# **Employee Absenteeism**

# 

**Bharti Sharma**

**Table of Contents**

[1. Introduction 4](#_Toc529012078)

[1.1 Problem Statement 4](#_Toc529012079)

[1.2 Data 4](#_Toc529012080)

[2. Methodology 5](#_Toc529012081)

[2.1 Exploratory Data Analysis 5](#_Toc529012082)

[2.3 Data Preprocessing 19](#_Toc529012083)

[2.3.1 Missing Values 19](#_Toc529012084)

[2.3.2 Outlier Analysis: 25](#_Toc529012085)

[2.3.4 Feature Scaling 33](#_Toc529012086)

[2.4 Sampling 34](#_Toc529012087)

[2.5 Model Development 35](#_Toc529012088)

[2.5.1 Model Evaluation 35](#_Toc529012089)

[2.5.2 Error Measure 36](#_Toc529012090)

[2.5.3 Confusion Matrix 36](#_Toc529012091)

[2.5.4 Cross Validation 36](#_Toc529012092)

[2.6 Model Selection 37](#_Toc529012093)

[2.6.1 Linear Regression Model 37](#_Toc529012094)

[2.6.2 Decision Tree 41](#_Toc529012095)

[2.6.3 Random Forest 43](#_Toc529012096)

[2.6.4 KNN 44](#_Toc529012097)

[2.6.5 Model Selection 45](#_Toc529012098)

[2.7 Generate Output File 46](#_Toc529012099)

[3.Conclusion 46](#_Toc529012100)

[3.1 “What changes company should bring to reduce the number of absenteeism?” 47](#_Toc529012101)

[3.2 How much losses every month can we project in 2011 if same trend of absenteeism continues? 48](#_Toc529012102)

[3.3 Challenges Faced 50](#_Toc529012103)

[Appendix – A 51](#_Toc529012104)

[Outlier Analysis Using Boxplot Method 51](#_Toc529012105)

[boxplots to visualize Outliers after replacing outliers 55](#_Toc529012106)

[Density plot and Histogram of variables 59](#_Toc529012107)

[Histogram showing Mean and skewed data of variables 62](#_Toc529012108)

[Appendix – B 68](#_Toc529012109)

[Python Code 68](#_Toc529012110)

[Problem : Employee Absenteeism 68](#_Toc529012111)

[Data 69](#_Toc529012112)

[Exploratory Data Analysis 69](#_Toc529012113)

[Data Preprocessing 72](#_Toc529012114)

[Missing Values Imputation 72](#_Toc529012115)

[Outlier Analysis 74](#_Toc529012116)

[Feature selection 75](#_Toc529012117)

[Feature Scaling 79](#_Toc529012118)

[Model development 79](#_Toc529012119)

[Appendix – C 85](#_Toc529012120)

[R-Code 85](#_Toc529012121)

# **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 its dataset and requested to have 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?

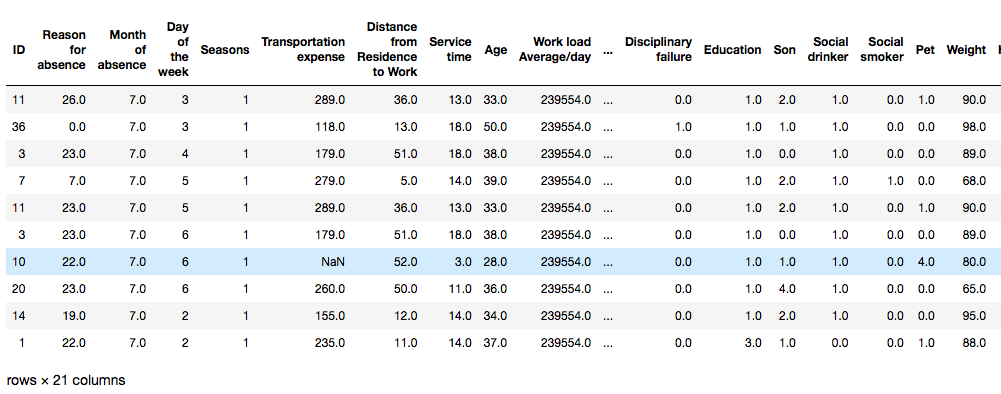
The aim of the project is to predict losses every month. We are applying some statistical techniques and machine learning algorithms on a public dataset that has features for analysing the predictions. We are developing an algorithm to predict losses every month based on given dataset *.*

## **1.2 Data**

Our task is to build regression model which will predict the absenteeism time in hours based on predictors as:

* ID
* Reason for absence
* Month of absence
* Day of the week
* Seasons
* Transportation expense
* Distance from Residence to Work
* Service time
* Age
* Work load Average/day
* Hit target
* Disciplinary failure
* Education
* Son
* Social drinker
* Social smoker
* Pet
* Weight
* Height
* Body mass index

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



# **2. Methodology**

## **2.1 Exploratory Data Analysis**

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.

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

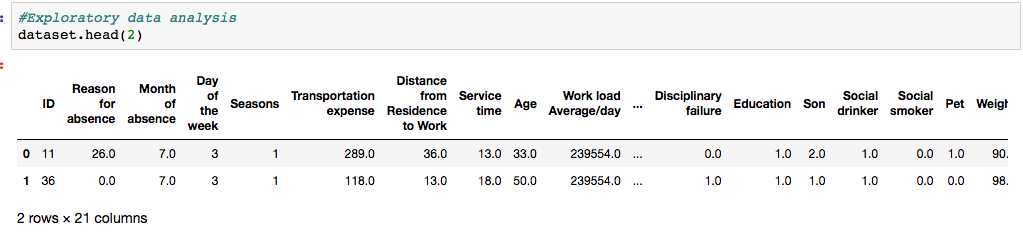
We have performed some steps for exploratory data analysis in python and R code to understand data in a better way and also to get a glimpse of dataset.

I have shared some screenshots from python code. Entire code has been shared in Appendix A and B.

* Loaded the dataset in notebook.



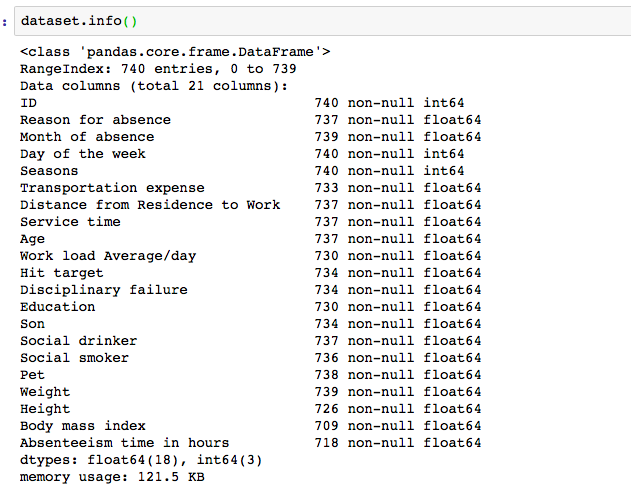
* To take a closer look at the data took help of “ .head()”function of pandas library which returns first five observations of the data set.Similarly “.tail()” returns last five observations of the data set.



* We found out the total number of rows and columns in the data set using “.shape”.

../Screen%20Shot%202018-10-15%20at%2011.49.02%20AM.png

* Dataset comprises of 740 observations and 21 characteristics. Out of which one is dependent variable and rest 20 are independent variablesWe will know the columns and their corresponding data types, along with finding whether they contain null values or not using “info()” function .



* Data has only float and integer values. Since some of the variables should be categorical for example : Id , Reason for absence, Month of absence, Day of the week , Seasons etc, So as per analysis we need to convert them from continuous to categorical variables. Since they are not carrying any continuous type of information whereas they are carrying a categorical type of information.
  + some variable column has null/missing values.
* We will use “describe()” function in pandas which helps in getting various summary statistics. This function returns the count, mean, standard deviation, minimum and maximum values and the quantiles of the data.

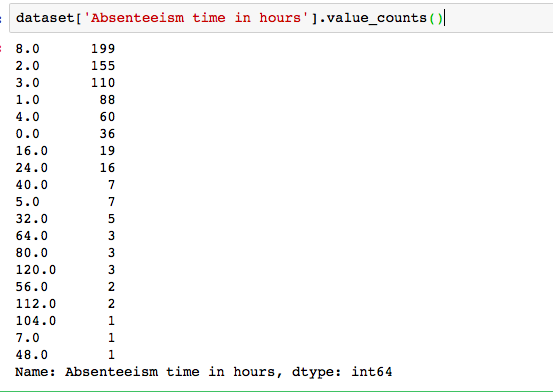


* Here as you can notice mean value is less than median value of some of column which is represented by 50%(50th percentile) in index column.

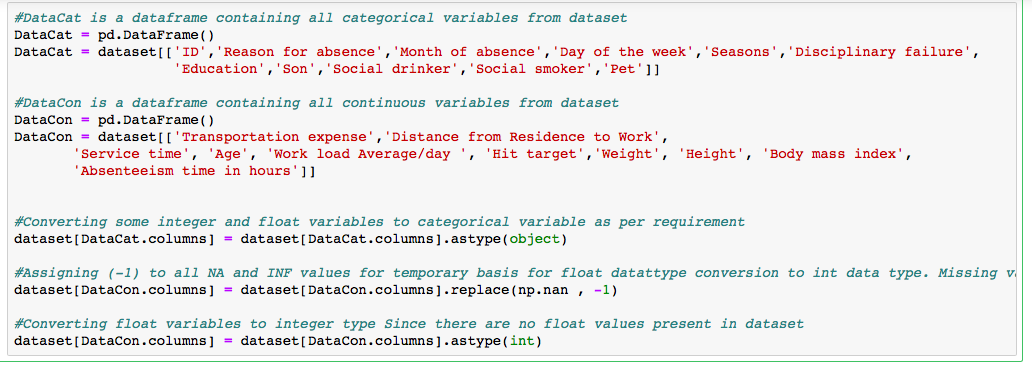
There is notably a some difference between 75th %tile and max values of predictors.

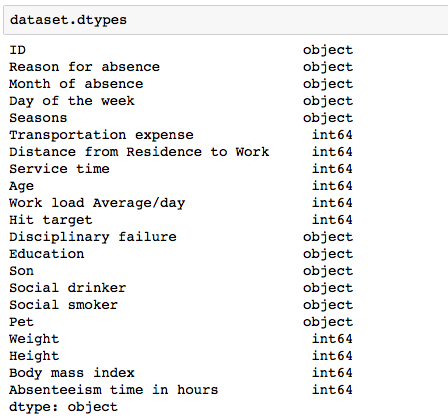
Thus observations 1 and 2 suggests that there are some Outliers present in our data set.

* Checking frequency of the employee who have maximum hours of ‘Absenteeism time in hours'
* This tells us vote count of ‘Absenteeism time in hours' score in descending order .



* This tells us most of the employee have ‘Absenteeism time in hours' are 8 , 2,3,1 and 4 .
* Data has only float and integer values. Since some of the variables should be categorical for example : Id , Reason for absence, Month of absence, Day of the week , Seasons etc,
* So as per analysis we need to convert them from continuous to categorical variables. Since they are not carrying any continuous type of information whereas they are carrying a categorical type of information.
* Converting those variables as per project requirement to categories to understand data better.
* For continuous variable we will convert float type variables which are containing only integer type of data in the dataset.
* For this we have also created dataframe “DataCat” which contains all categorical variables and “DataCon” which contains all continuous variables.
* Assigning (-1) to all NA and INF values for temporary basis for float datatype conversion to integer data type. We will handle missing value as per process under the data pre processing.





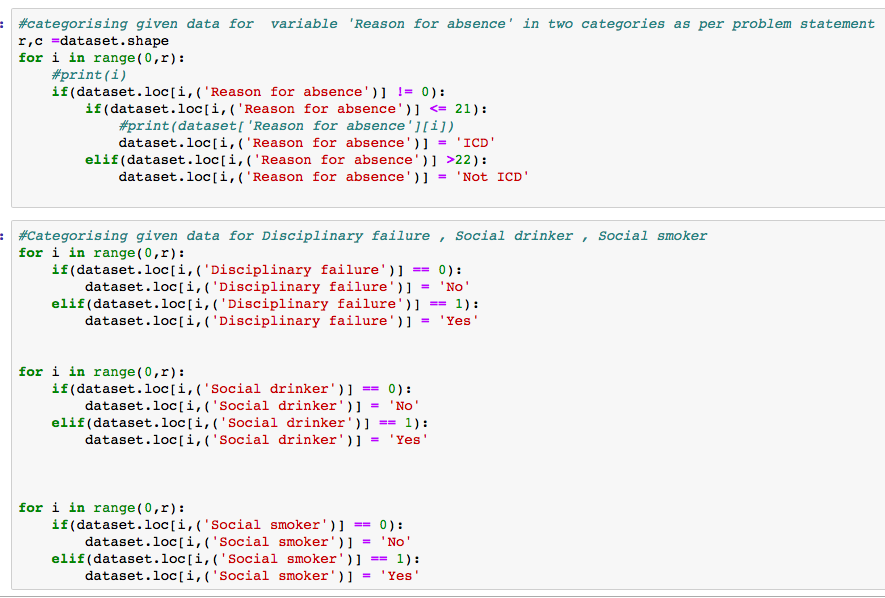
* After converting required variables to categorical type. We are going to mention their categories as per given in our problem statement so that we can understand our data better.
* As per our problem statement independent variable “Reason for absence”

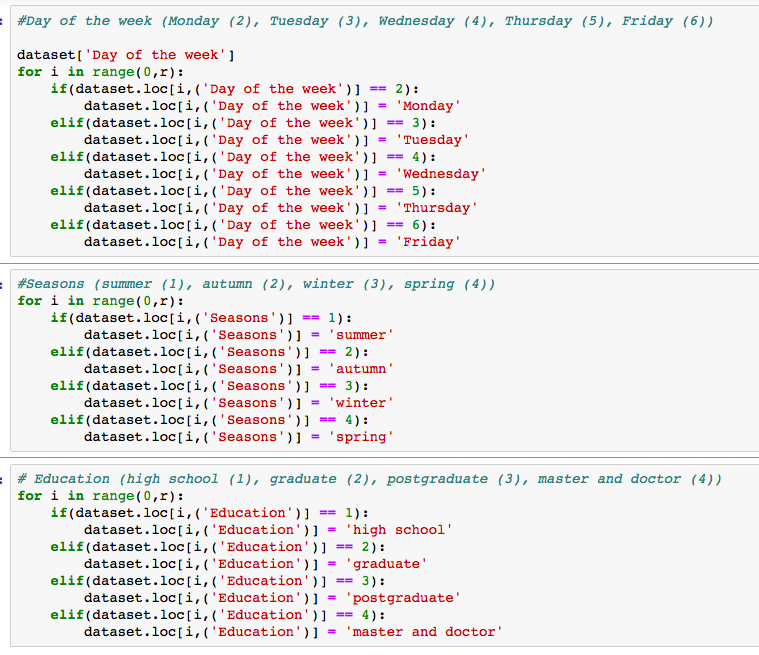
Have 28 categories. Absences attested by the International Code of Diseases (ICD) stratified into 21 categories (I to XXI) as follows: I Certain infectious and parasitic diseases II Neoplasms III Diseases of the blood and blood-forming organs and certain disorders involving the immune mechanism IV Endocrine, nutritional and metabolic diseases V Mental and behavioural disorders VI Diseases of the nervous system VII Diseases of the eye and adnexa VIII Diseases of the ear and mastoid process IX Diseases of the circulatory system X Diseases of the respiratory system XI Diseases of the digestive system XII Diseases of the skin and subcutaneous tissue XIII Diseases of the musculoskeletal system and connective tissue XIV Diseases of the genitourinary system XV Pregnancy, childbirth and the puerperium XVI Certain conditions originating in the perinatal period XVII Congenital malformations, deformations and chromosomal abnormalities XVIII Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified XIX Injury, poisoning and certain other consequences of external causes XX External causes of morbidity and mortality XXI Factors influencing health status and contact with health services. And 7 categories without (CID) patient follow-up (22), medical consultation (23), blood donation (24), laboratory examination (25), unjustified absence (26), physiotherapy (27), dental consultation (28).

