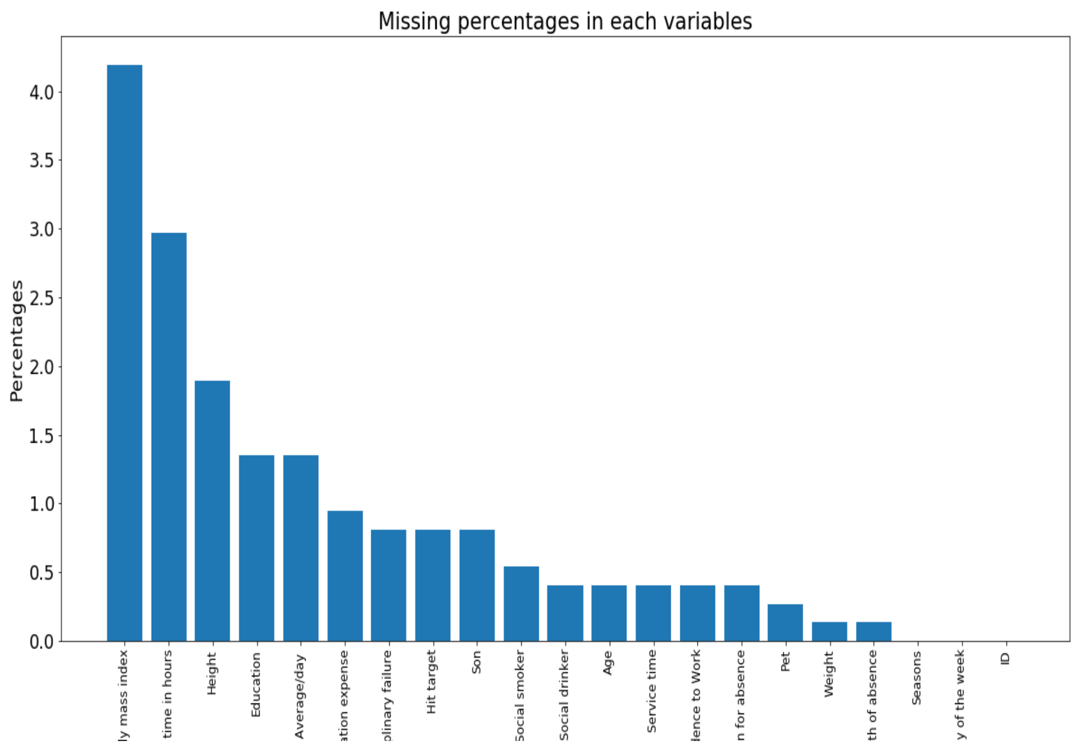


Fig 2.2 Missing values percentages in each variable

❖ Types of Missing Values:

- Missing Completely at Random (MCAR): Missing values have no relation to the variable in which missing values exists. Also they don't have any relation to other variables in dataset.
    - From the dataset, 'Reason for absence', 'Month of absence', 'Work load Average/day', 'Hit target', 'Disciplinary failure' , 'Absenteeism time in hours' have completely random missing values.
- Missing at Random (MAR): Missing values have no relation to the variable in which missing values exist. But they do have a relation with other variable.
    - Here in the given data, as "ID" column represent the individual employee id and thus 'Education', 'Son', 'Social drinker', 'Social smoker', 'Pet', 'Weight', 'Height', 'Body mass index',' Transportation expense', 'Distance from Residence to Work', 'Service time', 'Age' columns' missing values are dependent on "ID" column values as these values should be same for individual employee.
- Missing not at Random (MNAR) (ie in a pattern): Missing values have relation to the variable in which missing values exists.

❖ Methods to deal with missing values:

- Imputation:

  For <u>continuous variable</u>: use mean, median and regression methods.

For categorical variable: use of mode or classification methods. (As mode represents the highest frequency element and categorical variables have categories. So, replacing the missing values with the mode of these categories is preferred for categorical variables.)
Using ML models: Using KNN imputation by setting proper value of ‘k’ for better imputations for both type of variables. It should be noted here that for categorical variable value of ‘k’ should be odd always.

The assumption behind using KNN for missing values is that a point value can be approximated by the values of the points that are closest to it, based on other variables. The “fancyimpute” KNN algorithm works by calculating the k nearest neighbours which have the missing features available and then weights them based on Euclidean distance from the target row. The missing value is then calculated as a weighted mean from these neighbouring rows. However, this isn't a general implementation. We also ignore the possibility that both of the closest neighbours can be on the same side to reduce the complexity of the code.

Also multiple imputation is a good way to deal with MAR type missing values. ‘Multiple Imputation with Chained Equations’ (MICE) approach was used for multiple imputation. In R, “mice” library helps in implementing this algorithm whereas in python multiple imputation can be done by using mice method from “impyte” library.

- Deletion of missing values: (This leads to loss of data. So it’s always better to stick with imputation method)

  Row wise deletion

  Column wise deletion

  Pairwise deletion

Table 2.3 Missing value imputation

Imputation Method	Original Value	Imputed Value
Mean	**179**	221
Median		225
KNN imputation (k=5)		**181**
Multiple imputation (mice)		197

(Note: Value of “dataset['Transportation expense'].iloc[70] “ is considered from the dataset as original value.)

In case of given dataset, KNN gave best imputed value in comparison to other methods. So **KNN imputation was considered for missing value imputation (**KNN was selected in both R and Python implementation as it gave the best results**).**

2.1.3 Outlier Analysis:

Mean is most affected by outliers and outliers will not affect median, mode.

Graphical method of detection: Boxplot

Formula: If a value satisfies following condition, then it’s an outlier

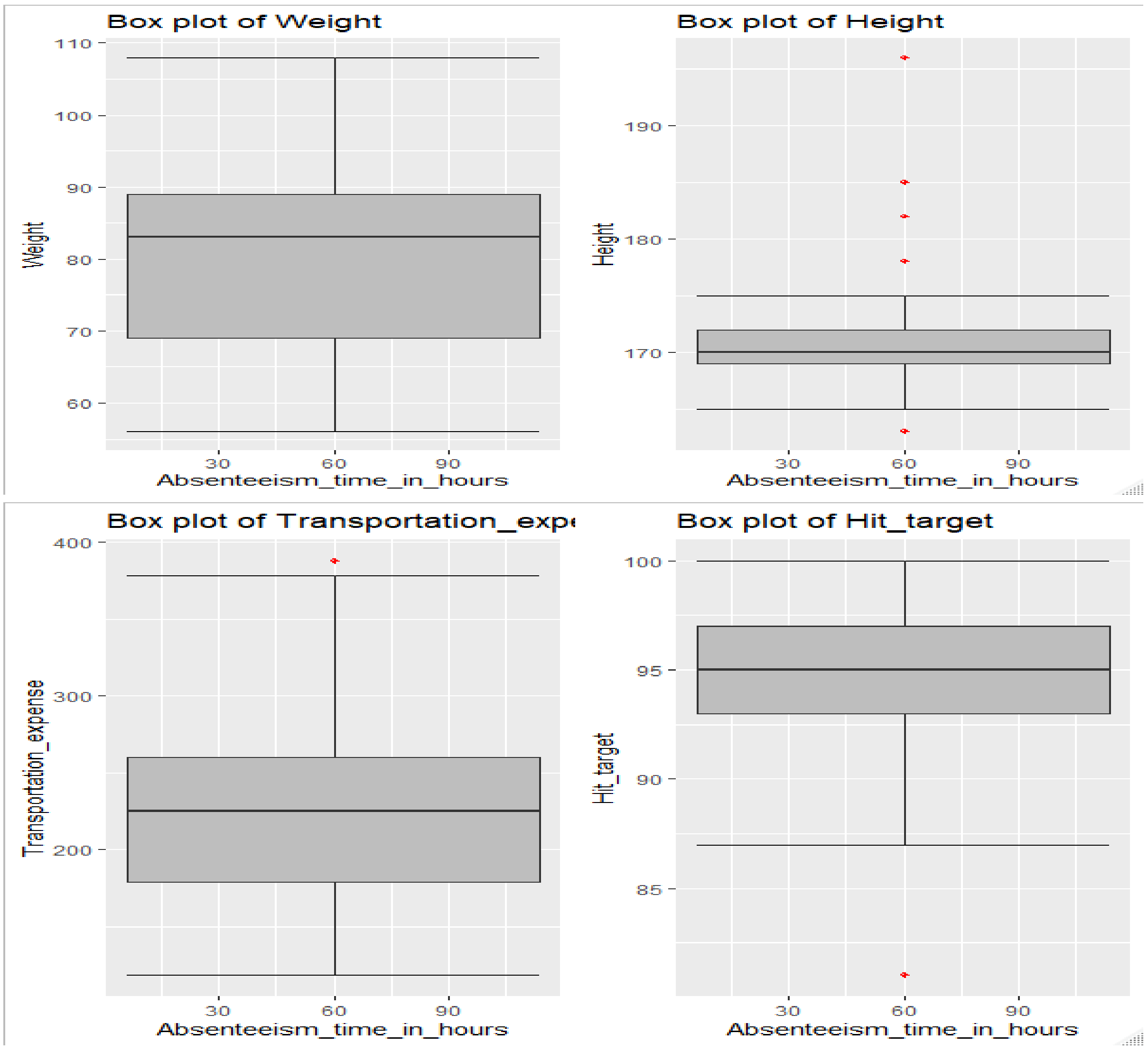
$$((Q_1 - 1.5 * IQR) > Value) \text{ or } (Value > (Q_3 + 1.5 * IQR))$$

