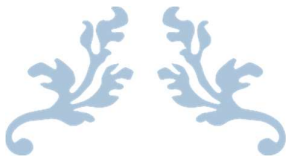


EMPLOYEE ABSENTEEISM PREDICTION

JANUARY 15, 2019

AKASH HEMANT SAXENA

Contents

1. Introduction

1.1. Problem Statement
1.2. Data

2. Methodology

2.1. Pre Processing
    2.1.1. Missing Value Analysis
    2.1.2. Feature Selection
    2.1.3. Feature Scaling
2.2. Modelling
    2.2.1. Model Selection
    2.2.2. k-Nearest Neighbours
    2.2.3. Decision Trees
    2.2.4. Random Forest
    2.2.5 Linear Regression

3. Conclusion

3.1. Model Evaluation
3.2 Model Selection
3.3 Summary

Appendix A – Python Code

Appendix B – R Code

References

Chapter 1

Introduction

1.1 Problem Statement

Employee absenteeism is a major problem faced by every employer which eventually lead to the backlogs, piling of the work, delay in deploying the project and can have a major effect on company finances. The aim of this project is to find an issue which eventually leads toward the absence of an employee and provide a proper solution to reduce the absenteeism in the company.

1.2 Data

The task is to build a regression model which will predict the number of absent hours in relation to the employee depend upon the various factor.

Given below is the sample of the data set which is used to predict the number of absent hours related to the employee.

Table 1.1: Employee Absenteeism Sample Data (Columns: 1-8)

ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time
11	26	7	3	1	289	36	13
36	0	7	3	1	118	13	18
3	23	7	4	1	179	51	18
7	7	7	5	1	279	5	14
11	23	7	5	1	289	36	13
3	23	7	6	1	179	51	18
10	22	7	6	1		52	3
20	23	7	6	1	260	50	11
14	19	7	2	1	155	12	14

Table 1.2: Employee Absenteeism Sample Data (Columns:9-18)

Age	Workload Average/day	Hit target	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Weight
33	2,39,554	97	0	1	2	1	0	1	90
50	2,39,554	97	1	1	1	1	0	0	98
38	2,39,554	97	0	1	0	1	0	0	89
39	2,39,554	97	0	1	2	1	1	0	68
33	2,39,554	97	0	1	2	1	0	1	90
38	2,39,554	97	0	1	0	1	0	0	89
28	2,39,554	97	0	1	1	1	0	4	80
36	2,39,554	97	0	1	4	1	0	0	65
34	2,39,554	97	0	1	2	1	0	0	95

Table 1.3: Employee Absenteeism Sample Data (Columns: 19-21)

Height	Body mass index	Absenteeism time in hours
172	30	4
178	31	0
170	31	2
168	24	4
172	30	2
170	31	
172	27	8
168	23	4
196	25	40

As per the above table following are the variable using which the regression model has to predict the number of hours the employee going to absent.

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

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

1. CERTAIN INFECTIOUS AND PARASITIC DISEASES
2. NEOPLASMS
3. DISEASES OF THE BLOOD AND BLOOD-FORMING ORGANS AND CERTAIN DISORDERS INVOLVING THE IMMUNE MECHANISM
4. ENDOCRINE NUTRITIONAL AND METABOLIC DISEASES
5. MENTAL AND BEHAVIOURAL DISORDERS
6. DISEASES OF THE NERVOUS SYSTEM
7. DISEASES OF THE EYE AND ADNEXA
8. DISEASES OF THE EAR AND MASTOID PROCESS
9. DISEASES OF THE CIRCULATORY SYSTEM
10. DISEASES OF THE RESPIRATORY SYSTEM
11. DISEASES OF THE DIGESTIVE SYSTEM
12. DISEASES OF THE SKIN AND SUBCUTANEOUS TISSUE
13. DISEASES OF THE MUSCULOSKELETAL SYSTEM AND CONNECTIVE TISSUE
14. DISEASES OF THE GENITOURINARY SYSTEM
15. PREGNANCY, CHILDBIRTH AND THE PUERPERIUM
16. CERTAIN CONDITIONS ORIGINATING IN THE PERINATAL PERIOD
17. CONGENITAL MALFORMATIONS, DEFORMATIONS AND CHROMOSOMAL ABNORMALITIES
18. SYMPTOMS, SIGNS AND ABNORMAL CLINICAL AND LABORATORY FINDINGS, NOT ELSEWHERE CLASSIFIED
19. INJURY, POISONING AND CERTAIN OTHER CONSEQUENCES OF EXTERNAL CAUSES
20. EXTERNAL CAUSES OF MORBIDITY AND MORTALITY
21. FACTORS INFLUENCING HEALTH STATUS AND CONTACT WITH HEALTH SERVICES.
22. PATIENT FOLLOW-UP
23. MEDICAL CONSULTATION
24. BLOOD DONATION
25. LABORATORY EXAMINATION
26. UNJUSTIFIED ABSENCE
27. PHYSIOTHERAPY
28 DENTAL CONSULTATION.