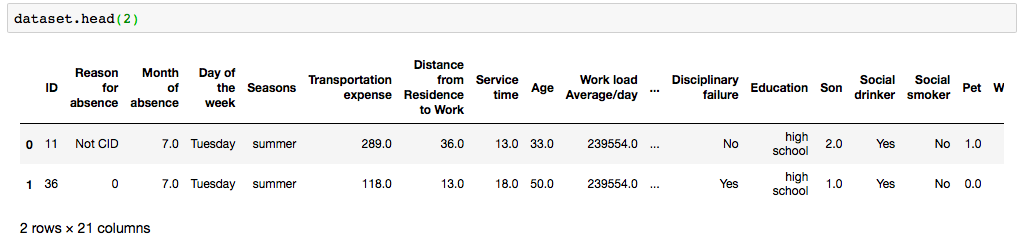
Which are majorly classified into two categories

1. ICD
2. Not ICD

* Categories given data as per problem statement
* For Disciplinary failure , Social drinker , Social smoke to Yes/No
* Day of the week to (Monday (2), Tuesday (3), Wednesday (4), Thursday (5), Friday (6))
* Seasons to (summer (1), autumn (2), winter (3), spring (4))
* Education to (high school (1), graduate (2), postgraduate (3), master and doctor (4))







We have got the data after doing manipulations on categorical variables. Python and R Code has been shared in Appendix – B,C

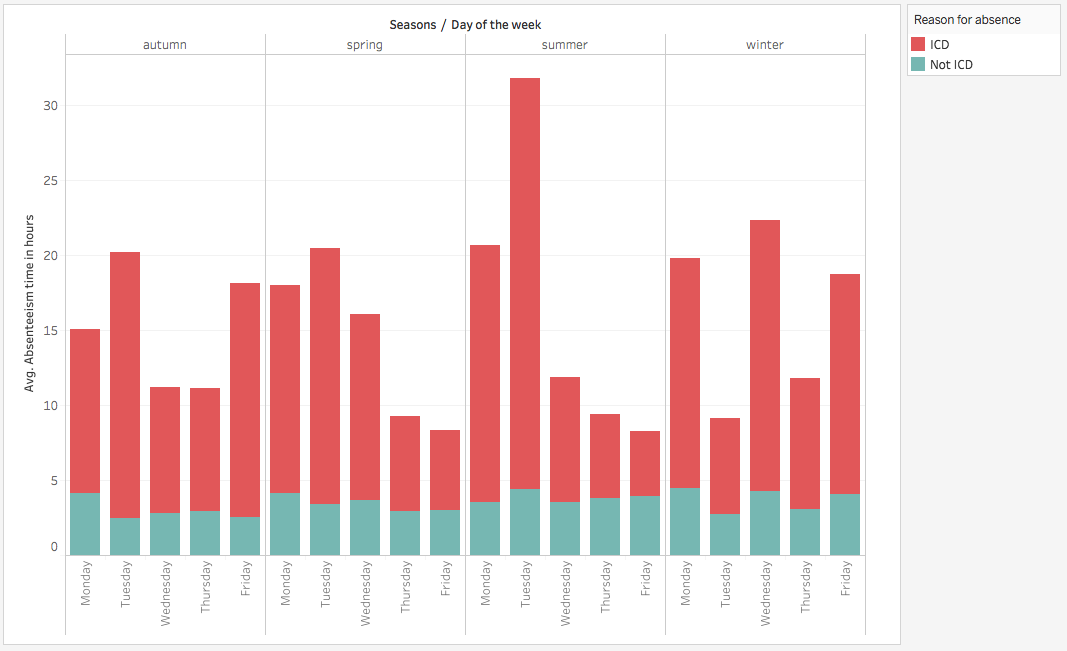
**2.2 Visualizations**

Data visualization refers to the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization is an accessible way to see and understand trends, outliers, and patterns in data.

As per our problem statement, we are going to analyse relationship of target variable with categorical and continuous variables of given dataset to get an initial insights of data.So that before going for data pre-processing we have an idea about initial impact of different factors from all variables. For detailed observations we will be doing data pre-processing and model development.  
We have divided similar type of variables ( variables which are sharing related information ex. Season , Weekdays , months . They are representing form of time.) to compare with target variable. For these variables we will plot graphs using Tableau.

1. First graph is showing stacked bar graph of 'Reason for absence','Day of the week','Seasons' and 'Absenteeism time in hours'.

Sum of absenteeism time in hours for each day of the week broken down by ‘Seasons’. Colors shows details about ‘Reason of absence’. The view is filtered on ‘Reasons for absence’, which keeps ICD and NOT ICD.



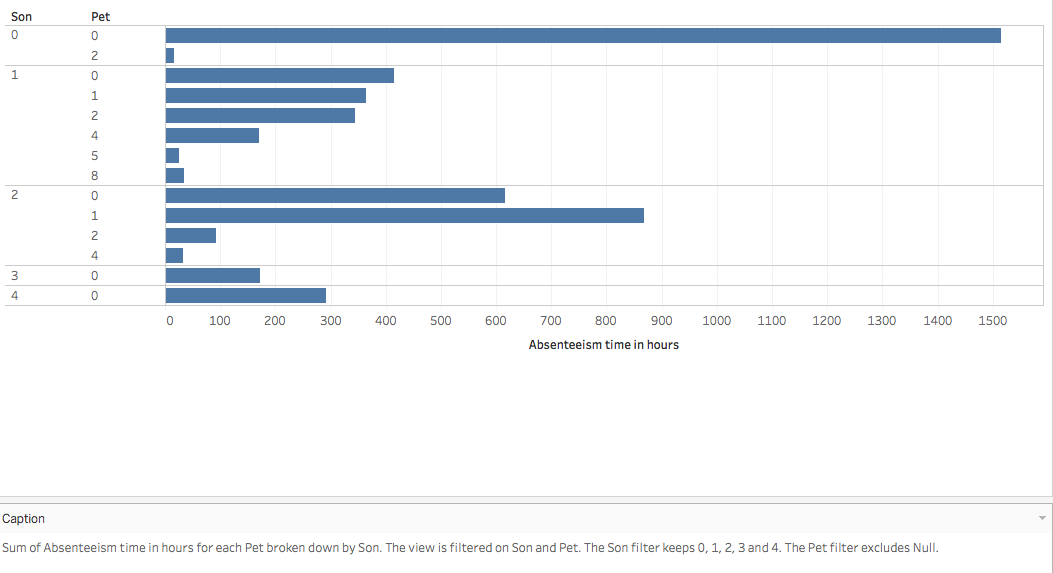
We have some observation

* Overall on starting two days(Mon,Tue) of the week Avg. 'Absenteeism time in hours' is higher than mid week days(wed,Thur).
* There is one exception that in summer season on Tuesdays 'Absenteeism time in hours' is much higher than avg. 'Absenteeism time in hours' in other Seasons. So it should

be discussed with employee to know if there is some particular reason behind it.

* ‘Reason for absence’ is most of the time ‘ICD’ type comparatively ‘NOT ICD’.
* So as per this graph , absenteeism is higher because in most of the cases ‘Reason for absence ‘ is ‘ICD’ which means employees have serious health related issues. Which cannot be ignored or postponded .
* To improve this , company can provide monthly health checkups , start giving healthy breakfast or also have some physical fitness activities to keep the employees healthy and with this the problem of absenteeism can be avoided or reduced.

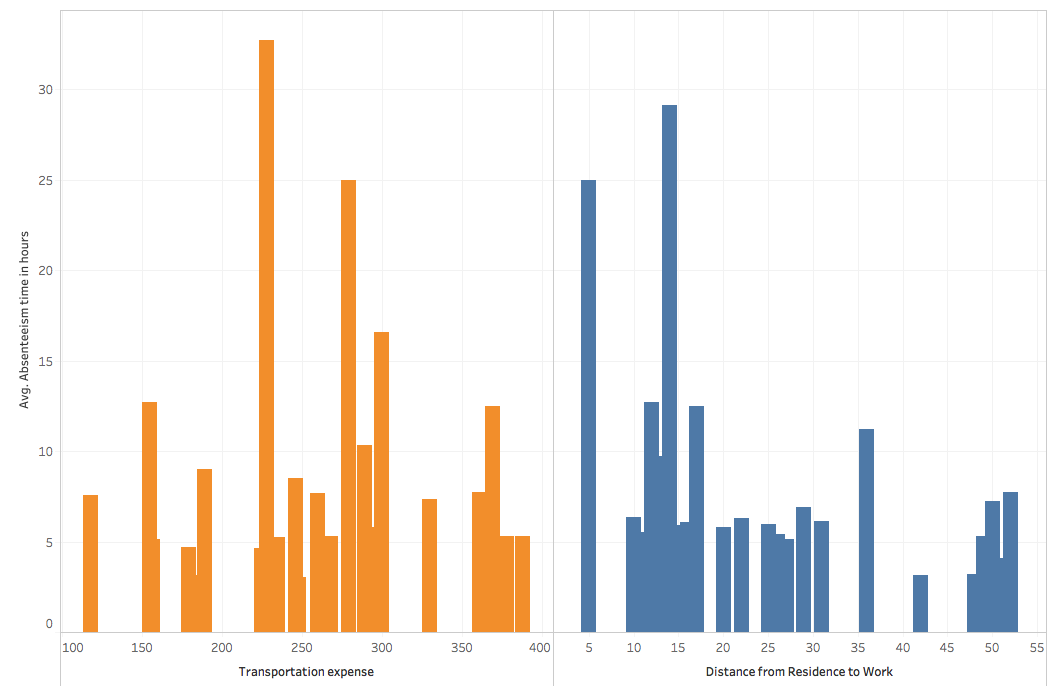
1. Second graph is showing us horizaontal bar graph for ‘Son’ ,’Pet’ and 'Absenteeism time in hours'.Sum of absenteeism time in hours for each pet broken down by Son. The view is filtered on Son and Pet.



From the above graph we can say if employee do not have any pet or children to take care. They are more likely absent comparatively who have children or pet.

1. Third graph is showing us bar graph for ‘Transportation Expense ,’Distance from Residence to Work’ and 'Absenteeism time in hours'.Sum of absenteeism time in hours for each pet broken down by Son. The view is filtered on Son and Pet.

The plots of average of Absenteeism time in hours for Transportation expense and Distance from Residence to Work. The data on Absenteeism time in hours, which ranges from 0 to 30. The view on Transportation expense and Distance from Residence to Work. The Transportation expense ranges from 0 to 388. The Distance from Residence to Work ranges from 0 to 52.



We can say from this graph that Absenteeism time in hours for Transportation expense from range 225 – 300 is higher than other ranges of Transportation Expenses and Distance of 5Km and 10Km from Residence to Work has particularly have higher Absenteeism time in hours.

1. The fourth graph shows plots of average of Absenteeism time in hours for Age, Body mass index, Height and Weight. The data is filtered on Transportation expense, Distance from Residence to Work and Absenteeism time in hours. The Absenteeism time in hours filter ranges from 0 to 120. The view is filtered on Age, Body mass index, Height and Weight. The Age filter ranges from 0 to 58. The Body mass index filter ranges from 0 to 38. The Height filter ranges from 0 to 196. The Weight filter ranges from 0 to 108.



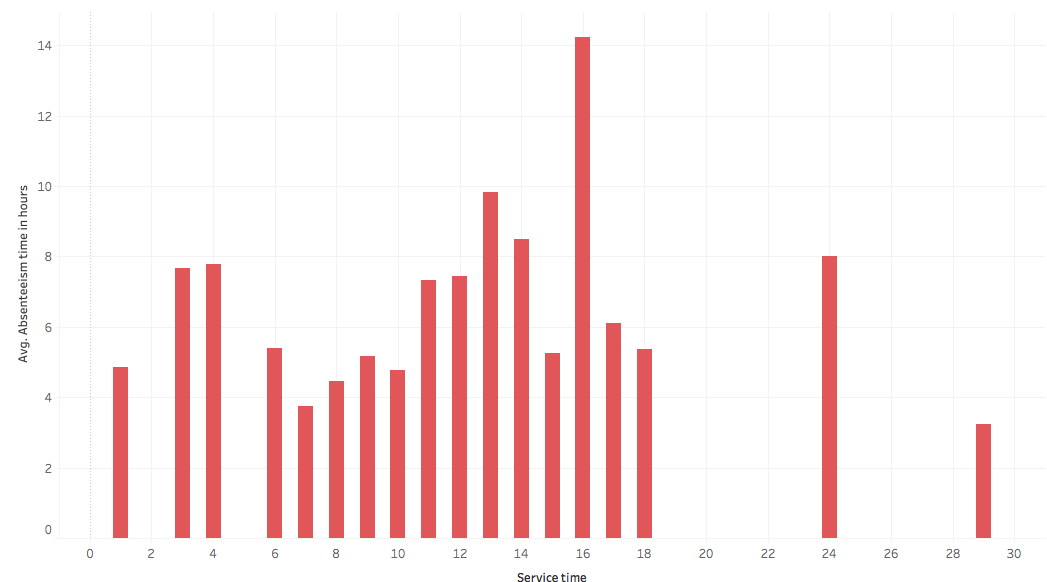
Observation from the graph is If age of employee is near 60 or 58 then he tends to absent more frequently for long hours. Whereas BMI , Height, Weight doesn’t show much variability in graph.

This is visible that employee of old age may have some health/travel concerns.

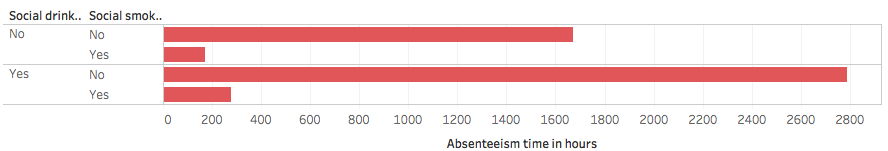
So personally for old age group people some travel/health benefits can be offered to reduce absenteeism.

1. The fifth graphs shows plots of average of Absenteeism time in hours for

Service Time. In this case there are some outliers present. In a day there are 24 hours But the data here shows service time as 29 hours and 24 hours, Which is not possible. So they are clearly outliers and will be taken care under outlier analysis.



1. The sixth graph shows sum of absenteeism time in hours for each social smoker broken down by social drinker.



From graph we can observe that if employee is social smoker than his absenteeism time is lesser than others.

Whereas if an employee is only social drinker than his absenteeism is higher than others.

**Box Plot**

A box plot (or box-and-whisker plot) shows the distribution of quantitative data in a way that facilitates comparisons between variables. The box shows the quartiles of the dataset while the whiskers extend to show the rest of the distribution.

* The box plot (a.k.a. box and whisker diagram) is a standardized way of displaying the distribution of data based on the five number summary:

Minimum

First quartile

Median

Third quartile

Maximum.

In the simplest box plot the central rectangle spans the first quartile to the third quartile (the interquartile range or IQR).



A segment inside the rectangle shows the median and “whiskers” above and below the box show the locations of the minimum and maximum.

The above figure shows boxplot for Service time. As we have already discussed in service time and absenteeism in hours graph , there is one outlier present for this. We can also observe the particular outlier here.

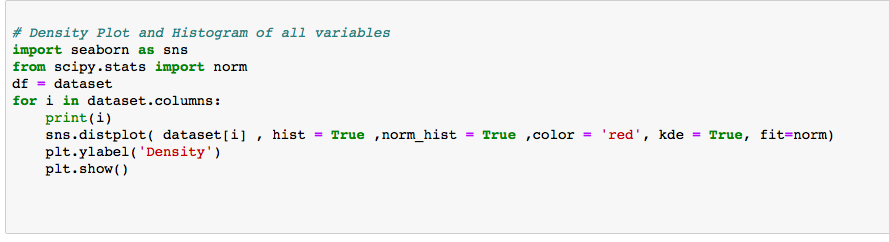
Similarly we have plotted boxplot for all variables. Code and figures for them has been shared in Appendix- A.

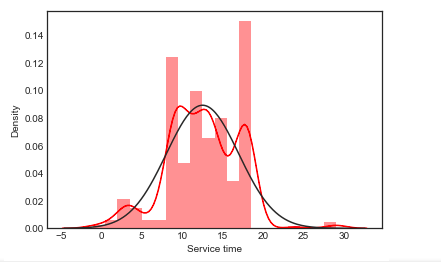
**Distribution Plot**

We will look at all the probability distributions of the variables. Most analysis like regression, require the data to be normally distributed. We can visualize that in a glance by looking at the probability distributions or probability density functions of the variable.

In Figure 2.1 we have plotted the probability density Kernel density estimate (kde) is a quite useful tool for plotting the shape of a distribution. functions of all the continuous variable properties we have available in the data . The red lines indicate Kernel Density Estimations (KDE)1 of the variable. The black lines represent the normal distribution. So as you can see in the ﬁgure most variables either very closely, or somewhat imitate the normal distribution. Those plots can be viewed in the Appendix – A.

all independent variables are right skewed/positively skewed.





**Histogram for Skewed data with Mean**



**2.3 Data Preprocessing**

Data preprocessing is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects. Data-gathering methods are often loosely controlled, resulting in out-of-range values (e.g., Income: −100), impossible data combinations (e.g., Sex: Male, Pregnant: Yes), missing values, etc. Analyzing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first and foremost before running an analysis. Often, data preprocessing is the most important phase of a machine learning project.

**2.3.1 Missing Values**

Missing Value treatment becomes important since the data insights or the performance of your predictive model could be impacted if the missing values are not appropriately handled

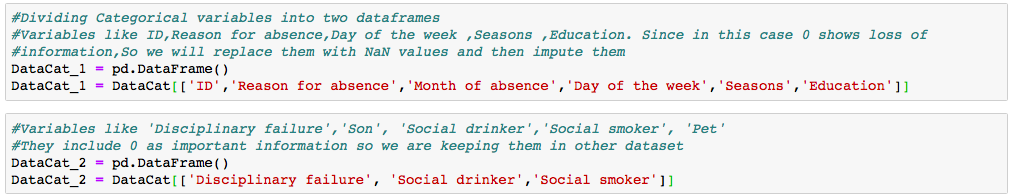
For our data first of all we will check for Missing values and then we will impute those values by checking suitable method.

**Invalid data**

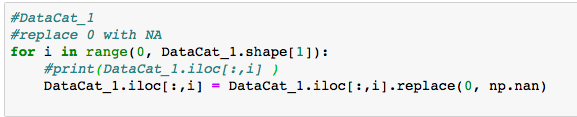
Along with missing values in dataset, we also have some invalid entries which does not carry meaningful information ex. In categorical variable ‘Day of Week’ we have 5 valid entries which are 2,3,4,5 and 6 whereas in dataset 0 is also present which doesn’t carry any information So this comes under invalid data. So we can replace them with NaN values and then impute with missing data.

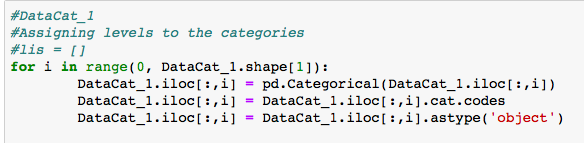
Similarly we have some categorical variables in which we have found invalid data. So to correct this behaviour in entire dataset, we have divided the data accordingly . And we have divided categorical dataset in two dataframes

‘DataCat\_1’ and ‘DataCat\_2’ .

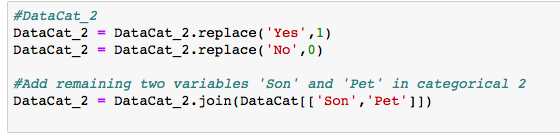


Variables like ID,Reason for absence,Day of the week ,Seasons ,Education. Since in this case 0 shows loss of information,So we will replace them with NaN values and then impute them.