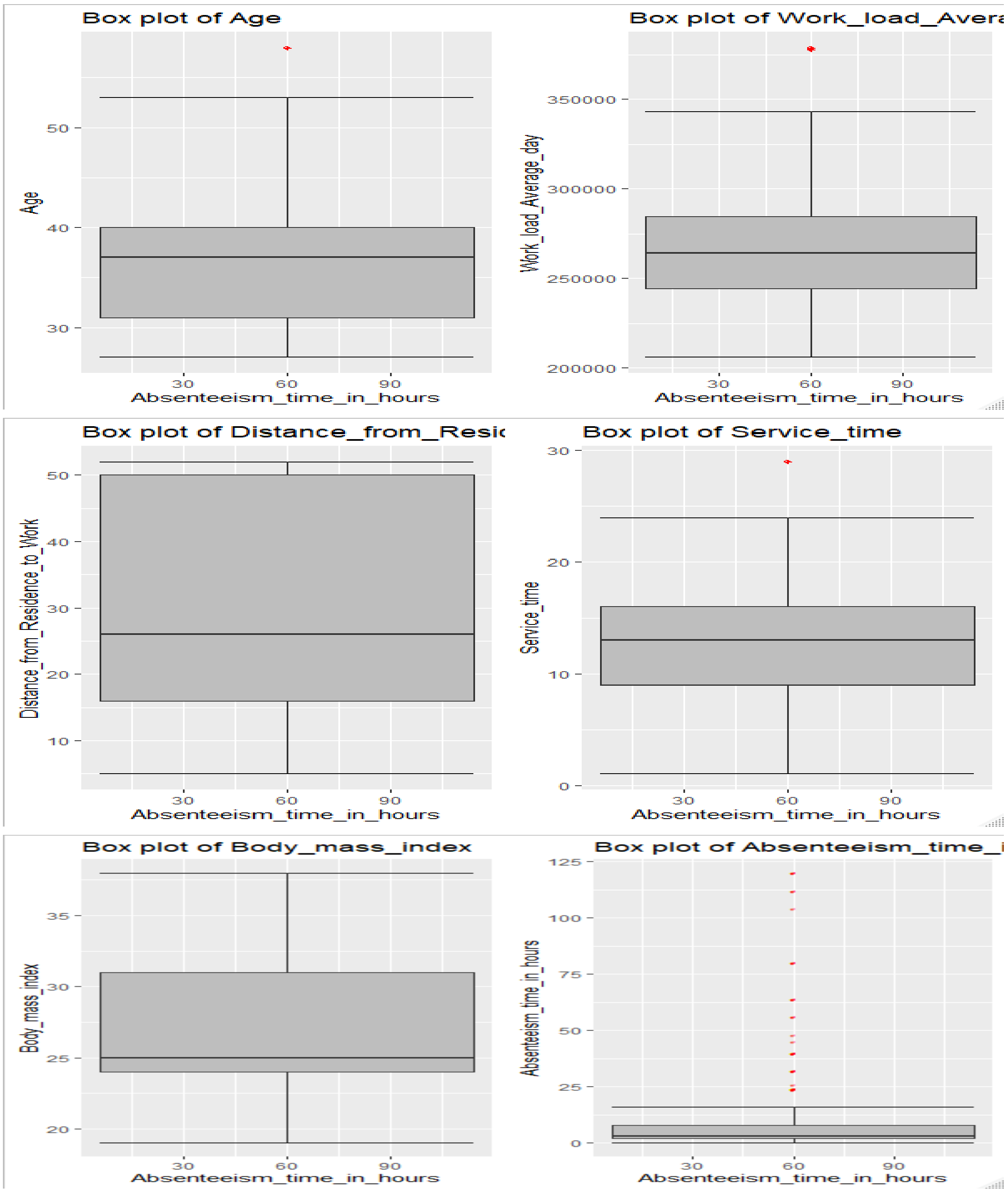
Where IQR=Q3-Q1 (Inter Quartile Range)

Treating Outliers: There are multiple ways for treatment of outliers

1. Deleting these outliers
2. Transforming and Binning these outliers (like taking log() of value)
3. Imputing outliers similar to missing value
4. Treat them separately. (Perform diff operations on these sets of variable and diff operations on remaining sets of variable.
5. Replace the outliers with cut-off values of the variables.

Fig 2.3 Boxplots of continuous variables





In case of given problem statement, out of 11 continuous predictors, 'Distance from Residence to Work', 'Weight', 'Body mass index' predictors don’t have any outliers. Outliers were visualized using boxplots of each continuous predictors.

2.1.4 Feature Selection:

Variable Importance is crucial in ML modelling where a subset of relevant features/variables are selected for use of model construction. Feature selection is done to avoid over-fitting, to make fast predictions and training, to decrease storage required for model and the dataset. Two techniques for feature selection:

1. Domain Knowledge
2. ML algorithm's usage.

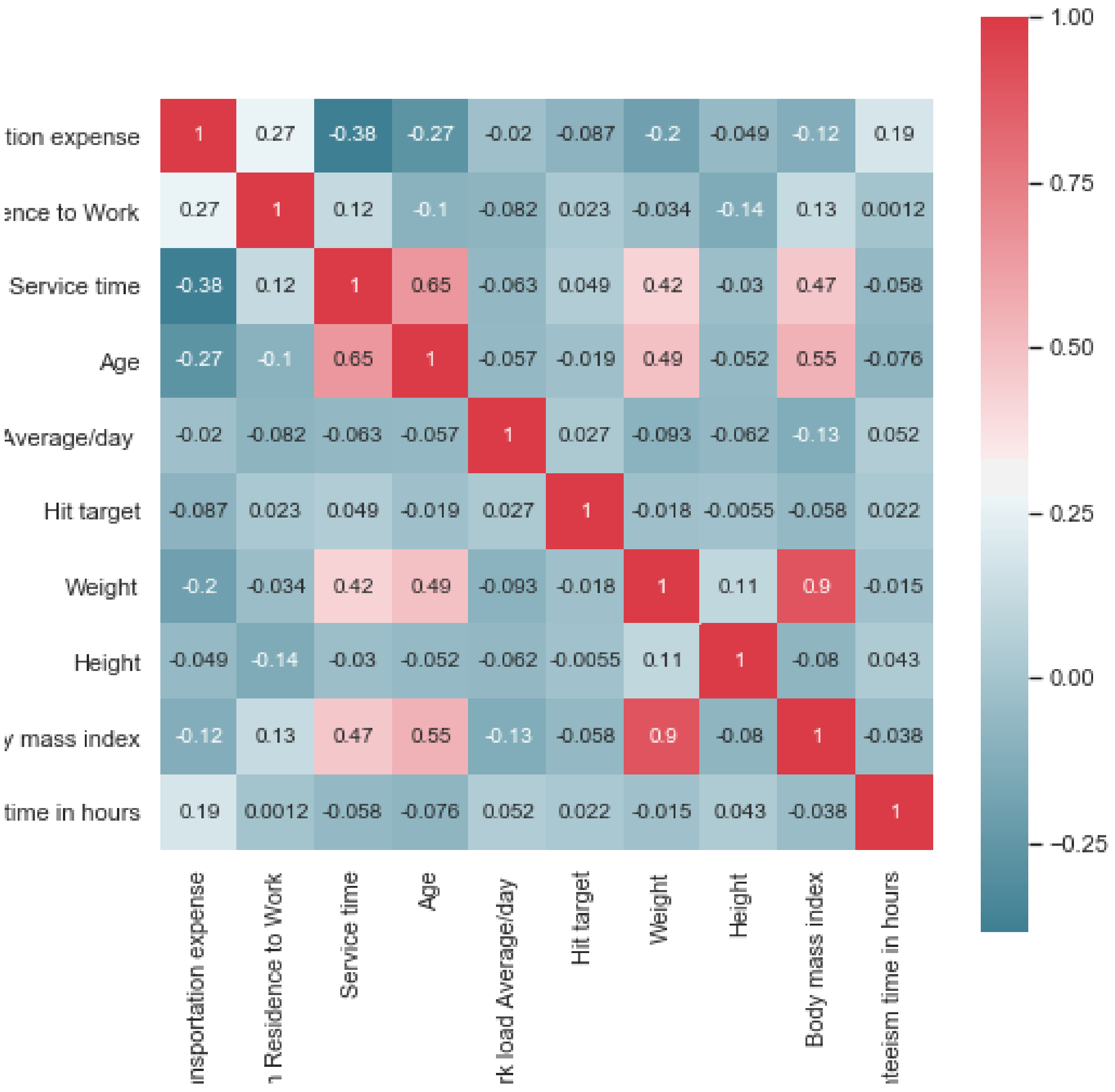


Fig 2.4 Correlation plot of continuous variables

Here correlation analysis helped in dropping the highly correlated continuous variables (i.e. "Body mass index" column).Also ANOVA test was used to check dependencies of categorical variable with the continuous target variable. ANOVA uses one categorical and one numerical variable to calculate the relevancy of that particular variable. Using the probability value generated by ANOVA test, those variables which were having p value less than 0.05 used as features for prediction. In case of implementation of ANOVA in R programming, even though columns such as Education, Seasons, Month of absence have statistically high p values, they were still considered as features because of logical importance of these features in prediction.

Following is the list of variables (features) selected for model construction:

1) ID
2) Reason for absence
3) Month of absence
4) Day of the week
5) Seasons
6) Transportation expense
7) Distance from Residence to Work
8) Service time
9) Age
10) Work load average/day
11) Hit target
12) Disciplinary failure
13) Education
14) Son
15) Social drinker
16) Social smoker
17) Pet
18) Weight
19) Height

2.1.5 Feature Scaling:

Performed only for continuous/numeric variables as there's a need to scale down these variables to same range of values.

Two ways to do this:

Normalization:

This will bring all the data in the range of zero to one [0, 1]. It's the process of reducing unwanted variation within either inside variables or between variables. It's nothing but bringing every value to same range. It is sensitive to outliers. So this process should be done after outlier removal.

$$X_{new} = \frac{(X_o - X\_\min)}{(X\_\max - X\_\min)}$$

Standardization: (Z-Score)

Works well if the data is uniformly distributed

$$Z = \frac{X - \mu}{\sigma}$$

μ => mean of population

σ => std. deviation of population data.

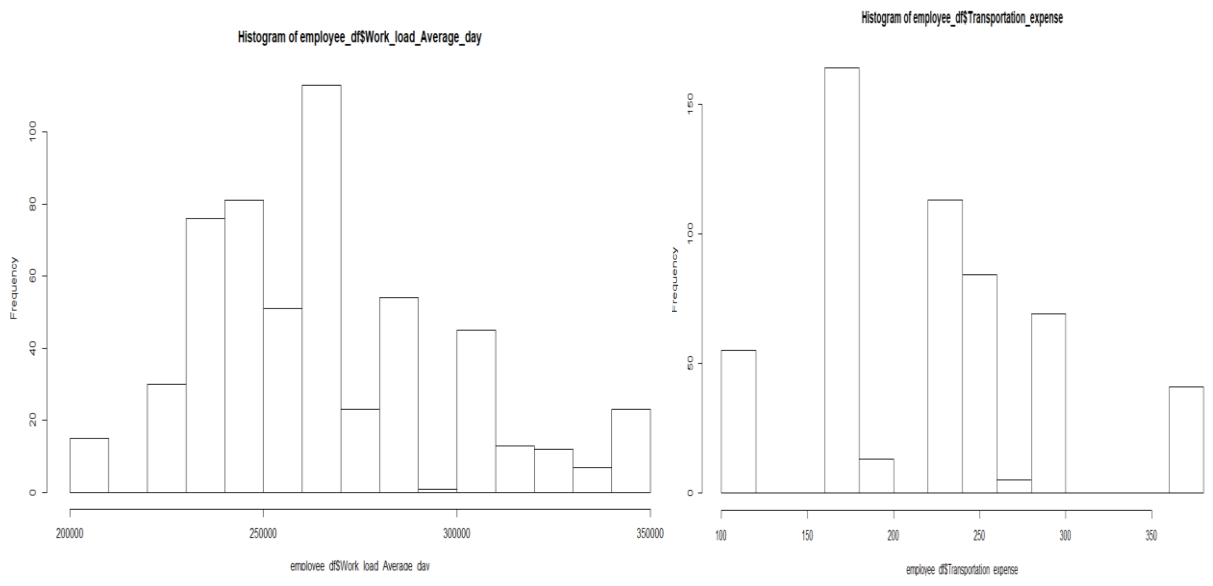
Z represents diff between raw score and population mean in the units of std. deviation

Z=> negative -----> X< μ

Z=>positive ------->X> μ

For given dataset, normalization was done for continuous variables and dummification was done in case of categorical variables. Following histograms show the difference in the range of values for two different variables.

Fig 2.5 Histogram plots some variables



2.2 Modelling:

2.2.1 Model Selection:

After pre-processing of the data, ML models are used for making predictions. Given problem is a regression problem. Thus regression based models such as linear regression, decision trees, random forest were selected to predict the target variable.

2.2.2 Linear Regression:

This algorithm is used to predict one variable using another variable when both of them are continuous in nature. It is a part of supervised learning algorithm. Linear regression is used for regression problems. R squared metric and RMSE (root mean squared error) metric will help in evaluating regression models.

Table 2.4 Evaluation metrics for Linear Regression

Linear Regression	R		Python	
	Train data	Test data	Train data	Test data
RMSE	2.1	3.0	2.41	685536704857
R squared	0.58	0.28	0.518	-5.01
MAE	1.4	2.0	1.56	79098769230

MSE	4.5	8.9	5.82	4.69

2.2.3 Decision Trees:

Decision Trees are a type of Supervised Machine Learning where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves. The leaves are the decisions or the final outcomes. And the decision nodes are where the data is split. A decision tree is a structure that describes a basic process to follow to reach a conclusion.

Table 2.5 Evaluation metrics for Decision Trees

Decision Tree	R		Python	
	Train data	Test data	Train data	Test data
RMSE	2.3	2.9	2.74	685536704857
R squared	0.53	0.3	0.38	-5.01
MAE	1.5	1.7	1.84	79098769230
MSE	5.1	8.4	7.51	4.7

2.2.3 Random Forest:

In random forest algorithm, number of decision trees created internally is decided by the error rate. It will build the trees until the error no longer decreases. Thus it is not possible to predict number of tree created in random forest algorithm if number of trees are not defined. Random Forest is an ensemble technique that consists of many decision trees. The idea behind Random Forest is to build n number of trees to have more accuracy in dataset.