3. Month of absence
4. Day of the week (2. MONDAY 3. TUESDAY 4. WEDNESDAY 5. THURSDAY 6. FRIDAY)
5. Seasons (1. SUMMER 2. AUTUMN 3. WINTER 4. SPRING)
6. Transportation expense
7. Distance from Residence to Work (kilometres)
8. Service time
9. Age
10. Workload Average/day
11. Hit target
12. Disciplinary failure (yes=1; no=0)
13. Education (HIGH SCHOOL (1), GRADUATE (2), POSTGRADUATE (3), MASTER AND DOCTOR (4))
14. Son (number of children)
15. Social drinker (yes=1; no=0)
16. Social smoker (yes=1; no=0)
17. Pet (number of pet)
18. Weight
19. Height
20. Body mass index
21. Absenteeism time in hours

Chapter 2

Methodology

2.1 Pre-processing

The output of the Machine Learning data is fully depended upon the data feed in the algorithm, ML algorithm does not generate the proper outcome on raw data. So it's important to transform the data so that the model accuracy of the model can be increased. In other words, we must apply some cleaning and processing for a better outcome with respect to the Machine learning algorithm. Some Machine learning algorithm required well-processed data like Decision Tree and random forest does not take null value. So it is important to process the data before feeding it in the respective machine learning algorithm.

Following are the pre-processing operation performed on the data to reduce the error rate and produce the optimal output.

2.1.1 Missing Value Analysis

There are several options to handle missing value, but it mainly depends upon the nature of the data set, which missing analytics produce the optimum solution. Missing of data can occur due to nonresponse occur for example privacy concern. Moreover, if the value is missing completely at random, the data sample still represents the population but if the value is missing in some pattern, analysis produces bias output. There is two form of random missing values: MCAR and MAR. MCAR exists when the missing values are randomly distributed across all observations. This form can be confirmed by partitioning the data into two parts: one set containing the missing values, and the other containing the non-missing values. The second form is missing at random (MAR). In MAR, the missing values are not randomly distributed across observations but are distributed within one or more sub-samples. Dropping the missing value is the good option if the data set has a low percentage of missing value and remaining data set can be used for further processing but it has its cons, dropping the missing value observation reduce the quantity of data. If our data set has a large number of missing value with respect to observation or feature, dropping is a good option. As fig 2.1 shows all the feature in the dataset is below 5% of missing value. Moreover, 13.65% of the data set has a missing value, so replacing the missing value with the statically analysis or clustering is a better solution. Multiple missing value approach can be implemented to replace the missing value with the predictive value generated by the approach. There are total 740 observations and 21 feature and by dropping all the missing value from the dataset to dimension reduce to 639 observations. This substantial reduction in the dimension can reduce to the accuracy of the machine learning algorithm.

Table 2.1: Missing Value Table

Variables	Missing_Val	Missing_per
Body mass index	31	4.19%
Absenteeism time in hours	22	2.97%
Height	14	1.89%
Work load Average/day	10	1.35%
Education	10	1.35%
Transportation expense	7	0.95%
Son	6	0.81%
Disciplinary failure	6	0.81%
Hit target	6	0.81%
Social smoker	4	0.54%
Age	3	0.41%
Reason for absence	3	0.41%
Service time	3	0.41%
Distance from Residence to Work	3	0.41%
Social drinker	3	0.41%
Pet	2	0.27%
Weight	1	0.14%
Month of absence	1	0.14%
Seasons	0	0%
Day of the week	0	0%
ID	0	0%

Table 2.2 represent all the approach parameter applied to check which missing value analysis produce the lowest variance with respect to the actual data and predicted data.

The first column represents the feature on which the missing value analysis applied. The second column represents the actual value in the cell. Sample array is randomly generated by the random generator and the value is replaced with NAN after fetching the actual value.

The third column shows the value when mean imputation is applied to the data. In mean imputation technique, the missing data is replaced with the mean of the respective feature. This technique produces a better result for a continuous variable. Mean imputation method does not produce the optimal output.

The fourth column represents the value with respect to the feature when the median imputation is applied to the data. Median imputation is similar to the mean imputation but rather than replacing the NAN value with the mean of the feature, this imputation method replaces it with the median of the feature. Similar to the mean imputation, median imputation should be applied on continues variable rather than applying on a categorical variable.

In the respective table, the mean and median column has NAN value because some feature in the data is of categorical nature.

The last third column represents the KNN imputation method. KNN imputation method checks all the nearest variable with respect to the missing value. KNN algorithm used for the matching point with the closest k neighbour in the multi-dimension. It can be used for data that are categorical, discrete and continuous which makes it more efficient with all kind of missing data. In the last three columns (KNN_n) where n represents the kth unit.

K is the number of neighbour voting on the for the missing value and that the main reason the kth value is an odd number because even kth value will generate conflict in the voting process. KNN_3 represent the value which was generated when applying KNN where k value = 3. Same for the KNN_5 and KNN_7 where k value was 5 and 7 respectively.