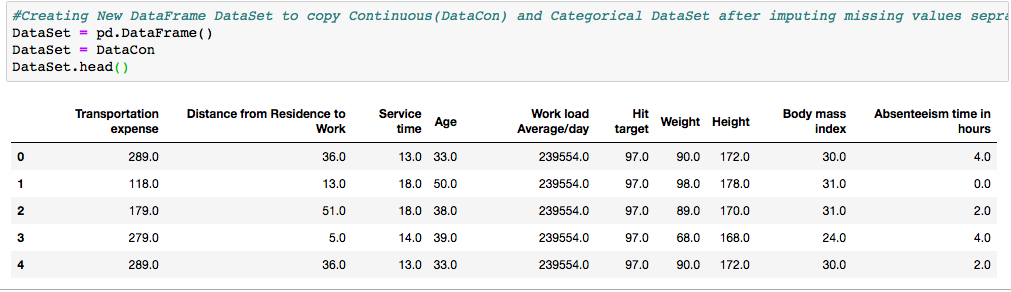


Whereas Variables like 'Disciplinary failure','Son', 'Social drinker','Social smoker', 'Pet' . They include 0 as important information so we are keeping them in other dataset.



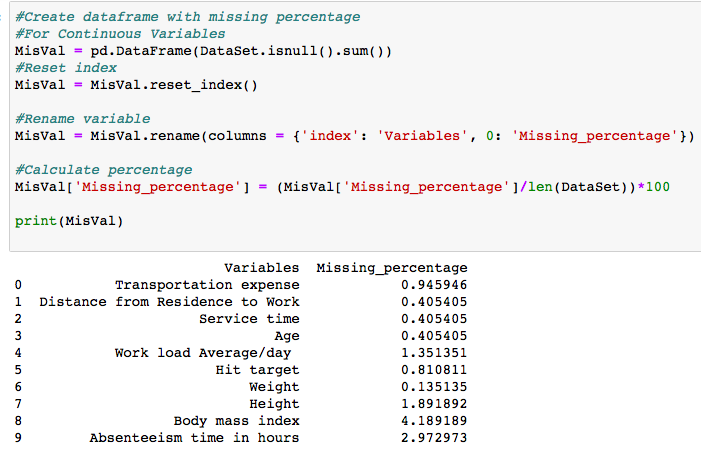
Creating New DataFrame ‘DataSet’ to copy Continuous(DataCon) and Categorical DataSet after imputing missing values separately . Since for categorical data we are using Mode Method because with any other method like KNN we sometimes get float numbers to categories . Which lead to incorrect data.So for this reason we are using mode method here.

whereas for continuous data we will check for Mean, Median and KNN Method. And best method will be used for this.



**-> Imputation for Continuous Variables**

Now we will check what are the percentile out of total observations which are missing. If we have more than 60% missing values in a column. Then we can drop the column. So let us find out what are the percentage of missing values in all the independent variables.



As per above screenshot we can say that we do not have any variable with more than 60% of missing values.So there is no need to drop any variable.

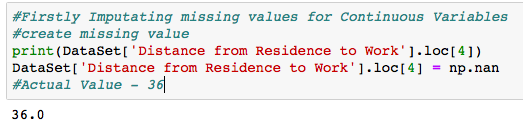
Now for missing value imputation we have three methods:

* Mean imputation
* Median imputation
* KNN imputation

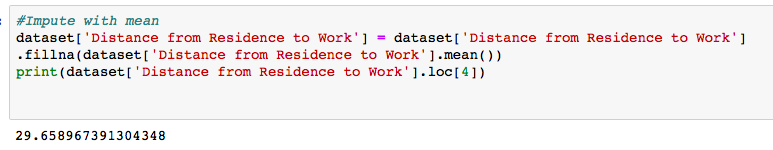
We will check for mean, median and KNN results for by imputing one variable. Whose actual value we already know, we will make that value null and then we can try imputing by all three methods one by one then we can select the method by which we get close results to actual value ,Now we will select the value which is very common or we can say mid range value from a variable . So that we can me most accurate in our evaluation.

There are few steps for finding out suitable method:

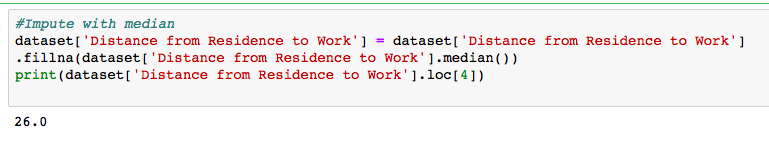
Step1. Making one observation ‘NA’



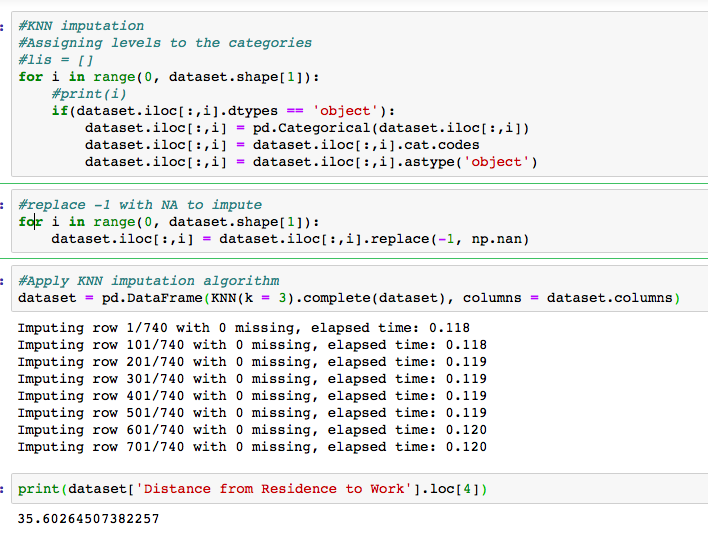
Step2. Imputing with Mean



Step3. Imputing with Median

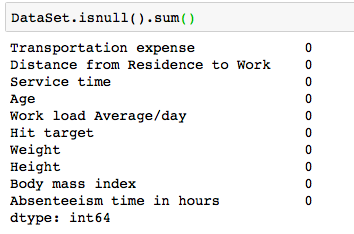


Step4. Imputing with KNN



So as per above results we can say that we can use KNN imputation is best for our dataset ,as it is giving most accurate result in all of the methods.

Step 5. Check If all the missing values have been imputed

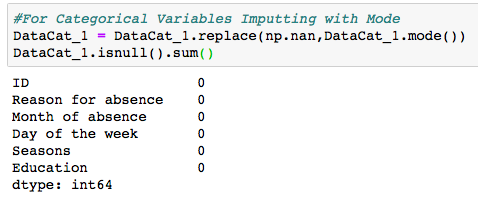


So by above screenshot we can say all missing values in continuous data have been imputed.

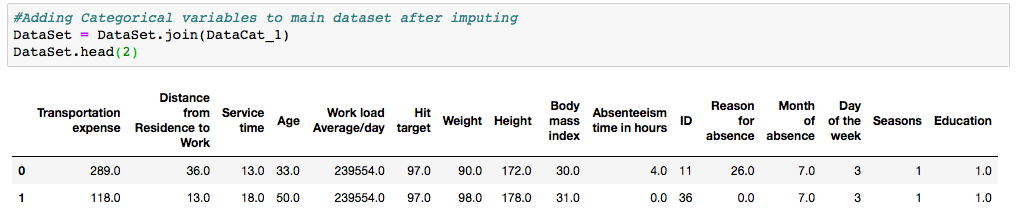
**-> Imputation for Categorical Variables**

For categorical data we are using Mode Method because with any other method like KNN we sometimes we get float numbers to categories . Which lead to incorrect data. So for this reason we are using mode method here.

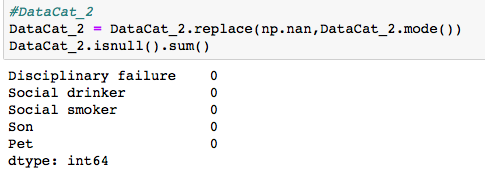
* Firstly we are imputing DataCat\_1,which is first division of categorical data using Mode method



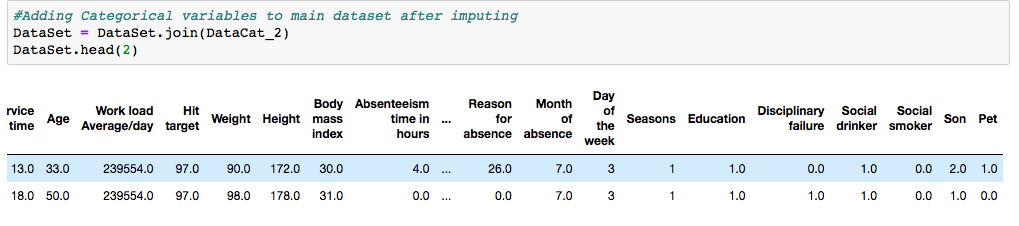
Now after missing value imputation ,Adding (DataCat\_1 )categorical dataset 1 to main DataSet.



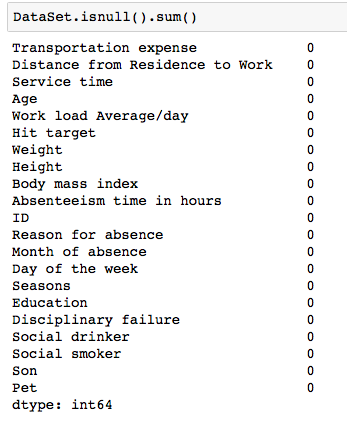
* we are imputing DataCat\_2,which is second division of categorical data using Mode method.



Now after missing value imputation ,Adding (DataCat\_2)categorical dataset 2 to main DataSet.



Final DataSet after imputing categorical and continuous variables. We will check for missing values .

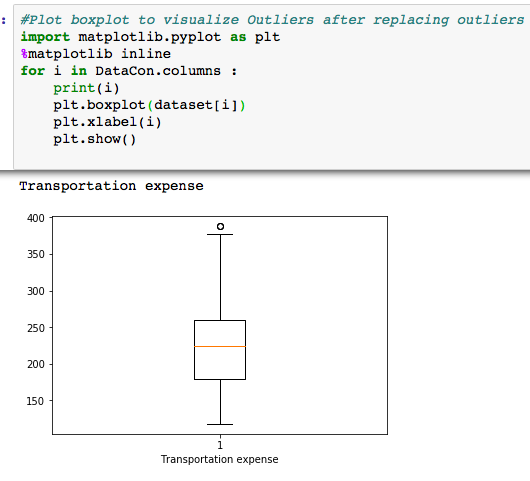


As we have not getting any missing value in dataset. So we can say missing value imputation has been done successfully.

**2.3.2 Outlier Analysis:**

Now we have to check if there are outliers present in dataset for continuous variables. As we have already divided dataset in two categories one is a data frame ’DataCat’ for categorical variables and another is ‘DataCon’ for continuous variables. Here we will check outliers present in continuous dataset.

I have used for loop to plot boxplot for all continuous variables. Here I have shared one plot for transportation expense. All the boxplots have been shared in appendix.



Here Now as we have seen that there are outliers present in dataset we will correct the outliers .Here we are replacing values which are greater than maximum by maximum value and minimum by minimum value as per given in below code.



Now we can check again if we have replaced outliers correctly by plotting boxplots again.



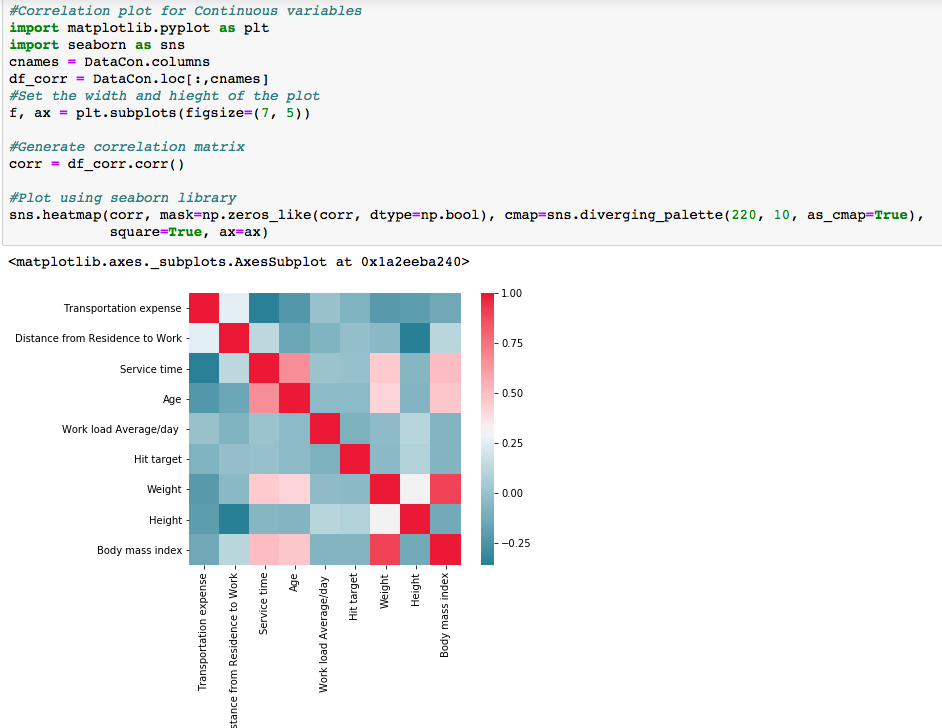
As per this there are no outliers present now in dataset. All the plots have been shared in Appendix.

**2.3.3 Feature Selection**

Before performing any type of modeling we need to assess the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of class prediction. There are several methods of doing that. Below we have used Random Forests to perform features selection. #For Continuous we will use correlation analysis#For Categorical we will use Chi square testAs we know there should be no dependency between independent variables. Also we have separated continuous and Categorical Variables in two groups as Continuous Data (DataCon) , Categorical Data (DataCat). So that we can apply individual suitable techniques for both the groups For Continuous we will use correlation analysis and For Categorical we will use Chi square test.

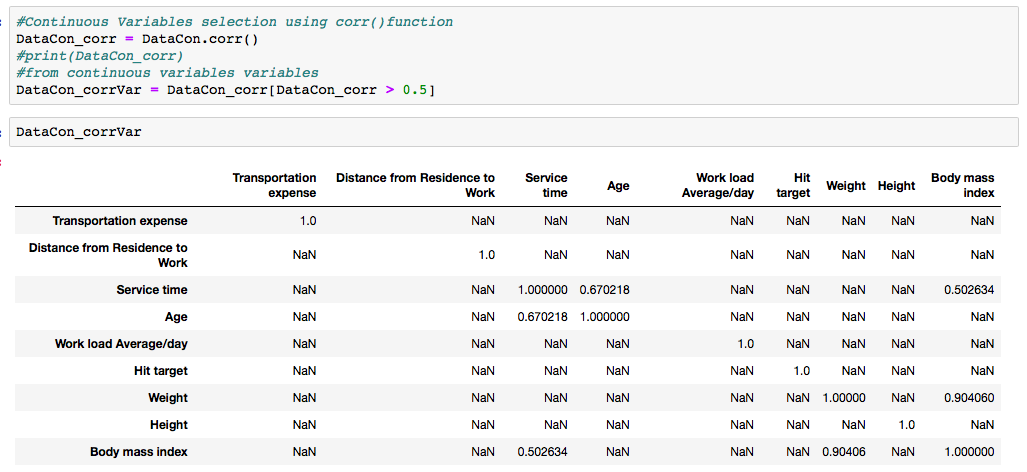
**->Feature Selection From Continuous Variables**

* Python has a visualization library ,[Seaborn](https://seaborn.pydata.org/" \t "_blank) which build on top of matplotlib. It provides very attractive statistical graphs in order to perform both [Univariate](http://www.statisticshowto.com/univariate/) and [Multivariate analysis](http://www.camo.com/multivariate_analysis.html).
* To use linear regression for modelling,its necessary to remove correlated variables to improve your model.One can find correlations using pandas “.corr()” function and can visualize the correlation matrix using a heatmap in seaborn.
* We have heatmap plot to have a vision about overall dataset, We will explore correlation in feature selection after dividing categorical and continuous variables in detail.

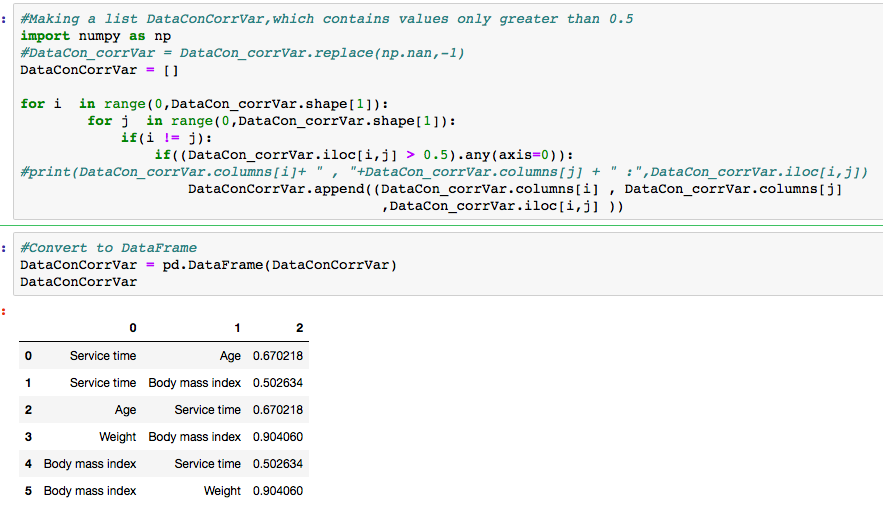


For continuous data we are using corr() function here to check the correlation among the continuous variables.