Table 2.6 Evaluation metrics for Random Forest

Random Forest	R		Python	
	Train data	Test data	Train data	Test data
RMSE	1.31	2.8	1.19	5.73
R squared	0.87	0.34	0.9	0.39
MAE	0.88	1.8	0.68	1.69
MSE	1.71	7.7	1.09	2.39

Chapter 3 **Conclusion**

This chapter deals with evaluation of models and selection of best model for the given problem statement as mentioned in chapter 1.1 and also patterns were analysed from visualizations made throughout the project.

3.1 Model Evaluation:

Model evaluation is done based the values of metrics such as RMSE, MSE, MAE, R-squared value. **RMSE** (Root Mean Squared Error) is the most popular evaluation metric used in regression problems. It follows an assumption that error are unbiased and follow a normal distribution. The power of 'square root' empowers this metric to show large number deviations. It represents the sample standard deviation of the differences between predicted values and observed values (called residuals). **MAE** is the average of the absolute difference between the predicted values and observed value. The MAE is a linear score which means that all the individual differences are weighted equally in the average. The MAE is also the most intuitive of the metrics since we're just looking at the absolute difference between the data and the model's predictions. RMSE penalizes the higher difference more than MAE. Generally, RMSE will be higher than or equal to MAE. The only case where it equals MAE is when all the differences are equal or zero. It is important to note that the units of both RMSE & MAE are same. The range of RMSE & MAE is from 0 to infinity. MAE is robust to outliers whereas RMSE is not. **The coefficient of determination, or R^2** (sometimes read as R-two), is another metric we may use to evaluate a model and it is closely related to MSE, but has the advantage of being scale-free — it doesn't matter if the output values are very large or very small, the R^2 is always going to be between negative infinity and 1. The absolute value of RMSE does not actually tell how bad a model is. It can only be used to compare across two models whereas R^2 easily does that. **R-squared** is a relative measure of fit, **RMSE** is an absolute measure of fit. Theoretically, if a model has adjusted R^2 equal to 0.05 then it is definitely poor. The maximum value of R^2 is 1 but minimum can be negative infinity. **MSE** basically measures average squared error of our predictions. For each point, it calculates square difference between the predictions and the target and then average those values. The higher this value, the worse the model is. It is never negative, since we're squaring the individual prediction-wise errors before summing them, but would be zero for a perfect model.

3.2 Model Selection:

The "**Random Forest**" model has best set of evaluation metric when compared with other models, and so it was chosen for modelling. Also, over-fitting will be less as evaluation parameters for both test and train do not differ much when compared.

3.3 Observations:

1. It was observed from the interactive plot that person having employee ID was absent for maximum amount of time. The courier company must consult with that employee.

2. People having education till high school were absent for most of the time. Company should help its employees by sponsoring educational and professional courses.
3. More than 50% employee who were absent had drinking habit. Company should put some efforts for increasing fitness of employees by giving bonus or membership to fitness centre or sport centre.
4. Most of the employee who don't smoke had no disciplinary failure.
5. Main reason for absence (about 45 % from all) for the most of the employee were: Medical consultation, Dental consultation and Physiotherapy. Thus employees prefer health issues to be cured within the time.