Table 2.2: Multiple Missing Value Analysis Test Table

```
SAMPLE TESTED ARRAY IS [597, 621, 699, 149, 25, 460, 708, 611, 563, 440, 377, 474, 454, 726, 413, 205, 419, 583]
```

Out[33]:

	Actual Value	Mean	Median	KNN_3	KNN_5	KNN_7
Reason for absence	27	NaN	NaN	27	27	27
Month of absence	3	NaN	NaN	3	3	3
Transportation expense	291	221.035	221.035	291	291	291
Distance from Residence to Work	42	29.6676	29.6676	39	39	39
Service time	3	12.5658	12.5658	3	3	3
Age	30	36.4491	36.4491	28	28	28
Work load Average/day	275089	271189	271189	245678	245678	245678
Hit target	97	94.5872	94.5872	96	96	96
Disciplinary failure	0	NaN	NaN	0	0	0
Education	3	NaN	NaN	2	2	2
Son	0	1.01771	1.01771	0	0	0
Social drinker	1	NaN	NaN	0	0	0
Social smoker	0	NaN	NaN	0	0	0
Pet	8	0.746612	0.746612	1	1	1
Weight	65	79.0636	79.0636	65	65	65
Height	196	172.153	172.153	195	195	195
Body mass index	31	26.6841	26.6841	28	28	28
Absenteeism time in hours	3	6.97772	6.97772	2	2	2

As per data represent in table 2.2, KNN imputation provides the optimum solution for absenteeism data with minimum variation with respect to actual data.

2.1.2 Feature Selection

Feature selection is another pre-processing technique which decreases the load over machine learning algorithm checking the correlation between other feature and check which feature is highly correlated to another feature. If two feature carries the same data, this will create a bias output. Feature selection is a process where you can select those features which contribute most to the prediction output. Moreover, having a relevant feature in the data set can decrease the accuracy of the model.

Fig 2.3 represents the correlation heatmap with respect to the data. Darker colour represents the positive correlation and lighter colour scheme represent the negative correlation with respect to other features. Body mass index is highly positively related towards weight feature in the dataset which is 0.9. Moreover, no other feature is correlated with respect to other feature. Moreover, body mass index is the ratio of weight with respect to the square of height, so by dropping the two feature from the data set will reduce the time complexity with the same efficiency.

Fig 2.3: Correlation Heat map

2.1.3 Feature Scaling

Feature scaling is a method to standardize the variable or the feature of the data set, It also knows as standardization of a data set. Some feature has magnitudes, unit or range and normalization should be performed when the scaling of the feature is irrelevant or have a high magnitude which will lead to the bases in the machine learning algorithm which will lean towards the higher magnitude dataset. Z-scoring/standardization is one technique in which the value of a feature is subtracted with the mean of the feature and their subtraction is divided by the standard deviation of the feature which will also set the feature in normalized bell shape form whereas another method involves the value of a feature is subtracted by the minimum value of the respective feature which is divided by the subtraction of the minimum value and maximum value of the respective feature. In our dataset, some feature has irrelevant scaling which will affect the machine learning algorithm. A feature like 'Transportation expense', 'Distance from Residence to Work','Service time','Workload Average/day ', 'Hit target','Height' and 'Weight' is scaled by applying the min-max scaling technique.

Min-Max scaling:

$$Yi = [Xi - min(X)]/[max(X) - min(X)]$$

2.2 Modelling

2.1.1 Model Selection

In our earlier stage of analysis, we came to know that our dependent variable in our case is *Absenteeism time in hours* is numerical, so we are using a **Regression analysis** over our data set.

As we observed in missing value analysis KNN imputation generated the best result with respect our data set. So we are going to apply K-Nearest Neighbours regression first and will check all the error metrics and after Decision tree and random forest machine learning algorithm will be applied and evaluate which machine learning algorithm performs best and produce best predict for our data set.

2.1.2 K-Nearest Neighbours

K-nearest Neighbour predicts the value by checking the distance with from other feature with respect to the K^{th} value.

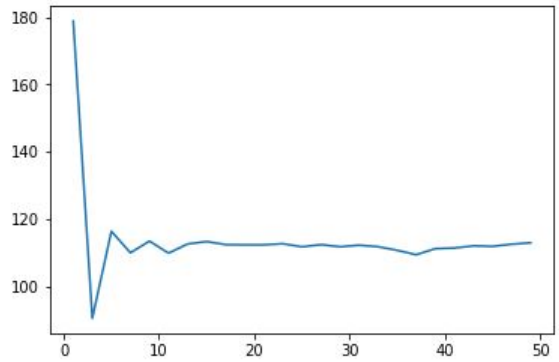
Applying the K-nearest Neighbours on our data set with k =3. Checking which k^{th} value fit best for the respective data set.