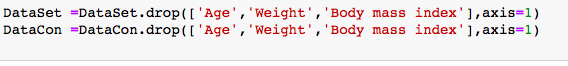
DataCon is representing continuous data here. If correlation is more than 0.5 than we have kept those variables in DataCon\_corrVar .



As we have got some nan values So we are removing them to find a clear dataframe of variables which are highly correlated. Which we can see in dataframe ‘DataConCorrVar’.



Now as we have found that ‘Service time’ and ‘Age’ are highly correlated so we dropped ‘Age’ and ‘Body mass index’ is highly correlated with ‘weight’ So we dropped ‘weight’ . Since ‘Service time ‘ is also highly correlated with ‘Body mass index’. So similarly we are keeping ‘Service Time’ because it is common in all.



**-> Feature Selection From Categorical Variables**

For Categorical we will use Chi square test. As we know there should be no dependency between independent variables. The chi square test tests the null hypothesis that the categorical data has the given frequencies.

The Chi-Squared test is a statistical hypothesis test that assumes (the null hypothesis) that the observed frequencies for a categorical variable match the expected frequencies for the categorical variable. The test calculates a statistic that has a chi-squared distribution, named for the Greek capital letter Chi (X) pronounced “ki” as in kite.

As we have multiple independent categorical variables and our target variable is continuous. So we are not able to find out dependency between independent and dependent variables with the help of chi square test.

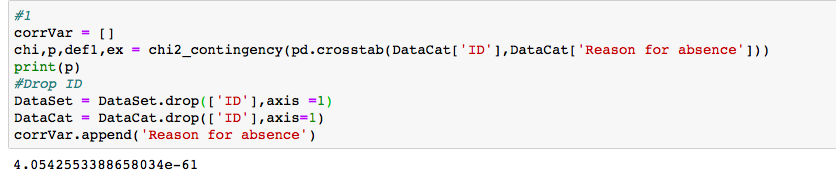
Here we are going to find dependency among independent categorical variables

With the help of chi square test.

* If p-value < 0.05 : significant result, reject null hypothesis (H0), dependent.
* If p-value > 0.05 : not significant result, fail to reject null hypothesis (H0), independent.

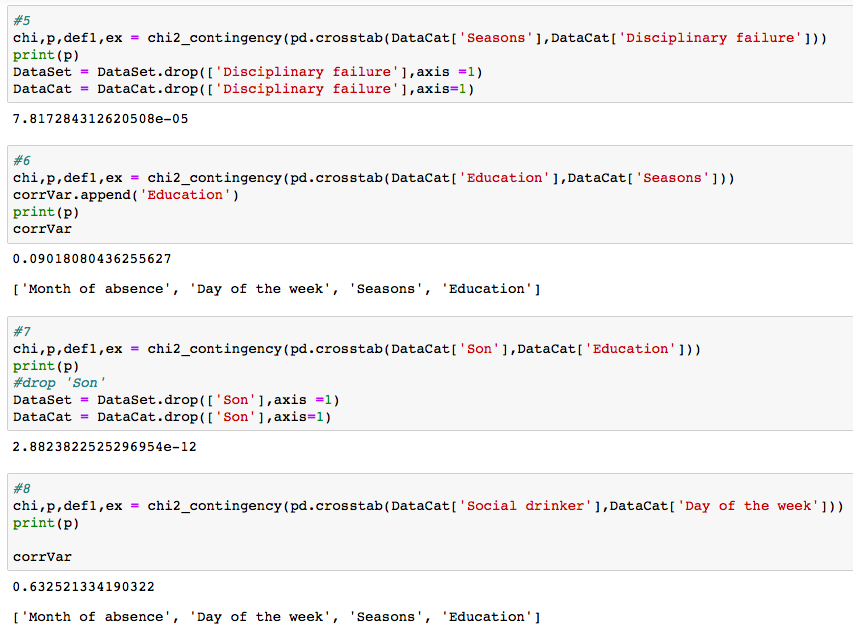
So if we have p-value < 0.05, we can drop one of the compared variables , to reduce collinearity in model.

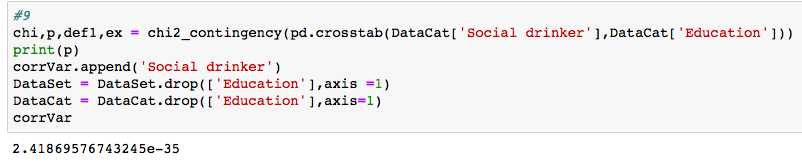
As in the below screenshot we have performed chi square test on two variables in which p-value is less than 0.05 ,So we can drop one variable of them.Since they are carrying same kind of data.Similarly we are going to check for all categorical variables. Untill we get all independent variables.

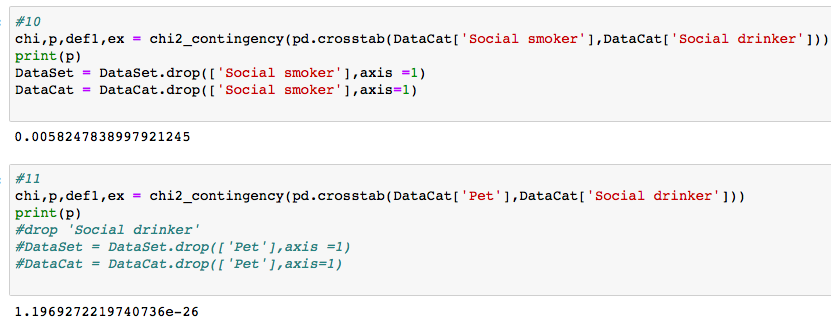


Similarly we are going to check for all categorical variables. Untill we get all independent variables.







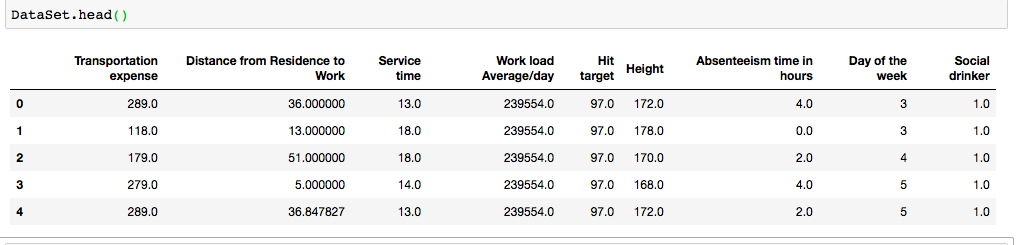


Here after applying chi square test on entire categorical data we are left with four variables 'Month of absence' , ‘Day of the week’ , ‘Seasons’, ’Social drinker’ .Now we will check correlation among them using same method



After applying chi square test on entire categorical dataset we have ‘Day of the week’ and ‘Social drinker’ Which are carrying important information.

We have got below DataSet after applying feature selection techniques for dimension reduction.



### 2.3.4 Feature Scaling

Feature scaling is the method to limit the range of the data. So that they can be compared on common ground.It can be performed only on continuous variables.

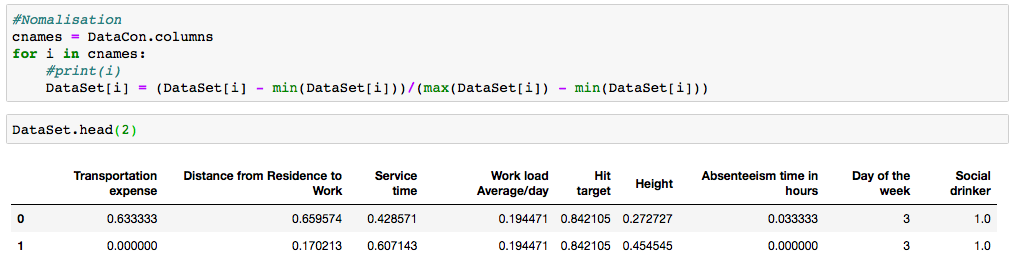
There are two methods of feature scaling:

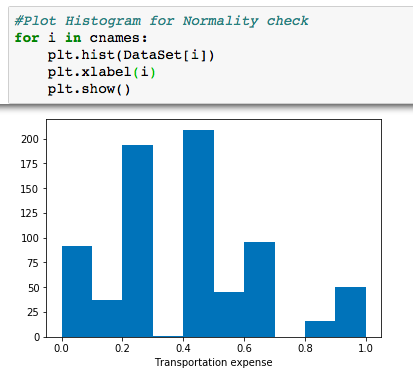
* Normalization
* Standardization / Z-score

Normalization is the process of scaling individual samples to have unit norm. This process can be useful if you plan to use a quadratic form such as the dot-product or any other kernel to quantify the similarity of any pair of samples.

Standardization works well if the data is uniformly distributed. As dataset for problem is not uniformly distributed as we have already seen in graphs that we have skewed data in our dataset. So as per requirement we have applied normalization on dataset to do feature scaling.

Applying Normalization on dataset for continuous variables.





All plots have been shared in Appendix- A.

### 

### 2.4 Sampling

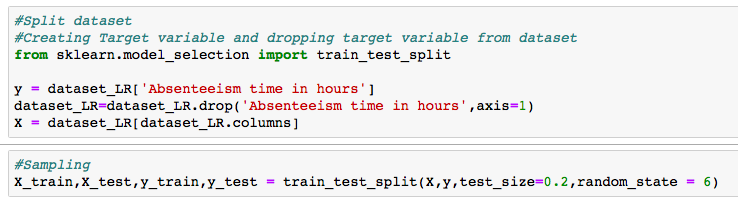
Sample is the subset of the population. The process of selecting a sample is known as sampling. Number of elements in the sample is the sample size.

There are different sampling techniques available like Simple Random Sampling, Stratified sampling, Systematic sampling, Cluster Sampling.

Here we are using simple random sampling method by using ‘train\_test\_split’

Function , which helps to divide dataset into two half’s, one is trained data and rest is test data.

We are not using stratify sampling method , Since stratified sampling method needs to have categorical dependent/target variable. Which results into a classification problem. But here we have continuous dependent variable. So it is a regression problem.That is why we using simple random Sampling method.



### 2.5 Model Development

In our early stages of analysis during pre-processing we have processed our dataset. So that we can get most accurate predictions from our regression model. Now going ahead both dependent and independent variables are in shape for our model production. Hence, we need to analyse the data sets and generate model on top of processed dataset.

Dependent variable, in our case, ‘Absenteeism time in hours’ is continuous. predictive analysis that we can perform is time series or regression,Here we are using regression model to solve the problem.

Firstly we will use statistical model ‘Linear Regression’ on our dataset, which is a regression model, to predict the target variable ‘Absenteeism time in hours’.

Afterwards we will use decision tree regressor, random forest regressor, KNN. And then one by one we can apply several models and check the error rate using different techniques. And on basis of error rate we can select one model which will be more suitable for our dataset.

### 2.5.1 Model Evaluation

Now that we have a few models for predicting the target variable, we need to decide which one to choose. There are several criteria that exist for evaluating and comparing models. We can compare the models using any of the following criteria:

1. Predictive Performance

2. Interpretability

3. Computational Efficiency

In our model Interpretability and Computation Efficiency, do not hold much significance. Therefore we will use Predictive performance as the criteria to compare and evaluate models.

Predictive performance can be measured by comparing Predictions of the models with real values of the target variables, and calculating some average error measure and confusion matrix.

### 2.5.2 Error Measure

There are three primary metrics used to evaluate linear models. These are: Mean absolute error (MAE), Mean squared error (MSE), or Root mean squared error (RMSE).MAE: The easiest to understand. Represents average error

MSE: Similar to MAE but noise is exaggerated and larger errors are “punished”. It is harder to interpret than MAE as it’s not in base units, however, it is generally more popular.

RMSE: Most popular metric, similar to MSE, however, the result is square rooted to make it more interpretable as it’s in base units. It is recommended that RMSE be used as the primary metric to interpret your model.

We are using MAPE and RMSE for our model evaluation.

We our considering RMSE over MAPE because this is a regression model. And RMSE is more suitable for regression models.

### 2.5.3 Confusion Matrix

In case of classification models ,Predictive performance can be measured by comparing Predictions of the models with real values of the target variables, and calculating confusion matrix.

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. Here our model is regression model so for accuracy evaluation we can not use confusion matrix. So we have only used error matrix.

### 2.5.4 Cross Validation

The validation set approach

Cross Validation is a technique which involves reserving a particular sample of a dataset on which you do not train the model. Later, you test your model on this sample before finalizing it.

In this approach, we reserve 50% of the dataset for validation and the remaining 50% for model training.

### 2.6 Model Selection

### 2.6.1 Linear Regression Model

linear regression is a linear approach to modelling the relationship between a scalar response (or dependent variable) 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.

We have multiple independent variables explaining our dependent variable .So in this case we are doing multiple linear regression.In linear regression, we are attempting to build a model that allows us to predict the value of new data, given the training data used to train our model.

../Screen%20Shot%202018-11-02%20at%2012.41.38%20PM.png

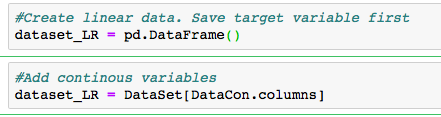
Here B0 is intercept, B1, B2…Bp are coefficients/weights of independent variables x1,x2….xp , where p is the number of independent variables and y is target/dependent variable.

We have two approaches for linear regression one is api from statsmodel and another is using linear\_model from scikit learn library.Here we are using statsmodel Since we get summary matrix in this.So we do not have to call score,coef or intercept individually.whereas we can get reasonable information about model with the help of summary matrix.

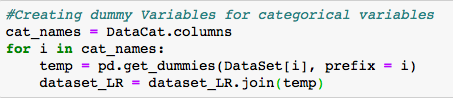
In statsmodel we are using OLS method , OLS stands for ordinary least square and the method ‘Least Square’ means that we are trying to fit a regression line that would minimise the squares of distance from the regression line.

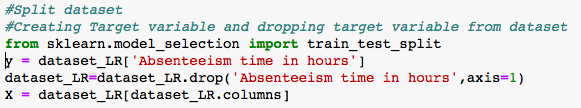
Here we are using statsmodels.api.OLS for regression problem. For model development we are following some steps:

Step1:



Step2:



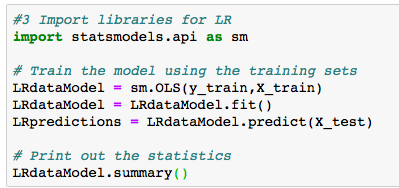
Step3:

Step4:

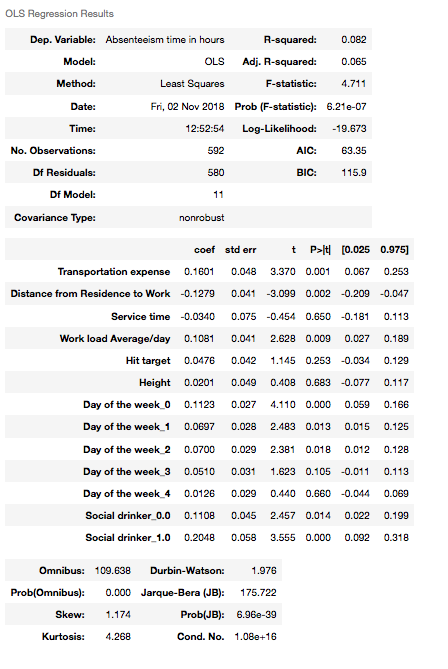
Here test data is 20% of the dataset and train data is 80%

../Screen%20Shot%202018-11-02%20at%204.42.20%20PM.png

Step5:



**Summary Statistics**



**Summary Statistics Description**

Summary statistics helps us to evaluate performance of our model based on some parameters.

From summary statistics matrix we can conclude some observations as:

Dep. Variable, Model, Method, Date, Time, No. Observations, Df model, Df Residuals, Covariance type. These parameters explain basic information regarding model .

F-statistics – It helps us to measure how significant the fit is. It is difference between actual values and predicted values. It should be as low as possible.

In our model F-statistics is 5.482 , Which explains less deviation between actual and predictive values.

Prob(F-statistics) – It is probability of F- statistics of model. It gives us overall performance of model, If probability is less than 0.05 then we can say model is reasonably good. In our model Prob(F-statistics) is ‘2.11e-08’ .

AIC – Alkalined information criterion , It adjusts the log likelihood based on the number of obsevations and complexity of model. In our model AIC is 78.75.

BIC - It should be higher than AIC. In our model BIC is 132.8 .

R-squared – It shows how much amount of variance it is giving for target variables.

Adj. R-squared – It will penalise the effect of extra variables in the model.

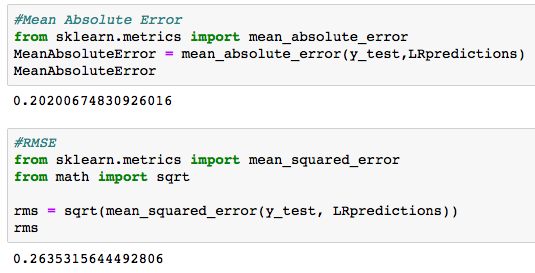
Adj. R-squared should be less than R-squared.

Coefficients – coefficients tells us the amount of information, each variable is carrying weight/explaining about target variables. In our model the coefficient which is highest is of variable Social drinker , Transportation expense, Work load Average/day , Day of the week as per summary statistics table. So we can conclude these variables are explaining much about target variable then rest of the variables.

Std-err – Standard error / deviation tells us the error of the estimate of coefficients.

t – t statistics measures how statistically significant the coefficient is because based on t- value we calculate p-value from t-table.

p-Value – If p-value is greater than 0.05 , So we can accept that the null hypothesis saying that this variable is not contributing much information to put into model. P-value should me greater than mode of t. As we can observe in our model we have p-values for all variables greater than t-value.