Chapter 4 Appendices

4.1 Appendix A- Extra Figures:

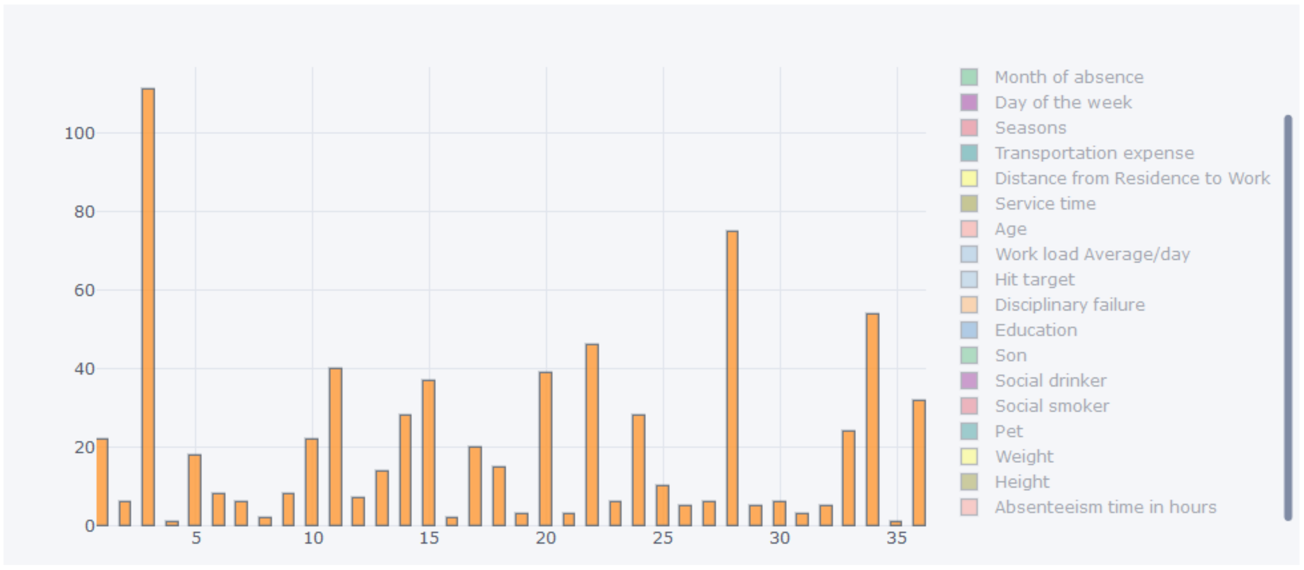


Fig 4.1 Histogram for ID column

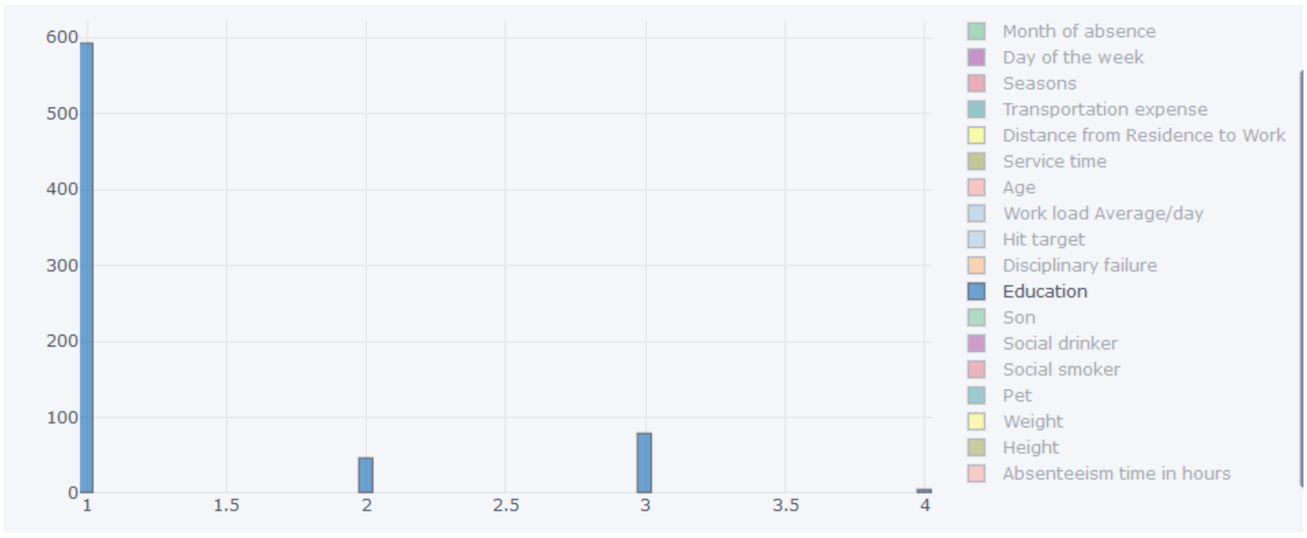


Fig 4.2 Histogram for Education column

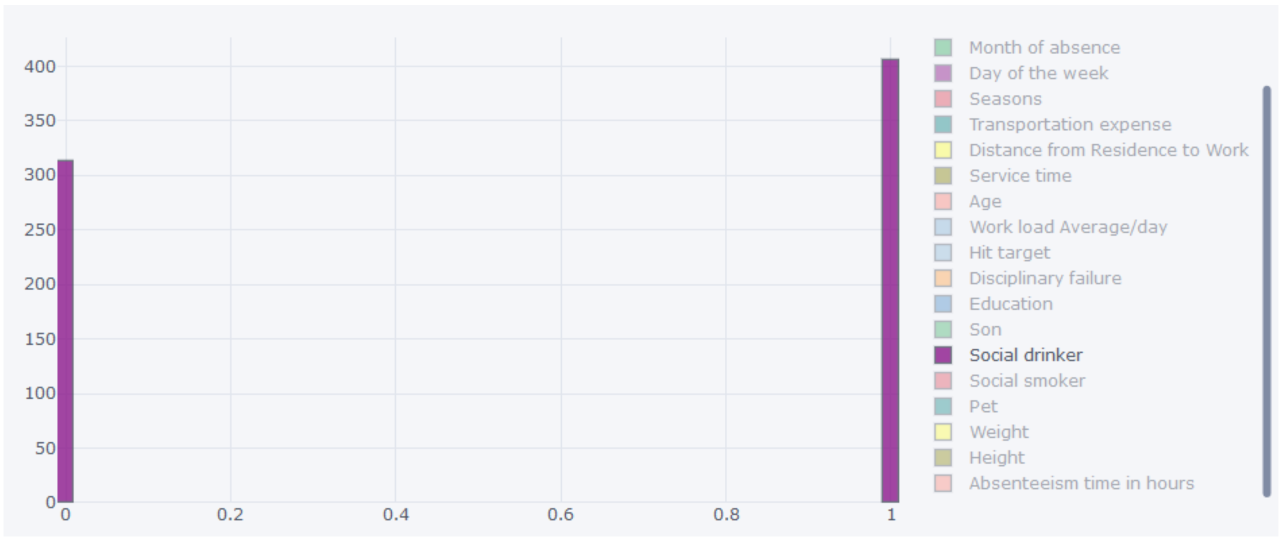


Fig 4.3 Histogram for Social Drinker

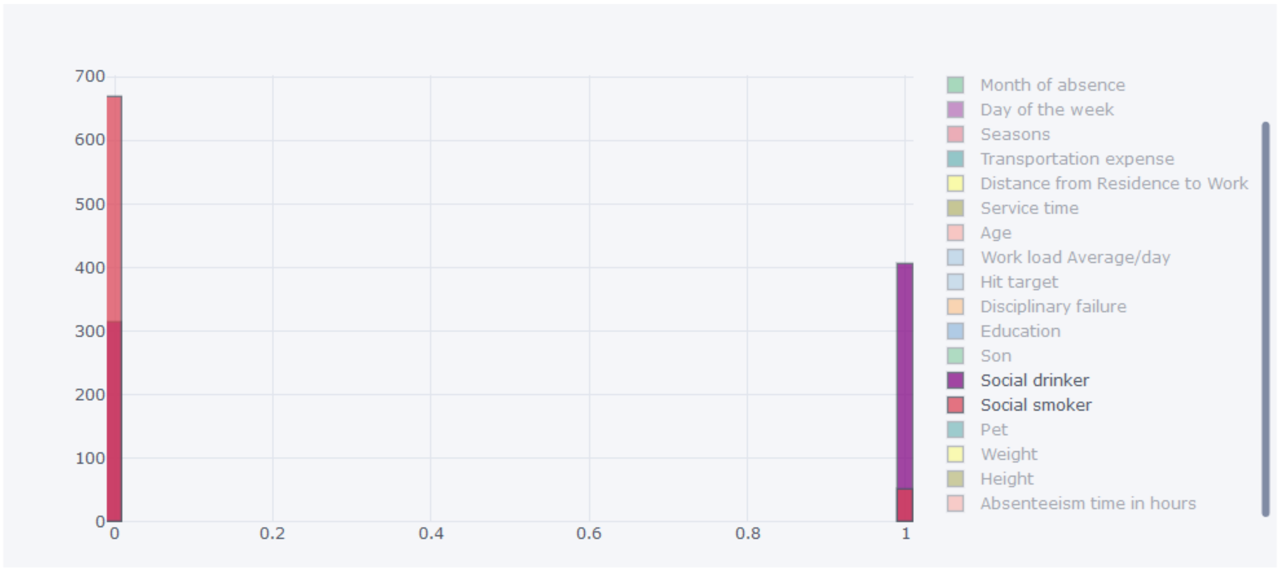


Fig 4.4 Combined histogram plot for Social drinker and social smoker

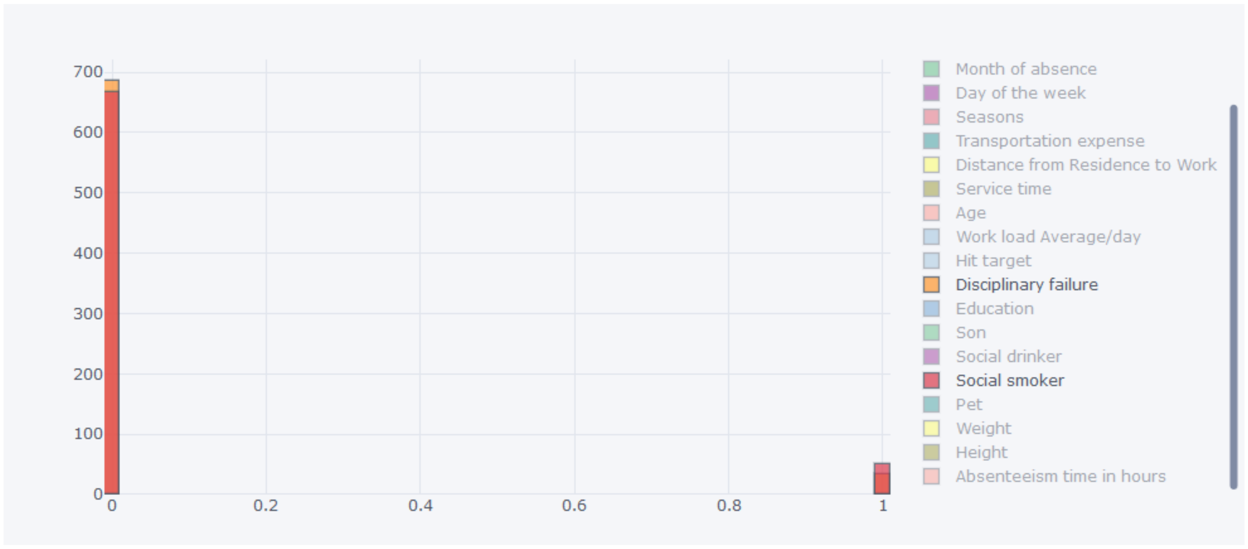


Fig 4.5 Combined histogram plot for Disciplinary failure and social smoker

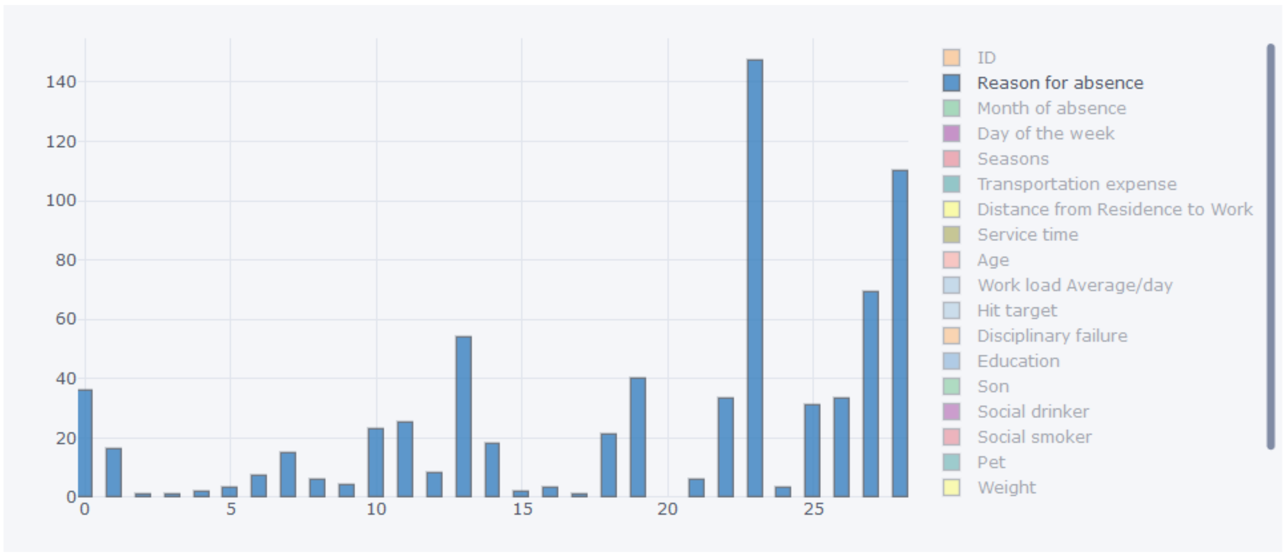


Fig 4.5 Histogram for Reason for absence

4.2 Appendix B- Codes:

4.2.1 Python Code:

```
#import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
import os
import fancyimpute
from fancyimpute import KNN
from impyute.imputation.cs import mice
from scipy.stats import chi2_contingency
from scipy import stats
from sklearn.model_selection import cross_val_score

#os.getcwd()
os.chdir(r"C:\Users\ELdrago\Desktop\Edwisor\Projects\Employee Absenteeism\Python_Code")
print(os.listdir("../Python_Code"))
# load the data
df=pd.read_excel("Absenteeism_at_work_Project.xls")
dataset=df.copy()
dataset.shape

# saving target variable name for future usage
target_var='Absenteeism time in hours'
```

```
# Exploratory Data Analysis
dataset.head(5)
dataset.columns
dataset.nunique()
# get the structure of the data
print(dataset.info())
dataset.describe()
#changing datatypes of some variables
dataset["ID"]=dataset["ID"].astype("object")
dataset["Reason for absence"]=dataset["Reason for absence"].astype("object")
dataset["Month of absence"]=dataset["Month of absence"].astype("object")
dataset["Day of the week"]=dataset["Day of the week"].astype("object")
dataset["Seasons"]=dataset["Seasons"].astype("object")
dataset["Education"]=dataset["Education"].astype("object")
dataset["Social drinker"]=dataset["Social drinker"].astype("object")
dataset["Social smoker"]=dataset["Social smoker"].astype("object")
dataset["Disciplinary failure"]=dataset["Disciplinary failure"].astype("object")
dataset["Pet"]=dataset["Pet"].astype("object")
dataset["Son"]=dataset["Son"].astype("object")
# separating continuous and categorical variables
categorical_data = dataset.select_dtypes(include=['object']).copy()
continuous_data = dataset.select_dtypes(include=['int64','float64']).copy()
cat_vars=categorical_data.columns
cont_vars=continuous_data.columns
dataset.describe(include=['object', 'bool'])
#barplots
for name in cat_vars:
        plt.figure(figsize=(14,5))
        dataset[name].value_counts(dropna=False, normalize=True).sort_index().plot.bar()
        plt.ylabel("Frequency")
        plt.title(name)
        plt.show()
#distribution plots
for i in cont_vars:
        if i ==target_var:
          continue
        sns.distplot(dataset[i].dropna(),bins = 'auto')
        plt.title("Distribution for "+str(i))
        plt.ylabel("Frequency")
        plt.show()
dataset_groupby_ID=dataset.groupby(by=["ID"])
for i in sorted(list(dataset["ID"].unique())):
```

```
                print(i, dataset_groupby_ID['Distance from Residence to Work'].get_group(i).unique(), sep=',')
```
# **Missing Value Analysis**
```
def show_missing_values(dataset):
                # checking missing values
                missing_values=pd.DataFrame(dataset.isnull().sum().sort_values(ascending= False))
                missing_values
                #reseting index
                missing_values=missing_values.reset_index()
                missing_values
                #renaming the column names of the dataframes
                missing_values= missing_values.rename(columns={'index':'Variables',0:'missing_percentage'})
                missing_values
                #calcualting % of missing values