Python:

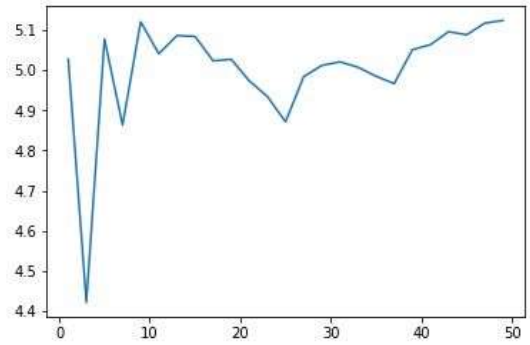
```
from sklearn.neighbors import KNeighborsRegressor
Pro_KNN_model = KNeighborsRegressor(n_neighbors=3).fit(X_TRAIN,Y_TRAIN)
Final_KNN_predict = Pro_KNN_model.predict(X_TEST)
reg_acc(Y_TEST,Final_KNN_predict)
```

Following graph shows which kth value fits the data set and provide the optimal outcome.

```
********** M S E **********
Kth Value: 3
Lowest MSE Value: 90.43993993993993
```



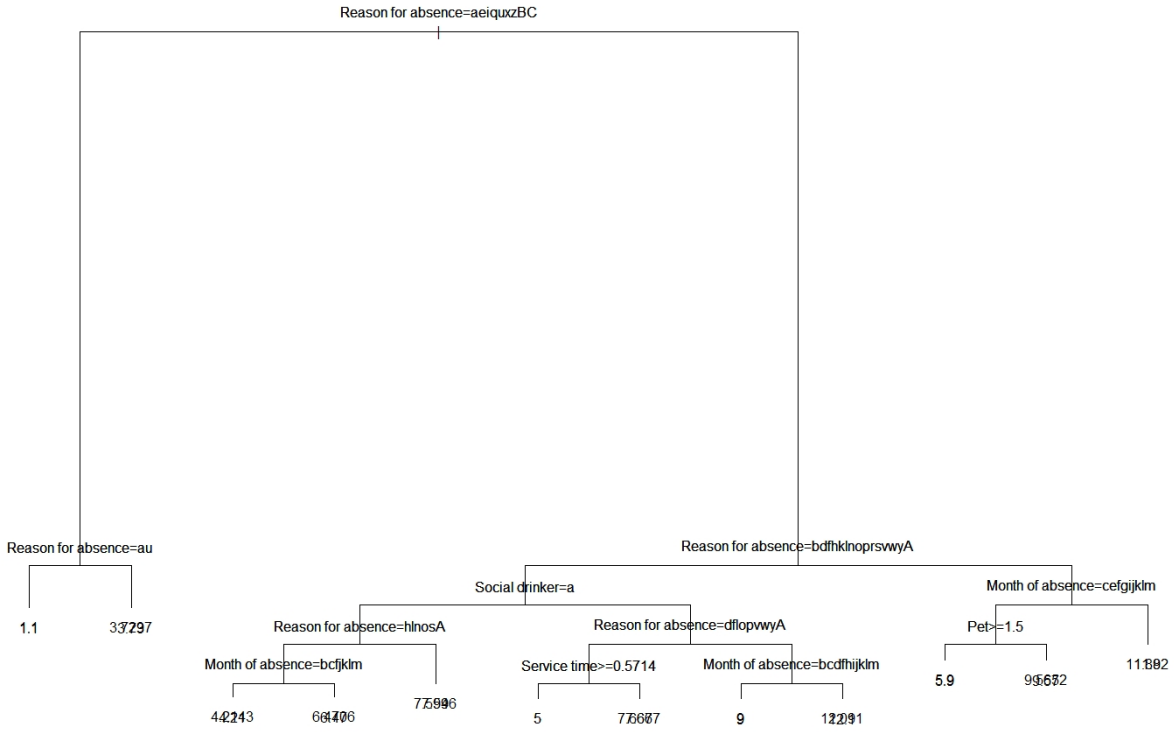

```
********** M A E **********
Kth Value: 3
Lowest MAE Value: 4.421921921921922
```



As seen in the above fig the k=3 is the optimum value for the K-nearest Neighbour machine learning algorithm to predict the value with the lowest error rate.

2.2.3. Decision Trees

Decision Tree algorithm creates a flow chat like tree structure where end nodes denote the outcome and the sub-nodes denote the decision flow of the tree. Following are the code and the decision tree with respect to the data set.



Python:

```
from sklearn import tree
Deci_tree_pre =
tree.DecisionTreeRegressor().fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
print('Decision Tree ML')
reg_acc(Deci_tree_pre,Y_TEST)
```

2.2.4. Random Forest

Similar to the decision tree, we also applied a random forest algorithm to check the prediction rate of a data set. Random forest works the same as decision tree but rather than creating a single tree, random forest creates multiple trees corresponding to the data set.