Step 6: 

Step 7:



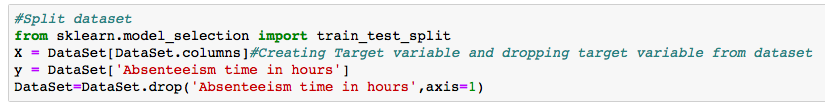
### 2.6.2 Decision Tree

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

A decision node has two or more branches, each representing values for the attribute tested. Leaf node represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

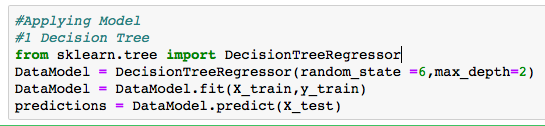
Here we are using Decision Tree Regressor for regression problem. For decision tree regressor model development we are following some steps:

Step 1: Sampling



../Screen%20Shot%202018-11-02%20at%204.49.20%20PM.png

Step 2:



Step 3:



Generated Code in tree1.dot file :

digraph Tree {

node [shape=box] ;

0 [label="Absenteeism time in hours <= 0.38\nmse = 0.068\nsamples = 592\nvalue = 0.299"] ;

1 [label="Absenteeism time in hours <= 0.147\nmse = 0.005\nsamples = 374\nvalue = 0.131"] ;

0 -> 1 [labeldistance=2.5, labelangle=45, headlabel="True"] ;

2 [label="mse = 0.002\nsamples = 222\nvalue = 0.083"] ;

1 -> 2 ;

3 [label="mse = 0.001\nsamples = 152\nvalue = 0.202"] ;

1 -> 3 ;

4 [label="Absenteeism time in hours <= 0.603\nmse = 0.046\nsamples = 218\nvalue = 0.586"] ;

0 -> 4 [labeldistance=2.5, labelangle=-45, headlabel="False"] ;

5 [label="mse = 0.0\nsamples = 168\nvalue = 0.47"] ;

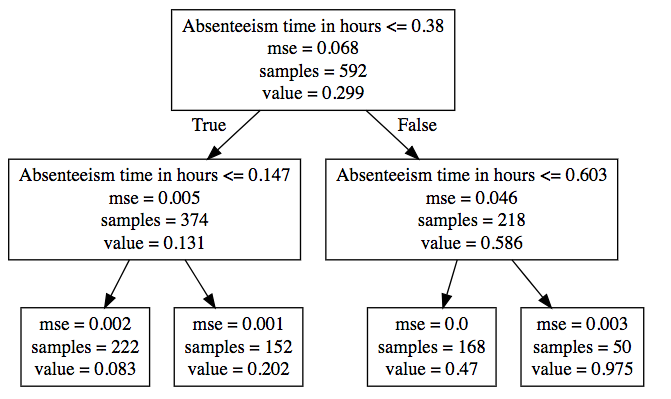
4 -> 5 ;

6 [label="mse = 0.003\nsamples = 50\nvalue = 0.975"] ;

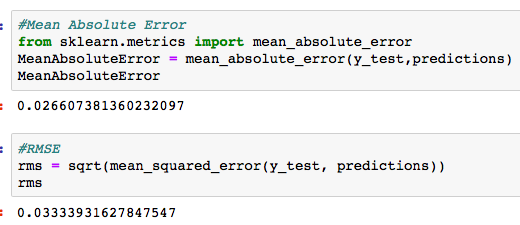
4 -> 6 ;

}

Nodes of Tree :



Step 4:



### 2.6.3 Random Forest

The random forest model is a type of additive model that makes predictions by combining decisions from a sequence of base models.

This broad technique of using multiple models to obtain better predictive performance is called model ensembling. In random forests, all the base models are constructed independently using a different subsample of the data.

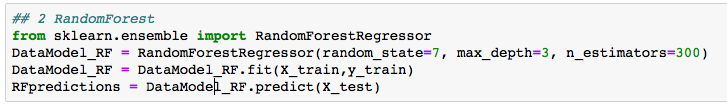
The random forest model is very good at handling tabular data with numerical features, or categorical features with fewer than hundreds of categories. Unlike linear models, random forests are able to capture non-linear interaction between the features and the target.

Here we are using Decision Forest Regressor for regression problem. For model development we are following some steps:

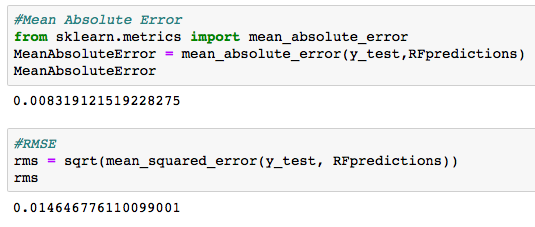
Step1:

../Screen%20Shot%202018-11-02%20at%204.53.19%20PM.png

Step2:



Step3:

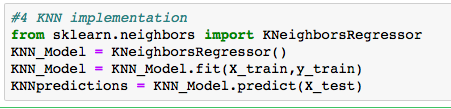


### 2.6.4 KNN

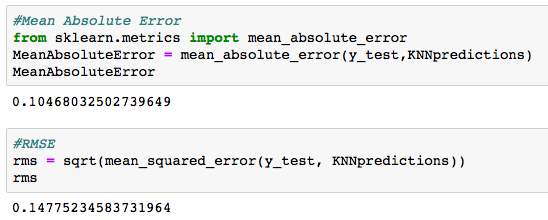
KNN can be used for both classification and regression problems. The algorithm uses ‘feature similarity’ to predict values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set.

Here we are using KNeighbors Regressor for regression problem. For model development we are following some steps:

Step1:../Screen%20Shot%202018-11-02%20at%204.56.08%20PM.png

Step2:

Step3:



### 2.6.5 Model Selection

You always start your model building most suitable model .Therefore we use statistical model first which is Linear Regression model. And then one by one we can apply several models and check the accuracy score and error measures.

Since we are solving regression problem our main focus is on error rate.

So on basis of error rate , we can select one model which will be more suitable for target variable prediction.

We have applied these four models on dataset, And we found error measures also from them .

1. Linear Regression Model

MAE : 20.20

RMSE : 26.35

1. Decision Tree Regression Model

MAE : 02.66

RMSE : 03.33

1. Random Forest Regression Model

MAE : 00.83

RMSE : 01.46

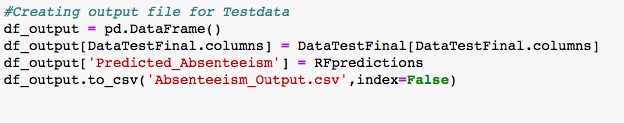
1. KNN Regression Model

MAE : 10.46

RMSE : 14.77

So we can conclude here from above results that error rate for Random Forest Regressor is less in compare to all other models.So Random forest regression model is most suitable model for the problem. And we can freeze Random Forest Regression model for the dataset.

### 2.7 Generate Output File



# **3.Conclusion**

We are concluding here by answering the question which are also got explained by our regression model.

Project Description and 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?

As first problem states that

## **3.1 “What changes company should bring to reduce the number of absenteeism?”**

As per analysis we have observed that here are various factors which are affecting employ absenteeism like ‘Day of the week’

* On starting two days(Mon,Tue) of the week Avg. 'Absenteeism time in hours' is higher than mid week days(wed,Thur).
* ‘Reason for absence’ is most of the time ‘ICD’ type comparatively ‘NOT ICD’.
* absenteeism is higher because in most of the cases ‘Reason for absence ‘ is ‘ICD’ which means employees have serious health related issues. Which cannot

be ignored or post ponded .

* To improve this , company can provide some solutions as:
  + Company can provide monthly health check ups
  + Company can start giving healthy breakfast
  + Company can have physical fitness activities to keep the employees

healthy

* + Company can also involve employees to some outdoor activities

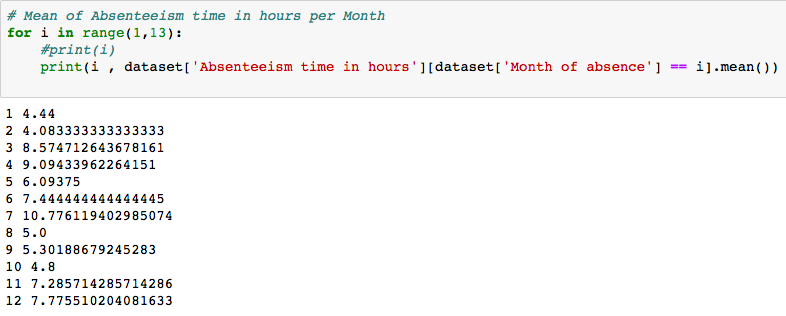
* This is visible that employee of old age may have some health/travel concerns. So for old age group people , some travel/health benefits can be offered to reduce absenteeism.
* As per analysis ‘Social drinker’ are more absent comparatively ‘Non Social Drinker’ . Which can be also reduced by giving motivational and self realization seminars frequently.

By using these techniques , the problem of absenteeism can be avoided or reduced due to which is related to health issues. Which is a major problem as per analysis.

## **3.2 “How much losses every month can we project in 2011 if same trend of absenteeism continues?”**

As per dataset absenteeism per month is given in below screenshot. So as per this

loses per month are as follows.



We have plotted one scatter plot, to show the loses[Absenteeism in hours ] of  every month depending upon independent variables from dataset.

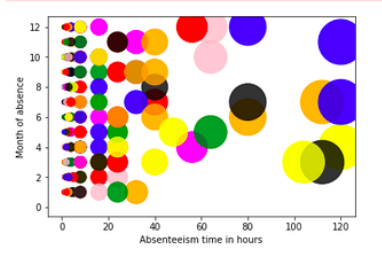
Here firstly we created a 'dict' in which we have mentioned all the features we have taken in our models and given them all different colors to know What is the impact they have on absenteeism every month.

And we have put them all in a list named col.

Then we have plotted a scatter plot between 'Absenteeism time in hours' and 'Month of absence' using col list .

And in this scatter plot colors explains all the features , y-axis explains every month and size of the bubble explains the feature which is causing total hours of absenteeism per month due to the particular feature .





## **3.3 Challenges Faced**

There are a number of important challenges that have been faced during development:

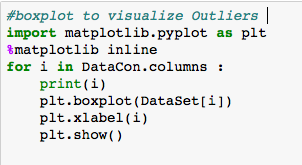
1. Figuring out what assumptions can be safely made about the data and the underlying system.
2. The data needed Exploratory Data Analysis and data preprocessing Since Dataset has missing values, outliers present in dataset and also we had subcategories explained in problem statement for some variables.
3. Apart missing data , there was also invalid data present in some of the variables. Which was handled under data pre processing.
4. Missing value imputation was done using two different techniques for categorical and continuous data to attain accuracy.
5. Feature selection has to be done individually for categorical and continuous data. And keeping only related variables which explain the dependent variable and are not collinear.
6. For preventing multicollinearity. We have dropped some variables , which were highly collinear.
7. For statistical model we have data in form of dummy variable for categorical data. But even after choosing very important variables we faced multicollinearity effect in Linear Regression Model.
8. In R code we have not created dummy variable to see the effects of multicollinearity reducing. And also we have used Vif in R to reduce multicollinearity.
9. R-squared will always increase as we add more features to the model, even if they are unrelated to the response.

* Selecting the model with the highest R-squared is not a reliable approach for choosing the best linear model.

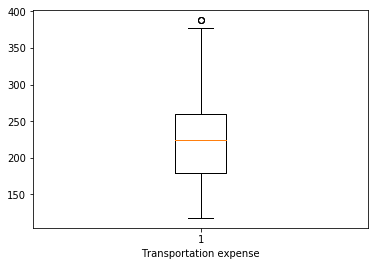
1. When we know what we are expected to predict and which assumptions we can make about the data, we can look at existing algorithms.
2. When choosing algorithms, we need to verify which algorithm or parameters work best on the data. Typically we use a validation set to check this. Of course, to judge which algorithm works best we need to look at different error measure or other ways to determine the quality of an algorithm.

# **Appendix – A**

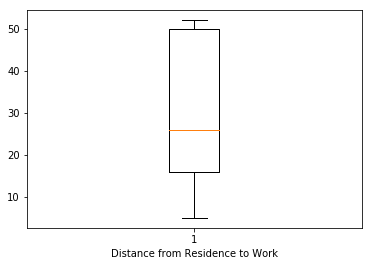
## **Outlier Analysis Using Boxplot Method**



Transportation expense



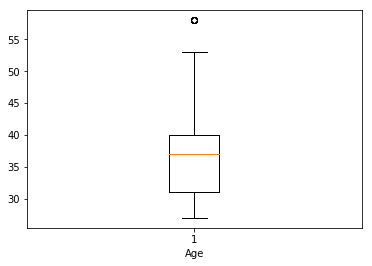
Distance from Residence to Work



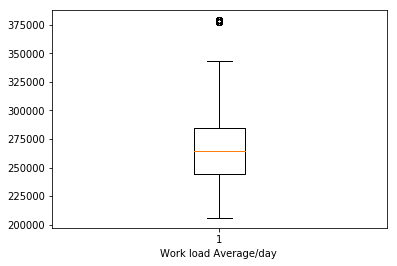
Service time



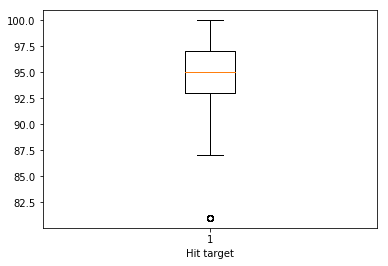
Age



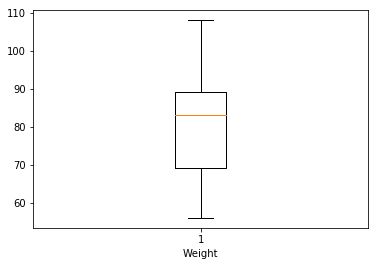
Work load Average/day



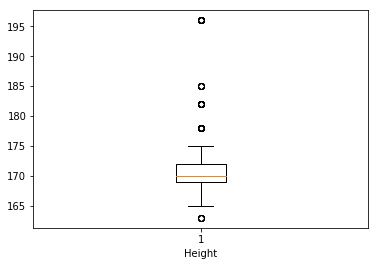
Hit target



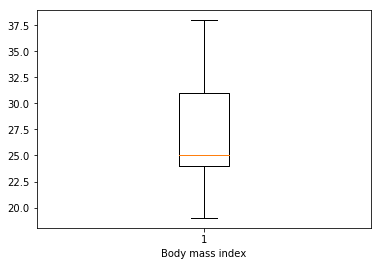
Weight



Height



Body mass index

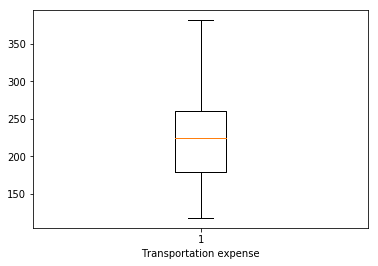


Absenteeism time in hours

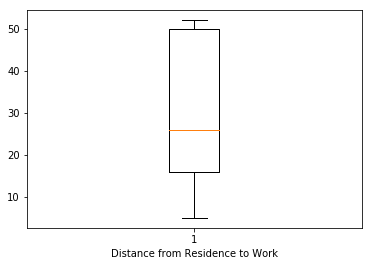


## **boxplots to visualize Outliers after replacing outliers**

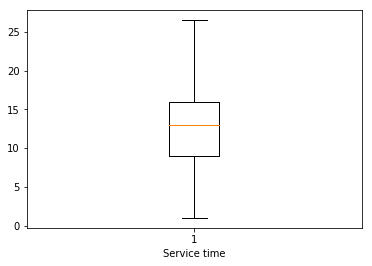
Transportation expense



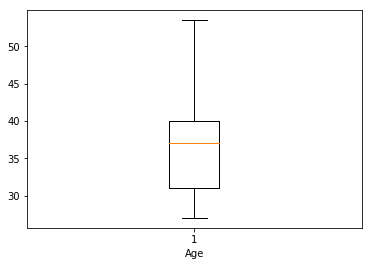
Distance from Residence to Work



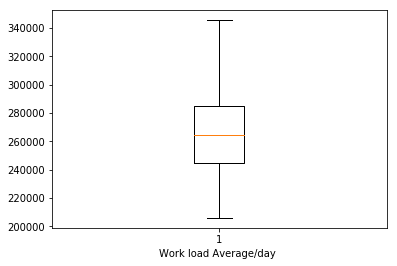
Service time



Age



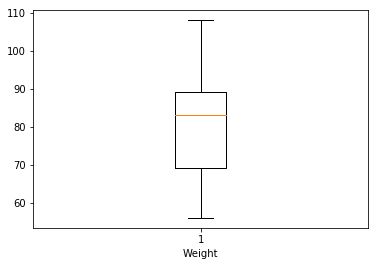
Work load Average/day



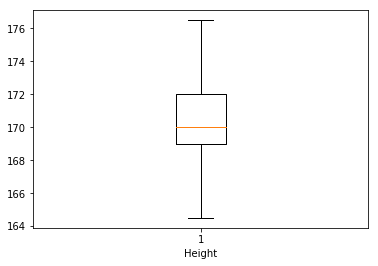
Hit target



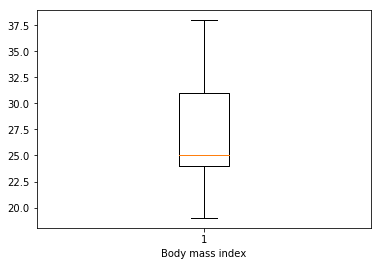
Weight



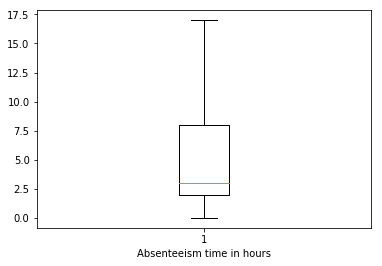
Height



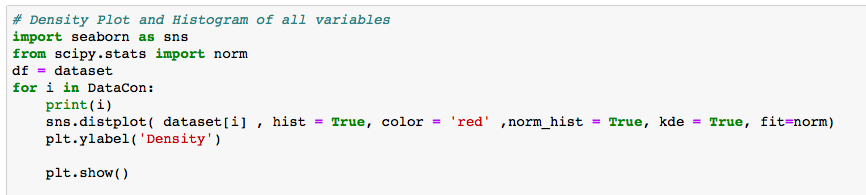
Body mass index



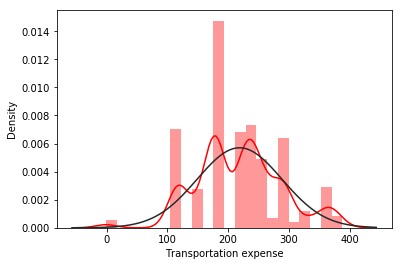
Absenteeism time in hours



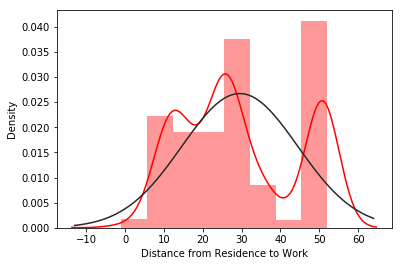
## **Density plot and Histogram of variables**