                missing_values['missing_percentage']=(missing_values['missing_percentage']/len(dataset))*100
                missing_values.to_csv("Missing_value_perc.csv", index = False)
                #plotting
                plt.figure(figsize=(20,10))
                plt.tick_params(axis='both', which='minor', labelsize=12)
                index=np.arange(len(missing_values.Variables))
                plt.bar(missing_values.Variables,missing_values.missing_percentage)
                plt.title("Missing percentages in each variables")
                plt.ylabel("Percentages")
                plt.xticks(index,missing_values.Variables, fontsize=12, rotation=90)

                #print(missing_values)
   return missing_values
show_missing_values(dataset)
plt.savefig("missing_values.png")
# missing values in categorical data
show_missing_values(categorical_data)
plt.savefig("missing_values_categorical.png")
# missing values in continuous data
show_missing_values(continuous_data)
plt.savefig("missing_values_continuous.png")

# Droping observation in which "Absenteeism time in hours" has missing value
dataset = dataset.drop(df[df[target_var].isnull()].index, axis=0)
#sorting the dataset by employee ID
dataset=(dataset.sort_values(by="ID")).reset_index().drop(columns='index')
dataset.shape
# creating a NaN value for testing purpose
dataset['Transportation expense'].iloc[70]=np.nan
```

```
#KNN imputaion of the variable
dataset=pd.DataFrame(KNN(k=5).fit_transform(dataset),columns=dataset.columns)
print(dataset['Transportation expense'].iloc[70])
dataset.isna().sum()

# Covert the categorical data in appropriate data type
for i in cat_vars:
        dataset.loc[:,i]=dataset.loc[:,i].round()
        dataset.loc[:,i]=dataset.loc[:,i].astype('object')

# Outlier Analysis
for name in cont_vars:
        if name in ['Absenteeism time in hours']:
          continue
        plt.figure(figsize=(8,5))
        plt.boxplot(dataset[name])
        plt.xlabel(name)
        plt.ylabel("values")
        plt.title("Boxplot of "+ str(name))
        plt.show()
# list of continuous variables not having any outliers
ignore=['Distance from Residence to Work','Weight','Body mass index']
#detect and replace the outlier from the data with NaN
def remove_outliers(dataset,cont_names,ignore):
        for i in cont_names:
                if i in ignore:
                        continue
                q75,q25=np.percentile(dataset[i],[75,25])
                iqr=q75-q25 #inter quartile range
                minimum=q25-(iqr*1.5)
                maximum=q75+(iqr*1.5)
                # Replacing all the outliers value to NA
                dataset.loc[dataset[i]< minimum,i] = np.nan
                dataset.loc[dataset[i]> maximum,i] = np.nan
   return print("Outliers Removed")
remove_outliers(dataset,cont_vars,ignore)
#imputing outliers
dataset = pd.DataFrame(KNN(k = 3).fit_transform(dataset), columns = dataset.columns)
#checking missing values
dataset.isna().sum()
```

# Feature Selection

```
# a) Continuous variables
```

```
df_corr=dataset.loc[:,cont_vars]
#Generate correlation matrix
corr = df_corr.corr()
#Set the width and hieght of the plot
f, ax = plt.subplots(figsize=(8, 8))
#Plot using seaborn library
sns.heatmap(corr, mask=np.zeros_like(corr, dtype=np.bool), cmap=sns.diverging_palette(220, 10,
as_cmap=True), square=True, ax=ax, annot=True)
plt.savefig("correlation_plot.png")
# b) Categorical variables (Use of ANOVA test as target variable is continuous in nature
for i in cat_vars:
        f, p = stats.f_oneway(dataset[i], dataset[target_var])
        print("P value for the variable '"+str(i)+"' is "+str(p))
#removing unnecessary features
dataset=dataset.drop(to_be_removed, axis = 1)
cleaned_data=dataset.copy()
cleaned_data.to_csv("cleaned_data.csv", index=False)
```