Python:

```
from sklearn.ensemble import RandomForestRegressor
Ran_for_pre =
RandomForestRegressor(n_estimators=50).fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
print('Random Forest ML')
reg_acc(Ran_for_pre,Y_TEST)
```

2.2.5 Linear Regression

As the dependent variable is a numerical data, we are using linear regression model to *'Absenteeism time in hours'*. Following are the coefficient metrics with respect to all the feature of the data set to predict the value of Y (*Absenteeism time in hours'*).

```
Call:
lm(formula = `Absenteeism time in hours` ~ ., data = train)

Residuals:
    Min       1Q       Median        3Q         Max
-6.3458  -1.1343      -0.0563    0.8606     15.6340

Coefficients: (1 not defined because of singularities)
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                     2.79226    1.95670   1.427 0.154088
ID                               -0.05947    0.01447  -4.110 4.51e-05 ***
`Reason for absence`1             6.52595    0.80135   8.144 2.20e-15 ***
`Reason for absence`2             9.46634    2.54393   3.721 0.000217 ***
`Reason for absence`3             6.22549    2.53192   2.459 0.014219 *
`Reason for absence`4             5.40690    1.85154   2.920 0.003628 **
`Reason for absence`5             6.13844    1.50245   4.086 4.99e-05 ***
`Reason for absence`6             7.79326    1.04993   7.423 3.91e-13 ***
`Reason for absence`7             5.81834    0.81231   7.163 2.31e-12 ***
`Reason for absence`8             5.36061    1.19432   4.488 8.59e-06 ***
`Reason for absence`9             8.55349    1.51529   5.645 2.54e-08 ***
`Reason for absence`10            6.34896    0.69213   9.173  < 2e-16 ***
`Reason for absence`11            4.71419    0.68135   6.919 1.16e-11 ***
`Reason for absence`12            7.22463    0.99589   7.254 1.24e-12 ***
`Reason for absence`13            6.57174    0.55826  11.772  < 2e-16 ***
`Reason for absence`14            5.47856    0.73605   7.443 3.39e-13 ***
`Reason for absence`15            5.69714    1.80309   3.160 0.001658 **
`Reason for absence`16            2.08292    1.53686   1.355 0.175824
`Reason for absence`17            6.45778    2.56007   2.523 0.011908 *
`Reason for absence`18            6.12417    0.72469   8.451  < 2e-16 ***
`Reason for absence`19            7.44727    0.59954  12.422  < 2e-16 ***
`Reason for absence`20            1.25915    2.54105   0.496 0.620409
`Reason for absence`21            5.27641    1.05271   5.012 7.08e-07 ***
`Reason for absence`22            6.02079    0.62831   9.582  < 2e-16 ***
`Reason for absence`23            2.65956    0.48355   5.500 5.61e-08 ***
`Reason for absence`24            6.26887    1.48822   4.212 2.91e-05 ***
`Reason for absence`25            3.27262    0.66702   4.906 1.19e-06 ***
`Reason for absence`26            5.45160    0.62949   8.660  < 2e-16 ***
`Reason for absence`27            2.60999    0.61164   4.267 2.30e-05 ***
`Reason for absence`28            2.38481    0.50809   4.694 3.32e-06 ***
`Month of absence`1              -1.35907    1.61833  -0.840 0.401356
`Month of absence`2              -1.03447    1.58414  -0.653 0.513996
`Month of absence`3              -0.75954    1.55412  -0.489 0.625213
`Month of absence`4              -1.05852    1.58550  -0.668 0.504627
`Month of absence`5              -2.08581    1.58425  -1.317 0.188473
`Month of absence`6              -0.47893    1.55922  -0.307 0.758830
`Month of absence`7               0.50256    1.61416   0.311 0.755647
`Month of absence`8               0.36962    1.63771   0.226 0.821516
`Month of absence`9               0.21858    1.65993   0.132 0.895279
`Month of absence`10             -0.32747    1.69494  -0.193 0.846862
`Month of absence`11             -0.13331    1.67565  -0.080 0.936614
`Month of absence`12             -0.30848    1.64247  -0.188 0.851084
`Day of the week`3               -0.14746    0.30291  -0.487 0.626570
`Day of the week`4               -0.16181    0.30396  -0.532 0.594674
`Day of the week`5               -0.75224    0.31758  -2.369 0.018166 *
`Day of the week`6               -0.49362    0.31546  -1.565 0.118162
Seasons2                          0.99788    0.74103   1.347 0.178611
```

```
Seasons3                           1.57433    0.66060   2.383 0.017471 *
Seasons4                           0.50369    0.64333   0.783 0.433963
`Transportation expense`           0.89124    0.54945   1.622 0.105312
`Distance from Residence to Work` -1.40046    0.44689  -3.134 0.001809 **
`Service time`                    -1.24143    1.21753  -1.020 0.308314
Age                                0.03273    0.02497   1.311 0.190367
`Work load Average/day`           -0.01704    0.54236  -0.031 0.974943
`Hit target`                       0.50158    0.84662   0.592 0.553772
`Disciplinary failure`1                 NA         NA      NA       NA
Education2                        -1.26460    0.49164  -2.572 0.010343 *
Education3                        -1.66579    0.45494  -3.662 0.000273 ***
Education4                         0.13991    1.31448   0.106 0.915273
Son                                0.15029    0.10831   1.388 0.165783
`Social drinker`1                  0.03635    0.30216   0.120 0.904281
`Social smoker`1                   0.28253    0.45714   0.618 0.536779
Pet                               -0.29831    0.10498  -2.842 0.004638 **
`Body mass index`                 -1.89423    0.67886  -2.790 0.005432 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.43 on 605 degrees of freedom
Multiple R-squared:  0.518,   Adjusted R-squared:  0.4686
F-statistic: 10.49 on 62 and 605 DF, p-value: < 2.2e-16
```

As seen in the linear regression model, the adjusted R-squared value, we can only explain 47% of the data using the linear regression model.