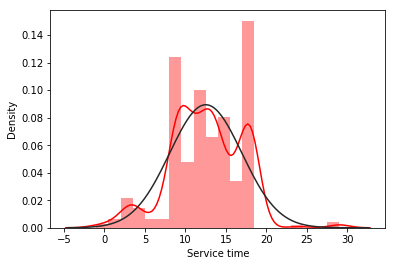
Transportation expense



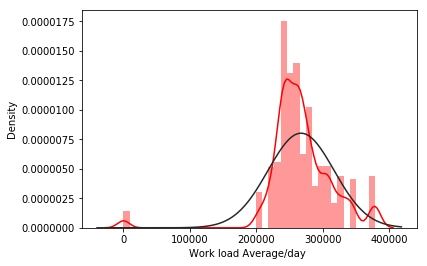
Distance from Residence to Work



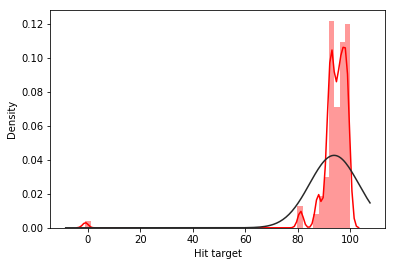
Service time



Work load Average/day



Hit target

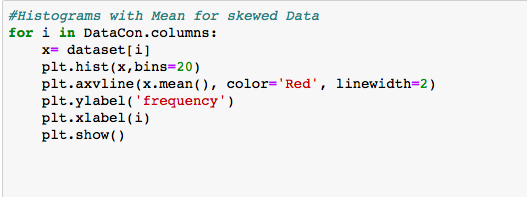


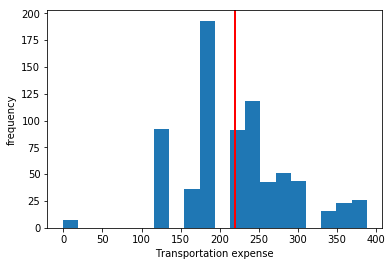
Height

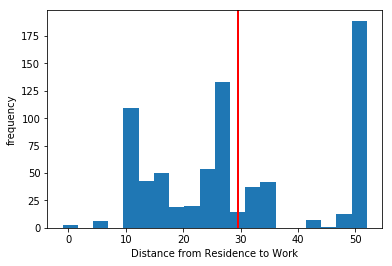


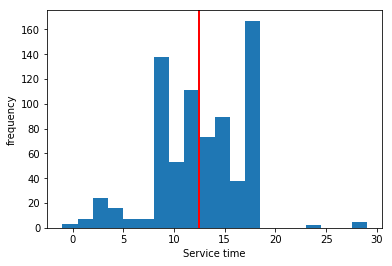
## 

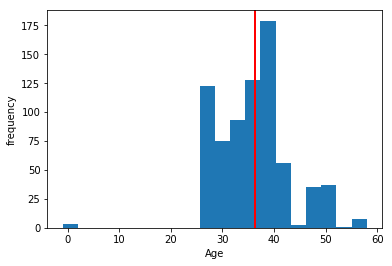
## **Histogram showing Mean and skewed data of variables**

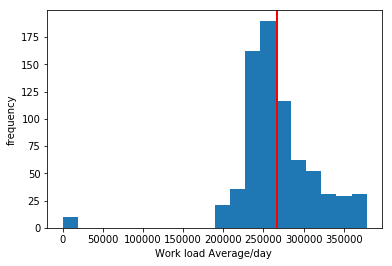


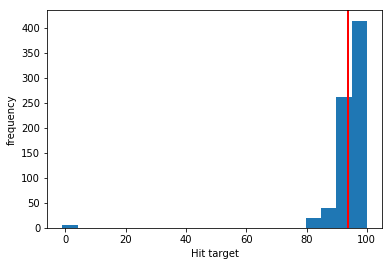


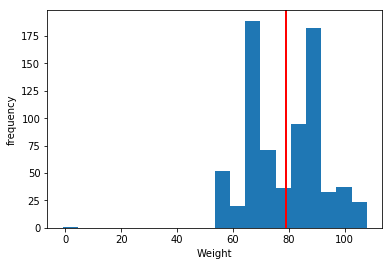


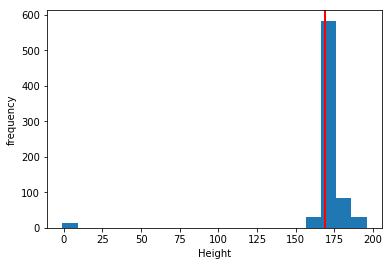


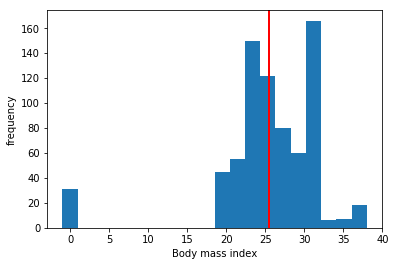




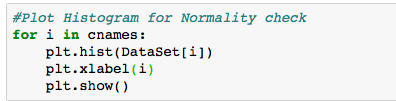


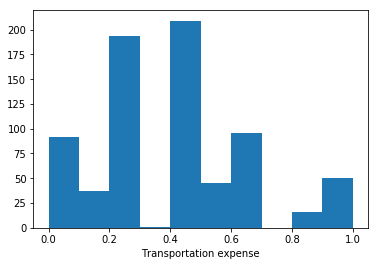


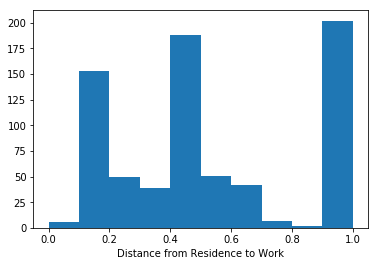




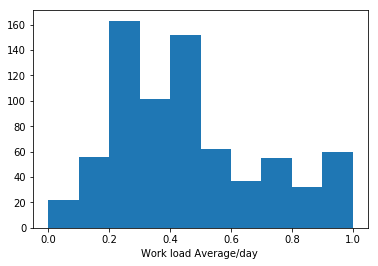
**Histograms after Normalizations**

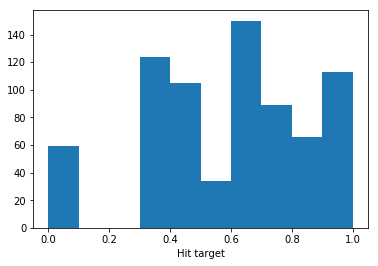


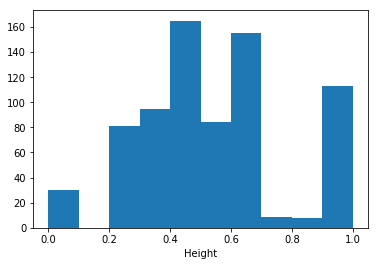














**Appendix – B**

# **Python Code**

**Problem : Employee Absenteeism**

In [ ]:

*#Load Libraries*

**import** **os**

**import** **pandas** **as** **pd**

**import** **numpy** **as** **np**

**from** **fancyimpute** **import** KNN

**from** **scipy.stats** **import** chi2\_contingency

In [ ]:

*#Loading dataSet*

os.chdir("/Users/bhartisharma/Desktop/Employee Absenteeism")

dataset = pd.read\_excel("Absenteeism\_at\_work\_Project.xls")

In [ ]:

dataset\_copy = pd.read\_excel("Absenteeism\_at\_work\_Project.xls")

**Data**

In [ ]:

dataset.head(2)

In [ ]:

*#Dimensions of Data*

dataset.shape

In [ ]:

*#Data Information*

dataset.info()

In [ ]:

*#Statistics of Data*

dataset.describe()

In [ ]:

*#Checking frequency of the employee who have maximum hours of ‘Absenteeism time in hours'*

dataset['Absenteeism time in hours'].value\_counts()

**Exploratory Data Analysis**

In [ ]:

*#DataCat is a dataframe containing all categorical variables from dataset*

DataCat = pd.DataFrame()

DataCat = dataset[['ID','Reason for absence','Month of absence','Day of the week','Seasons','Disciplinary failure',

'Education','Son','Social drinker','Social smoker','Pet']]

*#DataCon is a dataframe containing all continuous variables from dataset*

DataCon = pd.DataFrame()

DataCon = dataset[['Transportation expense','Distance from Residence to Work',

'Service time', 'Age', 'Work load Average/day ', 'Hit target','Weight', 'Height', 'Body mass index','Absenteeism time in hours']]

*#Converting some integer and float variables to categorical variable as per requirement*

dataset[DataCat.columns] = dataset[DataCat.columns].astype(object)

*#Assigning (-1) in continuous data to all NA and INF values for temporary basis for float datattype conversion to int data type. Missing value data will be handled as per process under the data preprocessing*

dataset[DataCon.columns] = dataset[DataCon.columns].replace(np.nan , -1)

*#Converting float variables to integer type Since there are no float values present in dataset*

dataset[DataCon.columns] = dataset[DataCon.columns].astype(int)

In [ ]:

dataset.dtypes

In [ ]:

*#categorising given data for variable 'Reason for absence' in two categories as per problem statement*

r,c =dataset.shape

**for** i **in** range(0,r):

*#print(i)*

**if**(dataset.loc[i,('Reason for absence')] != 0):

**if**(dataset.loc[i,('Reason for absence')] <= 21):

*#print(dataset['Reason for absence'][i])*

dataset.loc[i,('Reason for absence')] = 'ICD'

**elif**(dataset.loc[i,('Reason for absence')] >= 22):

dataset.loc[i,('Reason for absence')] = 'Not ICD'

In [ ]:

*#Categorising given data for Disciplinary failure , Social drinker , Social smoker*

**for** i **in** range(0,r):

**if**(dataset.loc[i,('Disciplinary failure')] == 0):

dataset.loc[i,('Disciplinary failure')] = 'No'

**elif**(dataset.loc[i,('Disciplinary failure')] == 1):

dataset.loc[i,('Disciplinary failure')] = 'Yes'

**for** i **in** range(0,r):

**if**(dataset.loc[i,('Social drinker')] == 0):

dataset.loc[i,('Social drinker')] = 'No'

**elif**(dataset.loc[i,('Social drinker')] == 1):

dataset.loc[i,('Social drinker')] = 'Yes'

**for** i **in** range(0,r):

**if**(dataset.loc[i,('Social smoker')] == 0):

dataset.loc[i,('Social smoker')] = 'No'

**elif**(dataset.loc[i,('Social smoker')] == 1):

dataset.loc[i,('Social smoker')] = 'Yes'

In [ ]:

*#Day of the week (Monday (2), Tuesday (3), Wednesday (4), Thursday (5), Friday (6))*

dataset['Day of the week']

**for** i **in** range(0,r):

**if**(dataset.loc[i,('Day of the week')] == 2):

dataset.loc[i,('Day of the week')] = 'Monday'

**elif**(dataset.loc[i,('Day of the week')] == 3):

dataset.loc[i,('Day of the week')] = 'Tuesday'

**elif**(dataset.loc[i,('Day of the week')] == 4):

dataset.loc[i,('Day of the week')] = 'Wednesday'

**elif**(dataset.loc[i,('Day of the week')] == 5):

dataset.loc[i,('Day of the week')] = 'Thursday'

**elif**(dataset.loc[i,('Day of the week')] == 6):

dataset.loc[i,('Day of the week')] = 'Friday'

In [ ]:

*#Seasons (summer (1), autumn (2), winter (3), spring (4))*

**for** i **in** range(0,r):

**if**(dataset.loc[i,('Seasons')] == 1):

dataset.loc[i,('Seasons')] = 'summer'

**elif**(dataset.loc[i,('Seasons')] == 2):

dataset.loc[i,('Seasons')] = 'autumn'

**elif**(dataset.loc[i,('Seasons')] == 3):

dataset.loc[i,('Seasons')] = 'winter'

**elif**(dataset.loc[i,('Seasons')] == 4):

dataset.loc[i,('Seasons')] = 'spring'

In [ ]:

*# Education (high school (1), graduate (2), postgraduate (3), master and doctor (4))*

**for** i **in** range(0,r):

**if**(dataset.loc[i,('Education')] == 1):

dataset.loc[i,('Education')] = 'high school'

**elif**(dataset.loc[i,('Education')] == 2):

dataset.loc[i,('Education')] = 'graduate'

**elif**(dataset.loc[i,('Education')] == 3):

dataset.loc[i,('Education')] = 'postgraduate'

**elif**(dataset.loc[i,('Education')] == 4):

dataset.loc[i,('Education')] = 'master and doctor'

In [ ]:

dataset.head(2)

In [ ]:

dataset.to\_csv('Absenteeism\_EDA.csv',index=**False**)

In [ ]:

*# Density Plot and Histogram of all variables*

**import** **seaborn** **as** **sns**

**from** **scipy.stats** **import** norm

df = dataset

**for** i **in** DataCon:

print(i)

sns.distplot( dataset[i] , hist = **True**, color = 'red' ,norm\_hist = **True**, kde = **True**, fit=norm)

plt.ylabel('Density')

plt.show()

In [ ]:

*# Sum of Absenteeism time in hours per Month*

**for** i **in** range(1,13):

*#print(i)*

print(i , dataset['Absenteeism time in hours'][dataset['Month of absence'] == i].sum())

In [ ]:

*# Mean of Absenteeism time in hours per Month*

**for** i **in** range(1,13):

*#print(i)*

print(i , dataset['Absenteeism time in hours'][dataset['Month of absence'] == i].mean())

**Data Preprocessing**

**Missing Values Imputation**

In [ ]:

*#Dividing Categorical variables into two dataframes*

*#Variables like ID,Reason for absence,Day of the week ,Seasons ,Education. Since in this case 0 shows loss of*

*#information,So we will replace them with NaN values and then impute them*

DataCat\_1 = pd.DataFrame()

DataCat\_1 = DataCat[['ID','Reason for absence','Month of absence','Day of the week','Seasons','Education']]

In [ ]:

*#Variables like 'Disciplinary failure','Son', 'Social drinker','Social smoker', 'Pet'*

*#They include 0 as important information so we are keeping them in other dataset*

DataCat\_2 = pd.DataFrame()

DataCat\_2 = DataCat[['Disciplinary failure', 'Social drinker','Social smoker']]

In [ ]:

*#DataCat\_1*

*#replace 0 with NA*

**for** i **in** range(0, DataCat\_1.shape[1]):

*#print(DataCat\_1.iloc[:,i] )*

DataCat\_1.iloc[:,i] = DataCat\_1.iloc[:,i].replace(0, np.nan)

In [ ]:

DataCat\_1.isnull().sum()

In [ ]:

*#DataCat\_1*

*#Assigning levels to the categories*

*#lis = []*

**for** i **in** range(0, DataCat\_1.shape[1]):

DataCat\_1.iloc[:,i] = pd.Categorical(DataCat\_1.iloc[:,i])

DataCat\_1.iloc[:,i] = DataCat\_1.iloc[:,i].cat.codes

DataCat\_1.iloc[:,i] = DataCat\_1.iloc[:,i].astype('object')

In [ ]:

*#DataCat\_1*

*#replace -1 with NA to impute*

**for** i **in** range(0, DataCat\_1.shape[1]):

DataCat\_1.iloc[:,i] = DataCat\_1.iloc[:,i].replace(-1, np.nan)

In [ ]:

*#DataCat\_2*

DataCat\_2 = DataCat\_2.replace('Yes',1)

DataCat\_2 = DataCat\_2.replace('No',0)

*#Add remaining two variables 'Son' and 'Pet' in categorical 2*

DataCat\_2 = DataCat\_2.join(DataCat[['Son','Pet']])

In [ ]:

*#Creating New DataFrame DataSet to copy Continuous(DataCon) and Categorical DataSet after imputing missing values seprately*

DataSet = pd.DataFrame()

DataSet = DataCon

DataSet.head()

In [ ]:

*#Create dataframe with missing percentage*

*#For Continuous Variables*

MisVal = pd.DataFrame(DataSet.isnull().sum())

*#Reset index*

MisVal = MisVal.reset\_index()

*#Rename variable*

MisVal = MisVal.rename(columns = {'index': 'Variables', 0: 'Missing\_percentage'})

*#Calculate percentage*

MisVal['Missing\_percentage'] = (MisVal['Missing\_percentage']/len(DataSet))\*100

print(MisVal)

In [ ]:

*#Firstly Imputating missing values for Continuous Variables*

*#create missing value*

print(DataSet['Distance from Residence to Work'].loc[4])

DataSet['Distance from Residence to Work'].loc[4] = np.nan

*#Actual Value - 36*

In [ ]:

*#Impute with mean*

*#DataSet['Distance from Residence to Work'] = DataSet['Distance from Residence to Work'].fillna(DataSet['Distance from Residence to Work'].mean())*

*#print(DataSet['Distance from Residence to Work'].loc[4])*

In [ ]:

*#Impute with median*

*#DataSet['Distance from Residence to Work'] = DataSet['Distance from Residence to Work'].fillna(DataSet['Distance from Residence to Work'].median())*

*#print(DataSet['Distance from Residence to Work'].loc[4])*

In [ ]:

*#Apply KNN imputation algorithm for ontinuous Variables*

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

In [ ]:

DataSet.isnull().sum()

In [ ]:

*#For Categorical Variables Imputting with Mode*

DataCat\_1 = DataCat\_1.replace(np.nan,DataCat\_1.mode())

DataCat\_1.isnull().sum()

In [ ]:

*#Adding Categorical variables to main dataset after imputing*

DataSet = DataSet.join(DataCat\_1)

DataSet.head(2)

In [ ]:

*#DataCat\_2*

DataCat\_2 = DataCat\_2.replace(np.nan,DataCat\_2.mode())

DataCat\_2.isnull().sum()

In [ ]:

*#Adding Categorical variables to main dataset after imputing*

DataSet = DataSet.join(DataCat\_2)

DataSet.head(2)

In [ ]:

DataSet.isnull().sum()

**Outlier Analysis**

In [ ]:

*#boxplot to visualize Outliers*

**import** **matplotlib.pyplot** **as** **plt**

%**matplotlib** inline

**for** i **in** DataCon.columns :

print(i)

plt.boxplot(DataSet[i])

plt.xlabel(i)

plt.show()

In [ ]:

*# Outlier Analysis*

cnames = DataCon.columns

**for** col **in** cnames:

percentile = np.percentile(DataSet.loc[:,col],[75,25])

percentile = DataSet[col].quantile([0.25,0.75]).values

iqr = percentile[1] - percentile[0]

minimum = percentile[0] - (iqr\*1.5)

maximum = percentile[1] + (iqr\*1.5)

*#print(col,maximum,minimum)*

DataSet[col][DataSet[col] <= minimum] = minimum

DataSet[col][DataSet[col] >= maximum] = maximum

In [ ]:

*#Plot boxplot to visualize Outliers after replacing outliers*

**import** **matplotlib.pyplot** **as** **plt**

%**matplotlib** inline

**for** i **in** DataCon.columns :

print(i)

plt.boxplot(DataSet[i])

plt.xlabel(i)

plt.show()

**Feature selection**

In [ ]:

*#Correlation plot for Continuous variables*

**import** **matplotlib.pyplot** **as** **plt**

**import** **seaborn** **as** **sns**

cnames = DataCon.columns

df\_corr = DataCon.loc[:,cnames]

*#Set the width and hieght of the plot*

f, ax = plt.subplots(figsize=(7, 5))

*#Generate correlation matrix*

corr = df\_corr.corr()

*#Plot using seaborn library*

sns.heatmap(corr, mask=np.zeros\_like(corr, dtype=np.bool), cmap=sns.diverging\_palette(220, 10, as\_cmap=**True**),

square=**True**, ax=ax)

In [ ]:

*#Continuous Variables selection using corr()function*

DataCon\_corr = DataCon.corr()

*#from continuous variables variables*

DataCon\_corrVar = DataCon\_corr[DataCon\_corr > 0.5]

In [ ]:

DataCon\_corrVar

In [ ]:

*#Making a list DataConCorrVar,which contains values only greater than 0.5*

**import** **numpy** **as** **np**

*#DataCon\_corrVar = DataCon\_corrVar.replace(np.nan,-1)*

DataConCorrVar = []

**for** i **in** range(0,DataCon\_corrVar.shape[1]):

**for** j **in** range(0,DataCon\_corrVar.shape[1]):

**if**(i != j):

**if**((DataCon\_corrVar.iloc[i,j] > 0.5).any(axis=0)):

*#print(DataCon\_corrVar.columns[i]+ " , "+DataCon\_corrVar.columns[j] + " :",DataCon\_corrVar.iloc[i,j])*

DataConCorrVar.append((DataCon\_corrVar.columns[i] , DataCon\_corrVar.columns[j] ,DataCon\_corrVar.iloc[i,j] ))

In [ ]:

*#Convert to DataFrame*

DataConCorrVar = pd.DataFrame(DataConCorrVar)

DataConCorrVar

In [ ]:

DataSet.columns

In [ ]:

DataSet =DataSet.drop(['Age','Weight','Body mass index'],axis=1)

DataCon =DataCon.drop(['Age','Weight','Body mass index'],axis=1)

In [ ]:

*#Feature Selection for categorical data*

In [ ]:

DataCat.columns

In [ ]:

*#If p-value < 0.05: significant result, reject null hypothesis (H0), dependent.*

*#If independent variables are dependent to each other we can drop one of them .*

*#If p-value > 0.05: not significant result, fail to reject null hypothesis (H0), independent.*

*#1*

corrVar = []

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['ID'],DataCat['Reason for absence']))

print(p)

*#Drop ID*

DataSet = DataSet.drop(['ID'],axis =1)

DataCat = DataCat.drop(['ID'],axis=1)

corrVar.append('Reason for absence')

In [ ]:

*#2*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Month of absence'],DataCat['Reason for absence']))

print(p)

*#drop 'Month of absence'*

corrVar.append('Month of absence')

corrVar

*#Drop 'Reason for absence'*

DataSet = DataSet.drop(['Reason for absence'],axis =1)

DataCat = DataCat.drop(['Reason for absence'],axis=1)

corrVar.remove('Reason for absence')

corrVar

In [ ]:

*#3*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Month of absence'],DataCat['Day of the week']))

print(p)

corrVar.append('Day of the week')

corrVar

In [ ]:

*#4*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Day of the week'],DataCat['Seasons']))

print(p)

corrVar.append('Seasons')

corrVar

In [ ]:

*#5*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Seasons'],DataCat['Disciplinary failure']))

print(p)

DataSet = DataSet.drop(['Disciplinary failure'],axis =1)

DataCat = DataCat.drop(['Disciplinary failure'],axis=1)

In [ ]:

*#6*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Education'],DataCat['Seasons']))

corrVar.append('Education')

print(p)

corrVar

In [ ]:

*#7*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Son'],DataCat['Education']))

print(p)

*#drop 'Son'*

DataSet = DataSet.drop(['Son'],axis =1)

DataCat = DataCat.drop(['Son'],axis=1)

In [ ]:

*#8*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Social drinker'],DataCat['Day of the week']))

print(p)

corrVar

In [ ]:

*#9*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Social drinker'],DataCat['Education']))

print(p)

corrVar.append('Social drinker')

DataSet = DataSet.drop(['Education'],axis =1)

DataCat = DataCat.drop(['Education'],axis=1)

corrVar

In [ ]:

*#10*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Social smoker'],DataCat['Social drinker']))

print(p)

DataSet = DataSet.drop(['Social smoker'],axis =1)

DataCat = DataCat.drop(['Social smoker'],axis=1)

In [ ]:

*#11*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Pet'],DataCat['Social drinker']))

print(p)

DataSet = DataSet.drop(['Pet'],axis =1)

DataCat = DataCat.drop(['Pet'],axis=1)

In [ ]:

*# Now we have got 4 variables , and we will check correlation among them*

corrVar

In [ ]:

*#12*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Month of absence'],DataCat['Day of the week']))

print(p)

*#corrVar*

In [ ]:

*#13*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Month of absence'],DataCat['Seasons']))

print(p)

corrVar.remove('Seasons')

corrVar

DataSet = DataSet.drop(['Seasons'],axis =1)

DataCat = DataCat.drop(['Seasons'],axis=1)

In [ ]:

*#14*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Day of the week'],DataCat['Social drinker']))

print(p)

corrVar

In [ ]:

*#15*

chi,p,def1,ex = chi2\_contingency(pd.crosstab(DataCat['Month of absence'],DataCat['Social drinker']))

print(p)

corrVar

DataSet = DataSet.drop(['Month of absence'],axis =1)

DataCat = DataCat.drop(['Month of absence'],axis=1)

In [ ]:

DataSet.head()

**Feature Scaling**

In [ ]:

*#Nomalisation*

cnames = DataCon.columns

**for** i **in** cnames:

DataSet[i] = (DataSet[i] - min(DataSet[i]))/(max(DataSet[i]) - min(DataSet[i]))

In [ ]:

DataSet.head(2)

**Model development**

Linear Regression Model

In [ ]:

*#Create linear data. Save target variable first*

dataset\_LR = pd.DataFrame()

*#Add continous variables*

dataset\_LR = DataSet[DataCon.columns]

*#Creating dummy Variables for categorical variables*

cat\_names = DataCat.columns

**for** i **in** cat\_names:

temp = pd.get\_dummies(DataSet[i], prefix = i)

dataset\_LR = dataset\_LR.join(temp)

In [ ]:

*#Split dataset*

*#Creating Target variable and dropping target variable from dataset*

**from** **sklearn.model\_selection** **import** train\_test\_split

y = DataSet['Absenteeism time in hours']

dataset\_LR=dataset\_LR.drop('Absenteeism time in hours',axis=1)

X = dataset\_LR[dataset\_LR.columns]

In [ ]:

*#Sampling*

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state = 6)

In [ ]:

*#3 Import libraries for LR*

**import** **statsmodels.api** **as** **sm**

*# Train the model using the training sets*

LRdataModel = sm.OLS(y\_train,X\_train)

LRdataModel = LRdataModel.fit()

LRpredictions = LRdataModel.predict(X\_test)

*# Print out the statistics*

LRdataModel.summary()

Error Measure

In [ ]:

*#Mean Absolute Error*

**from** **sklearn.metrics** **import** mean\_absolute\_error

MeanAbsoluteError = mean\_absolute\_error(y\_test,LRpredictions)

MeanAbsoluteError

In [ ]:

*#RMSE*

**from** **sklearn.metrics** **import** mean\_squared\_error

**from** **math** **import** sqrt

rms = sqrt(mean\_squared\_error(y\_test, LRpredictions))

rms

In [ ]:

*#Cross Validation Technique*

*#Sampling 4*

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.5,random\_state = 6,)

In [ ]:

*# Train the model using the training sets*

LRdataModel = sm.OLS(y\_train,X\_train)

LRdataModel = LRdataModel.fit()

LRpredictions = LRdataModel.predict(X\_test)

*# Print out the statistics*

LRdataModel.summary()

In [ ]:

*#Mean Absolute Error*

**from** **sklearn.metrics** **import** mean\_absolute\_error

MeanAbsoluteError = mean\_absolute\_error(y\_test,LRpredictions)

MeanAbsoluteError

In [ ]:

*#RMSE*

**from** **sklearn.metrics** **import** mean\_squared\_error

**from** **math** **import** sqrt

rms = sqrt(mean\_squared\_error(y\_test, LRpredictions))

rms

In [ ]:

*#1 Decision Tree*

In [ ]:

*#convert categorical variables to codes, So that we can easily compute and impute easily*

**for** i **in** range(0,DataSet.shape[1]):

**if**(DataSet.iloc[:,i].dtypes == 'object'):

DataSet.iloc[:,i] = pd.Categorical(dataset.iloc[:,i])

dataset.iloc[:,i] = dataset.iloc[:,i].cat.codes

In [ ]:

*#Split dataset*

**from** **sklearn.model\_selection** **import** train\_test\_split

X = DataSet[DataSet.columns]

*#Creating Target variable and dropping target variable from dataset*

y = DataSet['Absenteeism time in hours']

DataSet=DataSet.drop('Absenteeism time in hours',axis=1)

In [ ]:

*#Sampling*

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state = 6)

In [ ]:

*#Applying Model*

**from** **sklearn.tree** **import** DecisionTreeRegressor

DataModel = DecisionTreeRegressor(random\_state =6,max\_depth=2)

DataModel = DataModel.fit(X\_train,y\_train)

predictions = DataModel.predict(X\_test)

In [ ]:

*#Create dot file to visualise tree #http://webgraphviz.com/*

**from** **sklearn** **import** tree

df = tree.export\_graphviz(DataModel, out\_file='tree1.dot' ,feature\_names = X.columns)

In [ ]:

*#Mean Absolute Error*

**from** **sklearn.metrics** **import** mean\_absolute\_error

MeanAbsoluteError = mean\_absolute\_error(y\_test,predictions)

MeanAbsoluteError

In [ ]:

*#RMSE*

rms = sqrt(mean\_squared\_error(y\_test, predictions))

rms

In [ ]:

*#2 RandomForest*

In [ ]:

*#Sampling*

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state = 6)

In [ ]:

*## 2 RandomForest*

**from** **sklearn.ensemble** **import** RandomForestRegressor

DataModel\_RF = RandomForestRegressor(random\_state=7, max\_depth=3, n\_estimators=300)

DataModel\_RF = DataModel\_RF.fit(X\_train,y\_train)

RFpredictions = DataModel\_RF.predict(X\_test)

In [ ]:

*#Mean Absolute Error*

**from** **sklearn.metrics** **import** mean\_absolute\_error

MeanAbsoluteError = mean\_absolute\_error(y\_test,RFpredictions)

MeanAbsoluteError

In [ ]:

*#RMSE*

rms = sqrt(mean\_squared\_error(y\_test, RFpredictions))

rms

KNN Model

In [ ]:

*#Sampling*

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.2,random\_state = 6)

In [ ]:

*#4 KNN implementation*

**from** **sklearn.neighbors** **import** KNeighborsRegressor

KNN\_Model = KNeighborsRegressor()

KNN\_Model = KNN\_Model.fit(X\_train,y\_train)

KNNpredictions = KNN\_Model.predict(X\_test)

In [ ]:

*#Mean Absolute Error*

**from** **sklearn.metrics** **import** mean\_absolute\_error

MeanAbsoluteError = mean\_absolute\_error(y\_test,KNNpredictions)

MeanAbsoluteError

In [ ]:

*#RMSE*

rms = sqrt(mean\_squared\_error(y\_test, KNNpredictions))

rms

Random Forest

In [ ]:

**from** **sklearn.ensemble** **import** RandomForestRegressor

DataModel\_RF = RandomForestRegressor(random\_state=7, max\_depth=3, n\_estimators=300)

DataModel\_RF = DataModel\_RF.fit(X\_train,y\_train)

RFpredictions = DataModel\_RF.predict(X)

In [ ]:

*#Creating output file for Testdata*

DataTestFinal= pd.read\_excel("Absenteeism\_at\_work\_Project.xls")

In [ ]:

*#Creating output file for Testdata*

df\_output = pd.DataFrame()

df\_output[DataTestFinal.columns] = DataTestFinal[DataTestFinal.columns]

df\_output['Predicted\_Absenteeism'] = RFpredictions

df\_output.to\_csv('Absenteeism\_Output.csv',index=**False**)

Scatter plot showing Loss Per Month

In [ ]:

*#Scatter plot for*

*#creating dictionary dict for holding important independent variables*

dict = {

'Transportation expense': 'red',

'Distance from Residence to Work':'blue',

'Service time': 'green' ,

'Height':'pink' ,

'Work load Average/day' : 'yellow' ,

'Hit target' : 'magenta' ,

'Day of the week' : 'black',

'Social drinker' : 'orange'

}

*#Creating list to hold dict values*

col = []

**for** i **in** dict :

col.append(dict[i])

In [ ]:

*#Creating a new dataframe df*

df = pd.DataFrame(dataset['Absenteeism time in hours'])

**if**(df.iloc[:,0].dtypes=='object'):

df.iloc[:,0]=pd.Categorical(df.iloc[:,0])

df.iloc[:,0] = df.iloc[:,0].cat.codes

In [ ]:

*#Scatter plot between 'Absenteeism time in hours' , 'Month of absence' and using cols and pop to show intensity of absenteeism as per featues and per month*

**import** **matplotlib.pyplot** **as** **plt**

pop = dataset['Absenteeism time in hours']

plt.scatter(df['Absenteeism time in hours'], dataset['Month of absence'],c = col , s = np.array(pop) \* 20 , alpha=0.8)

plt.xlabel('Absenteeism time in hours')

plt.ylabel('Month of absence')

plt.show()

# 

# **Appendix – C**

# **R-Code**

*setwd("/Users/bhartisharma/Desktop/Employee Absenteeism")*

*#Load Libraries*

*x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "unbalanced", "C50", "dummies", "e1071", "Information",*

*"MASS", "rpart", "gbm", "ROSE", 'sampling', 'DataCombine', 'inTrees')*

*#install.packages(x)*

*lapply(x, require, character.only = TRUE)*

*rm(x)*

*#Load Data*

*#install.packages("xlsx")*

*library(xlsx)*

*dataset = read.xlsx("Absenteeism\_at\_work\_Project.xls",sheetIndex =1,header = T)*

*#Converting dataset into a dataframe*

*dataset<-as.data.frame(dataset)*

*View(dataset)*

*#Exploratory data analysis*

*#Checking the default datatypes of dataset*

*str(dataset)*

*#statistics of dataset*

*summary(dataset)*

*#Creating two different dataframes DataCat and DataCon which contains categorical and continuos variables respectively as explained in the report*

*DataCon = data.frame()*

*DataCon = subset(dataset, select = c("Transportation.expense","Distance.from.Residence.to.Work" ,"Service.time","Age","Work.load.Average.day.","Hit.target","Weight" ,"Height","Body.mass.index","Absenteeism.time.in.hours"))*

*View(DataCon)*

*DataCat = data.frame()*

*DataCat = subset(dataset , select = c("ID","Reason.for.absence","Month.of.absence","Day.of.the.week","Seasons","Disciplinary.failure","Education","Son","Social.drinker","Social.smoker","Pet") )*

*View(DataCat)*

*#Dividing Categorical variables into two dataframes*

*#Variables like ID,Reason for absence,Day of the week ,Seasons ,Education. Since in this case 0 shows loss of information,So we will replace them with Nan values and then impute them*