# **Feature Scaling**

```
# Normalization
for i in cont_vars:
        if i == target_var:
                continue
        print(i)
        dataset[i] = (dataset[i] - min(dataset[i]))/(max(dataset[i]) - min(dataset[i]))
# Dummification
for i in cat_vars:
        temp=pd.get_dummies(dataset[i],prefix=i)
        dataset=dataset.join(temp)
print(dataset.shape)
```

#**Interactive Data Visualization using plotly**

```
# import required libraries
import plotly.plotly as py
import plotly.graph_objs as go
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
import cufflinks as cf
# For Notebooks setup
cf.go_offline(connected=True)
init_notebook_mode(connected=True)
# Histogram plot of
cleaned_data.iplot(kind='hist',y=target_var,bins=100)
```

```
#Model Development
# splitting dataset
from sklearn.model_selection import train_test_split
X=dataset.loc[:, dataset.columns != target_var]
y= dataset.loc[:, target_var]
X_train, X_test, y_train, y_test =  train_test_split( X, y, test_size = 0.2)

#Error Metrics
# Model Evaluation
#import the required library and module
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
# define a function to get error metrics
def get_error_metric(original,predicted):
        MSE=mean_squared_error(original,predicted)
        # root mean squred error
        RMSE=np.sqrt(MSE)
        #mean absolute error
        MAE=mean_absolute_error(original,predicted)
        # R square value
        r2=r2_score(original,predicted)
        results={'MSE':MSE,'RMSE':RMSE,'MAE':MAE,'R^2 score':r2}
        return results

#Linear Regression
# Importing libraries for Linear Regression
from sklearn.linear_model import LinearRegression
# Building model on top of training dataset
fit_LR = LinearRegression().fit(X_train , y_train)
# model fitting on train data
pred_train = fit_LR.predict(X_train)
# mdoel fitting on test data
pred_test = fit_LR.predict(X_test)
# check error metrics
print("Error Metrics for train data")
print(get_error_metric(y_train,pred_train))
print("Error Metrics for test data")
print(get_error_metric(y_test,pred_test))

# Decision Tree
# Importing libraries for Decision Tree
from sklearn.tree import DecisionTreeRegressor
```

```
# Building model on training dataset
fit_DT = DecisionTreeRegressor(max_depth =2).fit(X_train,y_train)
# model fitting on train data
pred_train = fit_DT.predict(X_train)
# model fitting on test data
pred_test = fit_LR.predict(X_test)
# check error metrics
print("Error Metrics for train data")
print(get_error_metric(y_train,pred_train))
print("Error Metrics for test data")
print(get_error_metric(y_test,pred_test))

# Random Forest
# Importing libraries for Decision Tree
from sklearn.ensemble import RandomForestRegressor
# Building model on training dataset
fit_RF = RandomForestRegressor(n_estimators = 500).fit(X_train,y_train)
# model fitting on train data
pred_train = fit_RF.predict(X_train)
# model fitting on test data
pred_test = fit_RF.predict(X_test)
# check error metrics
print("Error Metrics for train data")
print(get_error_metric(y_train,pred_train))
print("Error Metrics for test data")
print(get_error_metric(y_test,pred_test))
```