Python:

```
from sklearn.linear_model import LinearRegression
linReg = LinearRegression().fit(X_TRAIN,Y_TRAIN)
linear_predict = linReg.predict(X_TEST)
reg_acc(Y_TEST,linear_predict)
print('Intercept :',linReg.intercept_)
print('Coefficient')
for i,coef in zip(process_data.columns,linReg.coef_):
    print (i,"-->",coef)
```

3. Conclusion

3.1. Model Evaluation

As this is a regression model prediction, we are using MSE/RMSE and MAE to predict the accuracy and how well the value is predicted with respect to the machine learning algorithm.

Python Model Evaluation:

As the fig 3.1 show, python machine learning algorithm predicted the accuracy parameter.

MSE/RMSE parameter

MSE/RMSE of KNN : 130.57/11.42
MSE/RMSE of Decision Tree : 337.55/18.37
MSE/RMSE of Random Forest : 185.97/13.63
MSE/RMSE of Linear Regression : 136.75/11.69

MAE parameter

MAE of KNN : 4.94
MAE of Decision Tree : 6.73
MAE of Random Forest : 5.27
MAE of Linear Regression : 5.66

With respect to the above parameters, KNN and Random forest provided a better accurate result in comparison with the decision tree.

Fig 3.1: Error Metrics in python

```
In [65]: from sklearn.neighbors import KNeighborsRegressor
         Pro_KNN_model = KNeighborsRegressor(n_neighbors=3).fit(X_TRAIN,Y_TRAIN)
         reg_acc(Final_KNN_predict,Y_TEST)
         Final_KNN_predict = Pro_KNN_model.predict(X_TEST)

         MAE of data:  4.941441441441441
         MSE of data:  130.56456456456456
         R² :  -2.3510975476385143

In [64]: from sklearn import tree
         Deci_tree_pre = tree.DecisionTreeRegressor().fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
         reg_acc(Deci_tree_pre,Y_TEST)

         MAE of data:  6.73536036036036
         MSE of data:  337.55536786786786
         R² :  -0.5632435456267293

In [72]: from sklearn.ensemble import RandomForestRegressor
         Ran_for_pre = RandomForestRegressor(n_estimators=100).fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
         reg_acc(Ran_for_pre,Y_TEST)

         MAE of data:  5.267627654440154
         MSE of data:  185.9669215447047
         R² :  -2.3303200124516454

In [68]: from sklearn.naive_bayes import GaussianNB
         NB_pre = GaussianNB().fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
         reg_acc(NB_pre,Y_TEST)

         MAE of data:  5.101351351351352
         MSE of data:  185.15540540540542
         R² :  -24.29165913130242
```

R Model Evaluation:

As the fig 3.2 show, R machine learning algorithm predicted the accuracy parameter.

MSE/RMSE parameter

MSE/RMSE of KNN : 6.53/2.55

MSE/RMSE of Decision Tree : 8.54/2.92

MSE/RMSE of Random Forest : 4.21/2.05

MSE/RMSE of Linear Regression : 6.09/2.46

MAE parameter

MAE of KNN : 1.76

MAE of Decision Tree : 1.93

MAE of Random Forest : 1.49

MAE of Linear Regression : 1.67

With respect to the above parameters, KNN and Random forest provided better accurate result in comparison with the decision tree.

Fig 3.2: Error metric in R

```
> print('k-Nearest Neighbours')
[1] "k-Nearest Neighbours"
> measureMAE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_knn))
[1] 1.764286
> measureMSE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_knn))
[1] 6.536268
> measureRMSE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_knn))
[1] 2.556613
>
> print('Decision Trees')
> measureMAE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_Dec))
[1] 1.938776
> measureMSE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_Dec))
[1] 8.541694
> measureRMSE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_Dec))
[1] 2.922618
>
> print('Random Forest')
[1] "Random Forest"
> measureMAE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_rf))
[1] 1.496755
> measureMSE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_rf))
[1] 4.215732
> measureRMSE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_rf))
[1] 2.053225
> print('Linear Regession')
[1] "Linear Regession"
> measureMAE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_linreg))
[1] 1.679915
> measureMSE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_linreg))
[1] 6.093979
> measureRMSE(as.numeric(test$`Absenteeism time in hours`),as.numeric(predict_linreg))
[1] 2.468599
>
```