*DataCat\_1 = data.frame()*

*DataCat\_1 = subset(DataCat , select = c('ID','Reason.for.absence','Month.of.absence','Day.of.the.week','Seasons','Education'))*

*View(DataCat\_1)*

*#Variables like 'Disciplinary failure','Son', 'Social drinker','Social smoker', 'Pet' include 0 as important information so we are keeping them in other dataset*

*DataCat\_2 = data.frame()*

*DataCat\_2 = subset(DataCat, select = c('Disciplinary.failure', 'Social.drinker','Social.smoker'))*

*View(DataCat\_2)*

*#DataCat\_1*

*#replace 0 with NA*

*DataCat\_1[DataCat\_1 == 0] <- NA*

*View(DataCat\_1)*

*##Assigning (-1) to all NA and INF values for temporary basis for float datattype conversion to int data type. Missing value data will be handled as per process under the data preprocessing*

*dataset[is.na(dataset)] = -1*

*View(dataset)*

*#Missing Values*

*#missing\_val = data.frame(apply(dataset,2,function(x){sum(is.na(x))}))*

*#missing\_val*

*#Converting required float variables to integer*

*for (i in colnames(DataCon))*

*{ print(i)*

*dataset[,i] = as.integer(dataset[,i])*

*}*

*#Checking the converted datatypes of dataset*

*str(dataset)*

*#categorising given data for variable 'Reason for absence' in two categories as per problem statement*

*#dataset$Reason.for.absence[is.na(dataset$Reason.for.absence)] = -1*

*for (i in c(1,2:nrow(dataset)))*

*{*

*if(dataset[i,'Reason.for.absence'] > 0)*

*{*

*if(dataset[i,('Reason.for.absence')] <= 21)*

*{*

*dataset[i,('Reason.for.absence')] = 'ICD'*

*}*

*else*

*{*

*if(dataset[i,('Reason.for.absence')] >= 22)*

*{*

*dataset[i,('Reason.for.absence')] = 'Not ICD'*

*}*

*}*

*}*

*}*

*View(dataset['Reason.for.absence'])*

*#Categorising given data for Disciplinary failure , Social drinker , Social smoker*

*for (i in c(1,2:nrow(dataset)))*

*{*

*if(dataset[i,('Disciplinary.failure')] == 0)*

*{*

*dataset[i,('Disciplinary.failure')] = 'No'*

*}*

*else{*

*if(dataset[i,('Disciplinary.failure')] == 1)*

*{*

*dataset[i,('Disciplinary.failure')] = 'Yes'*

*}}*

*}*

*for (i in c(1,2:nrow(dataset)))*

*{*

*if(dataset[i,('Social.drinker')] == 0)*

*{*

*dataset[i,('Social.drinker')] = 'No'*

*}*

*else{*

*if(dataset[i,('Social.drinker')] == 1)*

*{*

*dataset[i,('Social.drinker')] = 'Yes'*

*}}*

*}*

*for (i in c(1,2:nrow(dataset)))*

*{*

*if(dataset[i,('Social.smoker')] == 0)*

*{*

*dataset[i,('Social.smoker')] = 'No'*

*}*

*else{*

*if(dataset[i,('Social.smoker')] == 1)*

*{*

*dataset[i,('Social.smoker')] = 'Yes'*

*}}*

*}*

*View(dataset)*

*#Day of the week (Monday (2), Tuesday (3), Wednesday (4), Thursday (5), Friday (6))*

*for (i in c(1,2:nrow(dataset)))*

*{*

*if(dataset[i,('Day.of.the.week')] == 2)*

*{*

*dataset[i,('Day.of.the.week')] = 'Monday'*

*}*

*if(dataset[i,('Day.of.the.week')] == 3)*

*{*

*dataset[i,('Day.of.the.week')] = 'Tuesday'*

*}*

*if(dataset[i,('Day.of.the.week')] == 4)*

*{*

*dataset[i,('Day.of.the.week')] = 'Wednesday'*

*}*

*if(dataset[i,('Day.of.the.week')] == 5)*

*{*

*dataset[i,('Day.of.the.week')] = 'Thursday'*

*}*

*if(dataset[i,('Day.of.the.week')] == 6)*

*{*

*dataset[i,('Day.of.the.week')] = 'Friday'*

*}*

*}*

*#Seasons (summer (1), autumn (2), winter (3), spring (4))*

*for (i in c(1,2:nrow(dataset)))*

*{*

*if(dataset[i,('Seasons')] == 1)*

*{*

*dataset[i,('Seasons')] = 'summer'*

*}*

*if(dataset[i,('Seasons')] == 2)*

*{*

*dataset[i,('Seasons')] = 'autumn'*

*}*

*if(dataset[i,('Seasons')] == 3)*

*{*

*dataset[i,('Seasons')] = 'winter'*

*}*

*if(dataset[i,('Seasons')] == 4)*

*{*

*dataset[i,('Seasons')] = 'spring'*

*}*

*}*

*# Education (high school (1), graduate (2), postgraduate (3), master and doctor (4))*

*dataset$Education[is.na(dataset$Education)] = -1*

*for (i in c(1,2:nrow(dataset)))*

*{*

*if(dataset[i,('Education')] == 1)*

*{*

*dataset[i,('Education')] = 'high school'*

*}*

*if(dataset[i,('Education')] == 2)*

*{*

*dataset[i,('Education')] = 'graduate'*

*}*

*if(dataset[i,('Education')] == 3)*

*{*

*dataset[i,('Education')] = 'postgraduate'*

*}*

*if(dataset[i,('Education')] == 4)*

*{*

*dataset[i,('Education')] = 'master and doctor'*

*}*

*}*

*View(dataset)*

*#Converting required variables to categorical and integer as explained in the project report*

*for (i in colnames(DataCat))*

*{*

*#print(dataset[,i])*

*dataset[,i] = as.factor(dataset[,i])*

*}*

*for (i in colnames(DataCon))*

*{*

*#print(dataset[,i])*

*dataset[,i] = as.integer(dataset[,i])*

*}*

*str(dataset)*

*#Missing Values*

*#Dividing Categorical variables into two dataframes*

*#Variables like ID,Reason for absence,Day of the week ,Seasons ,Education. Since in this case 0 shows loss of information,So we will replace them with Nan values and then impute them*

*DataCat\_1 = data.frame()*

*DataCat\_1 = subset(DataCat , select = c('ID','Reason.for.absence','Month.of.absence','Day.of.the.week','Seasons','Education'))*

*#DataCat\_1.isnull().sum()*

*#Variables like 'Disciplinary failure','Son', 'Social drinker','Social smoker', 'Pet' include 0 as important information so we are keeping them in other dataset*

*DataCat\_2 = data.frame()*

*DataCat\_2 = subset(DataCat, select = c('Disciplinary.failure', 'Social.drinker','Social.smoker'))*

*#DataCat\_1*

*#replace 0 with NA*

*DataCat\_1[DataCat\_1 == 0] <- NA*

*#creting a new dataframe for DataSet for implementing data preprocessing techniques*

*#Adding firstly continuous data*

*DataSet = data.frame()*

*DataSet = DataCon*

*View(DataSet)*

*#Missing values percentage for continuous data*

*missing\_val = data.frame(apply(DataSet,2,function(x){sum(is.na(x))}))*

*missing\_val$Columns = row.names(missing\_val)*

*names(missing\_val)[1] = "Missing\_percentage"*

*missing\_val$Missing\_percentage = (missing\_val$Missing\_percentage/nrow(dataset)) \* 100*

*row.names(missing\_val) = NULL*

*missing\_val = missing\_val[,c(2,1)]*

*missing\_val*

*#Firstly Imputating missing values for Continuous Variables*

*#Creating Null value (Actual value is 36)*

*DataSet[1,('Distance.from.Residence.to.Work')] = NA*

*DataSet[1,('Distance.from.Residence.to.Work')]*

*#Mean Method (29.659)*

*DataSet$Distance.from.Residence.to.Work[is.na(DataSet$Distance.from.Residence.to.Work)] = mean(DataSet$Distance.from.Residence.to.Work, na.rm = T)*

*DataSet[1,('Distance.from.Residence.to.Work')]*

*#Median Method(26)*

*DataSet$Distance.from.Residence.to.Work[is.na(DataSet$Distance.from.Residence.to.Work)] = median(DataSet$Distance.from.Residence.to.Work, na.rm = T)*

*DataSet[1,('Distance.from.Residence.to.Work')]*

*# kNN Imputation*

*DataSet = knnImputation(DataSet, k = 3)*

*DataSet[1,('Distance.from.Residence.to.Work')]*

*##For Categorical Variables Imputting with Mode*

*for (i in colnames(DataCat\_1))*

*{*

*#print(dataset[,i])*

*DataCat\_1[,i] = as.integer(DataCat\_1[,i])*

*}*

*#R does not have a standard in-built function to calculate mode. So we create a user function to calculate mode of a data set in R.*

*# Create the function.*

*getmode <- function(v) {*

*uniqv <- unique(v)*

*uniqv[which.max(tabulate(match(v, uniqv)))]*

*}*

*# Calculate the mode using the user function.*

*r1 <- getmode(DataCat\_1$Reason.for.absence)*

*r2 <- getmode(DataCat\_1$ID)*

*r3 <- getmode(DataCat\_1$Month.of.absence)*

*r4 <- getmode(DataCat\_1$Day.of.the.week)*

*r5 <- getmode(DataCat\_1$Seasons)*

*r6 <- getmode(DataCat\_1$Education)*

*DataCat\_1$Reason.for.absence[is.na(DataCat\_1$Reason.for.absence)] = r1*

*DataCat\_1$ID[is.na(DataCat\_1$ID)] = r2*

*DataCat\_1$Month.of.absence[is.na(DataCat\_1$Month.of.absence)] = r3*

*DataCat\_1$Day.of.the.week[is.na(DataCat\_1$Day.of.the.week)] = r4*

*DataCat\_1$Seasons[is.na(DataCat\_1$Seasons)] = r5*

*DataCat\_1$Education[is.na(DataCat\_1$Education)] = r6*

*View(DataCat\_1)*

*#For DataCat\_2*

*r7 <- getmode(DataCat\_2$Disciplinary.failure)*

*r8 <- getmode(DataCat\_2$Social.drinker)*

*r9 <- getmode(DataCat\_2$Social.smoker)*

*DataCat\_2$Disciplinary.failure[is.na(DataCat\_2$Disciplinary.failure)] = r7*

*DataCat\_2$Social.drinker[is.na(DataCat\_2$Social.drinker)] = r8*

*DataCat\_2$Social.smoker[is.na(DataCat\_2$Social.smoker)] = r9*

*View(DataCat\_2)*

*#reviewing missing values(If they got imputed correctly or not)*

*sum(is.na(DataCat\_1))*

*sum(is.na(DataCat\_2))*

*#Joining categorical data with main dataset as we have imputed it seperatly*

*df1 = cbind.data.frame(DataSet,DataCat\_1,DataCat\_2)*

*DataSet = df1*

*#DataSet is our final DataSet after imputing all the missing values*

*View(DataSet)*

*#Outlier Analysis*

*# ## BoxPlots - Distribution and Outlier Check*

*cnames = colnames(DataCon)*

*for (i in 1:length(cnames))*

*{*

*assign(paste0("gn",i), ggplot(aes\_string(y = (cnames[i]), x = "Absenteeism.time.in.hours"), data = subset(DataSet))+*

*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="Absenteeism.time.in.hours")+*

*ggtitle(paste("Box plot of responded for",cnames[i])))*

*}*

*# Plotting plots together*

*gridExtra::grid.arrange(gn1,gn5,gn2,ncol=3)*

*gridExtra::grid.arrange(gn6,gn7,ncol=2)*

*gridExtra::grid.arrange(gn8,gn9,ncol=2)*

*# #Remove outliers using boxplot method*

*df = DataSet*

*# # #loop to remove from all variables*

*for(i in cnames){*

*#print(i)*

*val = DataSet[,i][DataSet[,i] %in% boxplot.stats(DataSet[,i])$out]*

*#print(length(val))*

*DataSet = DataSet[which(!DataSet[,i] %in% val),]*

*}*

*cnames = colnames(DataCon)*

*## Correlation Plot*

*corrgram(DataSet[,cnames], order = F,*

*upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")*

*## Dimension Reduction*

*DataSet = subset(DataSet, select = -c(Age,Weight,Body.mass.index))*

*## Chi-square Test for categorical variables*

*#1*

*print(chisq.test(table(DataCat$ID,DataCat$Reason.for.absence)))*

*DataSet$ID = NULL*

*DataCat$ID = NULL*

*#2*

*print(chisq.test(table(DataCat$Month.of.absence,DataCat$Reason.for.absence)))*

*DataSet$Reason.for.absence = NULL*

*DataCat$Reason.for.absence = NULL*

*#3*

*print(chisq.test(table(DataCat$Month.of.absence,DataCat$Day.of.the.week)))*

*#4*

*print(chisq.test(table(DataCat$Seasons,DataCat$Day.of.the.week)))*

*#5*

*print(chisq.test(table(DataCat$Seasons,DataCat$Disciplinary.failure)))*

*DataSet$Disciplinary.failure = NULL*

*DataCat$Disciplinary.failure = NULL*

*#6*

*print(chisq.test(table(DataCat$Education,DataCat$Seasons)))*

*#7*

*print(chisq.test(table(DataCat$Son,DataCat$Education)))*

*DataSet$Son = NULL*

*DataCat$Son = NULL*

*#8*

*print(chisq.test(table(DataCat$Social.drinker,DataCat$Day.of.the.week)))*

*#9*

*print(chisq.test(table(DataCat$Social.drinker,DataCat$Education)))*

*DataSet$Education = NULL*

*DataCat$Education = NULL*

*#10*

*print(chisq.test(table(DataCat$Social.drinker,DataCat$Social.smoker)))*

*DataSet$Social.smoker = NULL*

*DataCat$Social.smoker = NULL*

*#11*

*print(chisq.test(table(DataCat$Pet,DataCat$Social.drinker)))*

*DataSet$Pet = NULL*

*DataCat$Pet = NULL*

*#12*

*print(chisq.test(table(DataCat$Month.of.absence,DataCat$Day.of.the.week)))*

*#13*

*print(chisq.test(table(DataCat$Month.of.absence,DataCat$Seasons)))*

*DataSet$Seasons = NULL*

*DataCat$Seasons = NULL*

*#14*

*print(chisq.test(table(DataCat$Day.of.the.week,DataCat$Social.drinker)))*

*#15*

*print(chisq.test(table(DataCat$Month.of.absence,DataCat$Social.drinker)))*

*DataSet$Month.of.absence = NULL*

*DataCat$Month.of.absence = NULL*

*#Dataset after feature Selection*

*View(DataSet)*

*#Normalisation*

*cnames = c(colnames(DataCon))*

*for(i in cnames){*

*print(i)*

*DataSet[,i] = (DataSet[,i] - min(DataSet[,i]))/*

*(max(DataSet[,i] - min(DataSet[,i])))*

*}*

*#Dataset after feature Scaling*

*View(DataSet)*

*#Divide the data into train and test*

*TrainData\_index = sample(1:nrow(DataSet), 0.8 \* nrow(DataSet))*

*TrainData = DataSet[TrainData\_index,]*

*TestData = DataSet[-TrainData\_index,]*

*#1. Linear Regression Model*

*#check multicollearity*

*#usdm helps us to do use vif function to find out multicollinearity effect*

*library(usdm)*

*vif(DataSet[,-7])*

*vifcor(DataSet[,-7], th = 0.9)*

*#As a result we haven't got min correlation( Day.of.the.week ~ Work.load.Average.day. ): -0.0001513119*

*# max correlation ( Social.drinker ~ Distance.from.Residence.to.Work ): 0.448258*

*#Which are far from our threshold value 0.9 , So we do not need to remove any of the variables from our current DataSet*

*#run regression model*

*lm\_model = lm(Absenteeism.time.in.hours ~., data = TrainData)*

*#Summary of the model*

*summary(lm\_model)*

*#Predict*

*predictions\_LR = predict(lm\_model, TestData[,-7])*

*#Calculate MAE*

*MAE( predictions\_LR,TestData[,7])*

*#Calculate RMSE*

*RMSE(predictions\_LR,TestData[,7])*

*#2. Decision Tree Regression Model*

*#Load Libraries*

*library(rpart)*

*#rpart for regression*

*fit2 = rpart(Absenteeism.time.in.hours ~ ., data = TrainData, method = "anova")*

*#Predict for new test cases*

*predictions\_DT = predict(fit2, TestData[,-7])*

*#calculate MAE*

*MAE(predictions\_DT,TestData[,7])*

*#Calculate RMSE*

*RMSE(predictions\_DT,TestData[,7])*

*#3. Random Forest Regression Model*

*fit3 = randomForest(formula = Absenteeism.time.in.hours ~ ., data = TrainData, ntree = 45, method = "anova")*

*#Predict for new test cases*

*predictions\_RF = predict(fit3, TestData[,-7])*

*#calculate MAE*

*MAE(predictions\_RF,TestData[,7])*

*#Calculate RMSE*

*RMSE(predictions\_RF,TestData[,7])*

*#Applying Random Forest on entire DataSet*

*fit\_N = randomForest(formula = Absenteeism.time.in.hours ~ ., data = DataSet, ntree = 45 , method = "anova")*

*#Predict test data using random forest model*

*predictions\_RF = predict(fit\_N, DataSet[,-7])*

*#Write output results*

*DataTestFinal = read.xlsx("Absenteeism\_at\_work\_Project.xls",sheetIndex =1,header = T)*

*df\_Output = data.frame(DataTestFinal)*

*df\_Output['Predicted\_Absenteeism'] = predictions\_RF*

*write.csv(df\_Output, "R\_Absenteeism\_Output.csv",row.names = F)*