4.2.2 R Code:

```
rm(list = ls())
#path="C:/Users/ELdrago/DesktopEdwisor/Projects/Employee Absenteeism"
#setwd(path)



#-------load required libraries
x=c("tidyverse", "corrgram", "DMwR", "caret", "randomForest", "unbalanced",
   "C50", "dummies", "e1071", "Information","MASS", "rpart", "gbm", "ROSE",'tidyverse','Hmisc','funModeling',
   'sampling', 'DataCombine', 'inTrees','readxl','mice','missMDA','data.table','lsr')
lapply(x,require,character.only= TRUE)
rm(x)

#load the dataset
employee_df=read_xls("C:/Users/ELdrago/Desktop/Edwisor/Projects/Employee
Absenteeism/R_Code/Absenteeism_at_work_Project.xls")
```

```
employee_df=as.data.frame(employee_df)
View(employee_df)
str(employee_df)
#------------------------------------------
#-------------------1)Exploratory Data Analysis

#replacing whitespaces from the column names with underscore (_) for simplicity
colnames(employee_df)<-gsub(" ","_",colnames(employee_df))
colnames(employee_df)<-gsub("/","_",colnames(employee_df))
colnames(employee_df)

cont_vars = c('Distance_from_Residence_to_Work', 'Service_time', 'Age',
              'Work_load_Average_day', 'Transportation_expense',
              'Hit_target', 'Weight', 'Height',
              'Body_mass_index', 'Absenteeism_time_in_hours')

cat_vars = c('ID','Reason_for_absence','Month_of_absence','Day_of_the_week',
              'Seasons','Disciplinary_failure', 'Education', 'Social_drinker',
              'Social_smoker', 'Son', 'Pet')
categorical_data=subset(employee_df,select=cat_vars)
continuous_data=subset(employee_df,select=cont_vars)
target_var='Absenteeism_time_in_hours'

# Univariate Analysis
num_eda <- function(data)
{
  glimpse(data)
  df_status(data)
  profiling_num(data)
  plot_num(data)
  describe(data)
}
cat_eda <- function(data)
{
  glimpse(data)
  df_status(data)
  freq(data)
  plot_num(data)
  describe(data)
}

cat_eda(categorical_data)
num_eda(numeric_data)
```

```
#-----------------2) Missing Value Analysis

show_missing_value<-function(dataset){
# creating dataframe with missing percentage
missing_val=data.frame(apply(dataset,2,function(x) {sum(is.na(x))}))
#convert row names into column
missing_val$columns=row.names(missing_val)
row.names(missing_val)= NULL

# renaming first variable name(column name)
names(missing_val)[1]="Missing_percentage"
#calculate percentage
missing_val$Missing_percentage=(missing_val$Missing_percentage/nrow(dataset))*100

# arrange in descending order
missing_val=missing_val[order(-missing_val$Missing_percentage),]
#rearranging the columns
missing_val=missing_val[,c(2,1)]
#saving this dataframe on disk
write.csv(missing_val,"Missing_val.csv",row.names=F)
#View(missing_val)
#save the plot
# 1. Open jpeg file
jpeg("missing_value_plot.jpg", width = 350, height = 350)
# data visualization using bar graph plot (only top three missing percentages)
ggplot(data = missing_val, aes(x=reorder(columns, -Missing_percentage),y = Missing_percentage))+
  geom_bar(stat = "identity",fill = "red")+xlab("Parameter")+
  ggtitle("Missing data percentage (Train)") + theme_bw()
# 3. Close the file
dev.off()
}

show_missing_value(employee_df)
#-----ordering data by column 'ID'
#employee_df=(employee_df[order(employee_df$ID),])
temp_data=copy(employee_df)
#View(temp_data)
employee_df=copy(temp_data)
#----pattern of missing values
md.pattern(employee_df)
#---------Imputation Results
```

```
# Original Value= 179
# Mean Imputation= 221
# Median Imputation= 225
# KNN (k=5) Imputation= 179
#------Imputation of missing values-------------------
# creating a test element
employee_df$Transportation_expense[170]=NA
# 1) imputation using mean of variable
#employee_df$Transportation_expense[is.na(employee_df$Transportation_expense)] =
mean(employee_df$Transportation_expense, na.rm = T)
#employee_df$Transportation_expense[170]
# 2) imputation using median of the variable
#employee_df$Transportation_expense[is.na(employee_df$Transportation_expense)] =
median(employee_df$Transportation_expense, na.rm = T)
#employee_df$Transportation_expense[170]
# 3) KNN imputation
employee_df = knnImputation(employee_df, k = 5)
employee_df$Transportation_expense[170]
# 4) MICE imputation
#employee_df <- mice(employee_df,m=5,maxit=50,meth='pmm',seed=500)
#employee_df$Transportation_expense[70]
# Checking for missing value
sum(is.na(employee_df))

#---------------3) Outlier Analysis
for (i in 1:length(cont_vars))
 {
  assign(paste0("gn",i), ggplot(aes_string(y = (cont_vars[i]), x = target_var), data = subset(employee_df))+
           stat_boxplot(geom = "errorbar", width = 0.5) +
          geom_boxplot(outlier.colour="red", fill = "grey" ,outlier.shape=18,
                       outlier.size=1, notch=FALSE) +
           theme(legend.position="bottom")+
           labs(y=cont_vars[i],x=target_var)+
          ggtitle(paste("Box plot of responded for",cont_vars[i])))
}
# plotting plots together
gridExtra::grid.arrange(gn1,gn2,ncol=2)
gridExtra::grid.arrange(gn3,gn4,ncol=2)
gridExtra::grid.arrange(gn5,gn6,ncol=2)
gridExtra::grid.arrange(gn7,gn8,ncol=2)
gridExtra::grid.arrange(gn9,gn10,ncol=2)
# removing outliers using boxplot method
df=copy(employee_df) # for backup
```

```
for(i in cont_vars){
   print(i)
   val=employee_df[,i][employee_df[,i] %in% boxplot.stats(employee_df[,i])$out ]
   employee_df=employee_df[which(!employee_df[,i] %in% val),]
}
str(employee_df)
# replacing outliers with NA and use KNN imputation
for(i in cont_vars){
    print(i)
    val=employee_df[,i][employee_df[,i] %in% boxplot.stats(employee_df[,i])$out ]
    employee_df[,i][(employee_df[,i] %in% val)]= NA
}
sum(is.na(employee_df))
#KNN imputation
employee_df=knnImputation(employee_df, k=5)

#----------------4) Feature Selection
# Correlation plot
jpeg("correlation_plot.jpg", width = 350, height = 350)
corrgram(employee_df[,cont_vars],order=F,upper.panel = panel.pie,text.panel = panel.txt, main="Correlation
Plot")
dev.off()

# ANOVA test for categorical variables
library("lsr")
anova_test = aov(Absenteeism_time_in_hours ~ ID + Day_of_the_week + Education + Social_smoker +
Social_drinker+ Pet + Son  + Reason_for_absence + Seasons + Month_of_absence + Disciplinary_failure, data =
employee_df)
summary(anova_test)

# dimension reduction
employee_df = subset(employee_df, select = -c(Body_mass_index))
cleaned_data=copy(employee_df)
write.csv(cleaned_data,"cleaned_data.csv",row.names=F)
#----------------5) Feauture Scaling
#Histogram for normality check
hist(employee_df$Absenteeism_time_in_hours)
hist(employee_df$Work_load_Average_day)
hist(employee_df$Transportation_expense)
cont_vars=c('Distance_from_Residence_to_Work', 'Service_time', 'Age',
          'Work_load_Average_day', 'Transportation_expense',
          'Hit_target', 'Height',
          'Weight')
```

```
cat_vars=c('ID','Reason_for_absence','Disciplinary_failure',
        'Social_drinker', 'Son', 'Pet', 'Month_of_absence', 'Day_of_the_week', 'Seasons',
        'Education', 'Social_smoker')
# Normalization
for(i in cont_vars)
{
  print(i)
  employee_df[,i] = (employee_df[,i] - min(employee_df[,i]))/(max(employee_df[,i])-min(employee_df[,i]))
}

# Creating dummy variables for categorical variables
employee_df = dummy.data.frame(employee_df, cat_vars)

#-------------------MODEL DEVELOPEMENT

rmExcept("employee_df")

# splitting the dataset
#Divide data into train and test using stratified sampling method
set.seed(1234)
train.index=sample(1:nrow(employee_df),0.8*nrow(employee_df))
#train.index=createDataPartition(employee_df$Absenteeism_time_in_hours,p=.80,list=F)
train_data=employee_df[train.index,]
test_data=employee_df[-train.index,]
# r squared evaluation
rsq <- function(x, y) summary(lm(y~x))$r.squared

#---------------------- 1) Linear Regression
set.seed(1234)
#Develop Model on training data
fit_LR = lm(Absenteeism_time_in_hours ~ ., data = train_data)
#Lets predict for training data
pred_LR_train = predict(fit_LR, train_data[,names(test_data) != "Absenteeism_time_in_hours"])
#Lets predict for testing data
pred_LR_test = predict(fit_LR,test_data[,names(test_data) != "Absenteeism_time_in_hours"])
# For training data
print(regr.eval(train_data[,"Absenteeism_time_in_hours"],pred_LR_train,stats=c("rmse","mse","mae")))
cat("rsq for train", rsq(train_data[,"Absenteeism_time_in_hours"],pred_LR_train))
# For testing data
print(regr.eval(test_data[,"Absenteeism_time_in_hours"],pred_LR_test,stats=c("rmse","mse","mae")))
cat("rsq for test", rsq(test_data[,"Absenteeism_time_in_hours"],pred_LR_test))
#-----------------------2) Decision Trees
set.seed(1234)
```

```
#Develop Model on training data
fit_DT = rpart(Absenteeism_time_in_hours ~., data = train_data, method = "anova")
#Summary of DT model
summary(fit_DT)
#write rules into disk
write(capture.output(summary(fit_DT)), "Rules.txt")
#Lets predict for training data
pred_DT_train = predict(fit_DT, train_data[,names(test_data) != "Absenteeism_time_in_hours"])
#Lets predict for training data
pred_DT_test = predict(fit_DT,test_data[,names(test_data) != "Absenteeism_time_in_hours"])
# For training data
print(regr.eval(train_data[,"Absenteeism_time_in_hours"],pred_DT_train,stats=c("rmse","mse","mae")))
cat("rsq for train", rsq(train_data[,"Absenteeism_time_in_hours"],pred_DT_train))
# For testing data
print(regr.eval(test_data[,"Absenteeism_time_in_hours"],pred_DT_test,stats=c("rmse","mse","mae")))
cat("rsq for test", rsq(test_data[,"Absenteeism_time_in_hours"],pred_DT_test))

#------------------------3) Random Forest
set.seed(1234)
#Develop Model on training data
fit_RF = randomForest(Absenteeism_time_in_hours~., data = train_data)
#Lets predict for training data
pred_RF_train = predict(fit_RF, train_data[,names(test_data) != "Absenteeism_time_in_hours"])
#Lets predict for testing data
pred_RF_test = predict(fit_RF,test_data[,names(test_data) != "Absenteeism_time_in_hours"])
# For training data
print(regr.eval(train_data[,"Absenteeism_time_in_hours"],pred_RF_train,stats=c("rmse","mse","mae")))
cat("rsq for train", rsq(train_data[,"Absenteeism_time_in_hours"],pred_RF_train))
# For testing data
print(regr.eval(test_data[,"Absenteeism_time_in_hours"],pred_RF_test,stats=c("rmse","mse","mae")))
cat("rsq for test", rsq(test_data[,"Absenteeism_time_in_hours"],pred_RF_test))
```

Chapter 5 **References**

1. http://thestatsgeek.com/2013/10/28/r-squared-and-adjusted-r-squared/?source=post_page
2. Martiniano, A., Ferreira, R. P., Sassi, R. J., & Affonso, C. (2012). Application of a neuro fuzzy network in prediction of absenteeism at work. In Information Systems and Technologies (CISTI), 7th Iberian Conference on (pp. 1-4). IEEE.
3. Graham, J. W. (2009). Missing data analysis: Making it work in the real world. *Annual Review of Psychology, 60*, 549-576.
4. Grigorios Papageorgiou, Stuart W Grant, Johanna J M Takkenberg, Mostafa M Mokhles, Statistical primer: how to deal with missing data in scientific research?, *Interactive CardioVascular and Thoracic Surgery*, Volume 27, Issue 2, August 2018, Pages 153–158,
5. http://www.cs.columbia.edu/~amueller/comsw4995s19/schedule/
6. [1] Buuren, S. V., & Groothuis-Oudshoorn, K. (2011). Mice: Multivariate Imputation by Chained Equations in R. Journal of Statistical Software