3.2 Model Selection

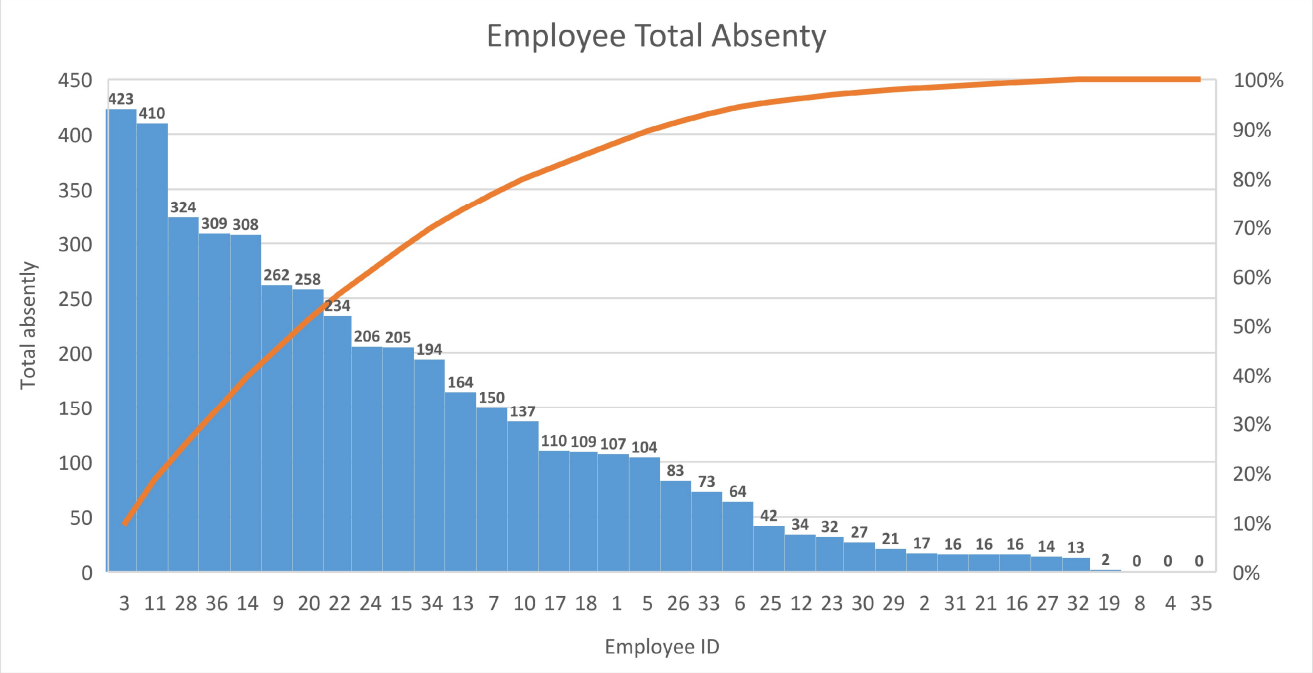
KNN machine learning algorithm produces the best outcome with respect to the data set and the prediction of the '*Absenteeism at work*' was predicted quite accurately. Moreover, the other algorithm also has a reasonable accuracy in predicting the '*Absenteeism at work*'.

3.3 Summary

With respect to the absenteeism data set, we have the following conclusion regarding the company absenteeism and the employees with respect to the employees.

The fig 3.3.1, show the employee ID 3 and 11 has the highest of 423 and 410 respectively. Moreover, these employees have a serious health condition which is leading towards the highest absenteeism.

Fig 3.3.1: Absenteeism with respect to Employee's



The fig 3.3.2 represents, the employees with the age group of 28,33 and 38 has the highest absenteeism in the company. Employee ID 11 has an age of 33, who has serious health issue.

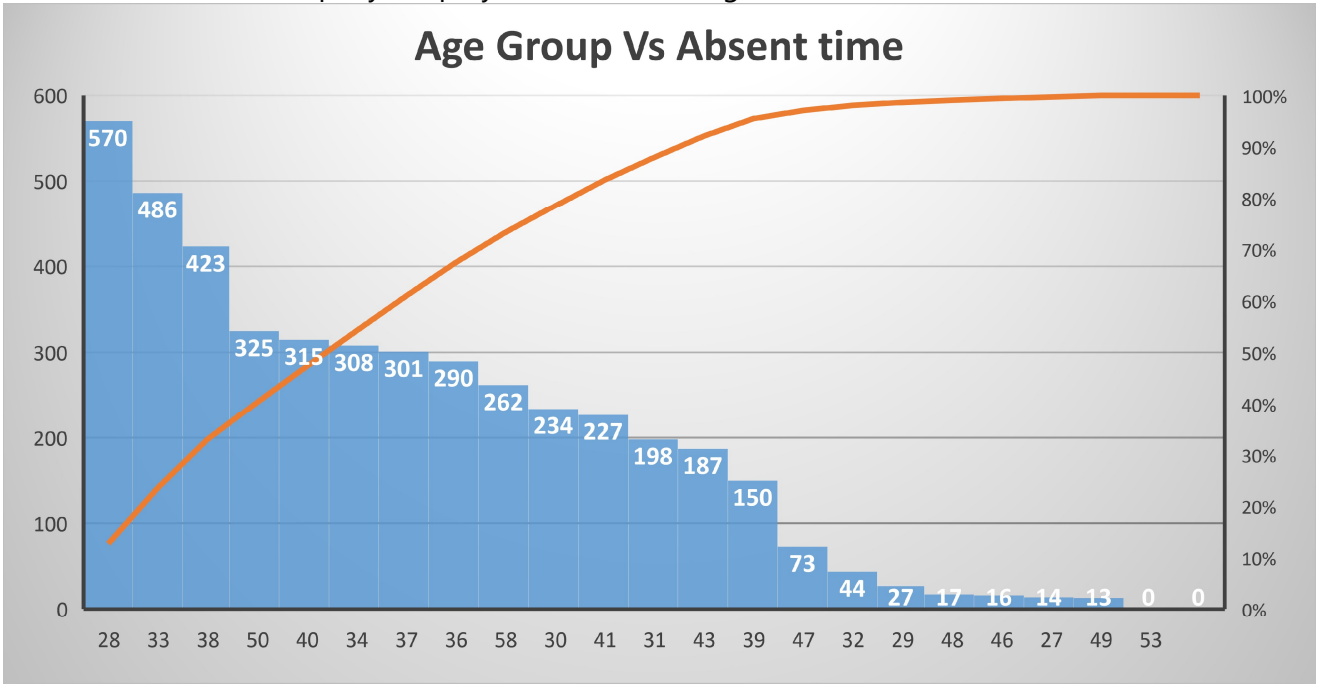
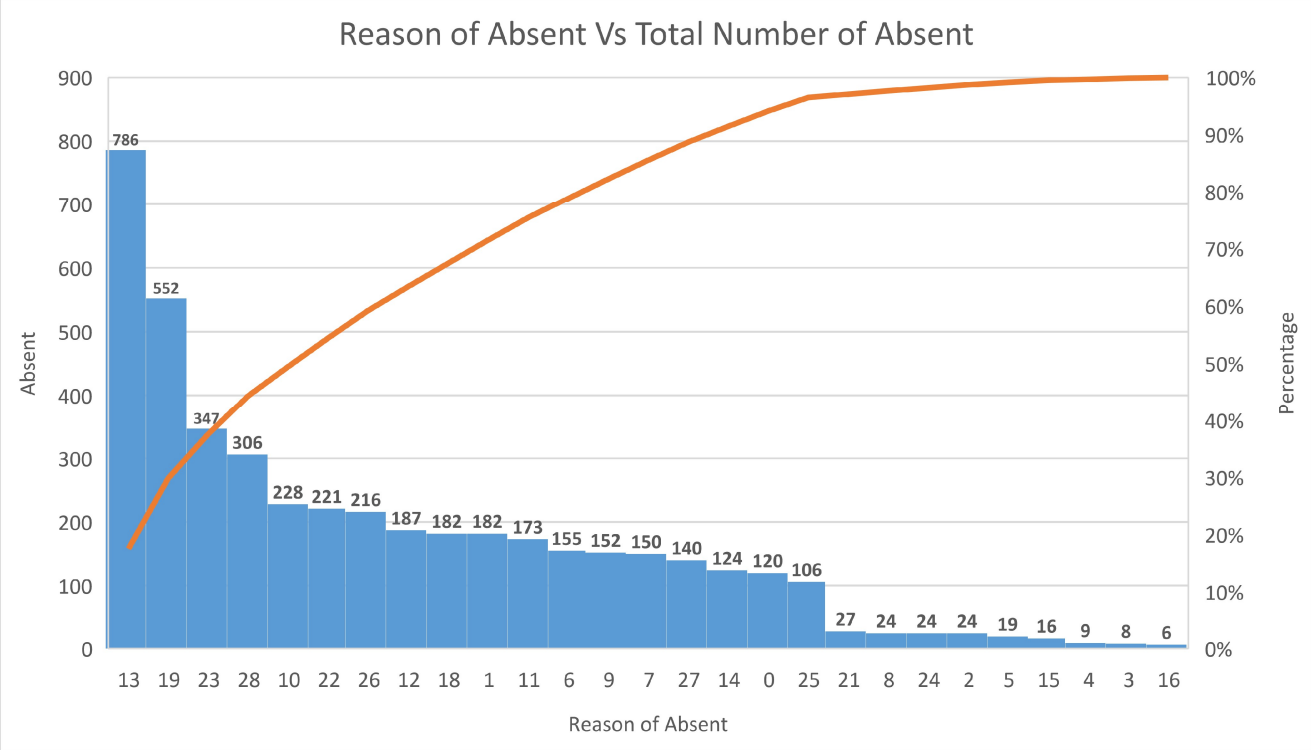


Fig 3.3.2: Age with respect to Absenteeism

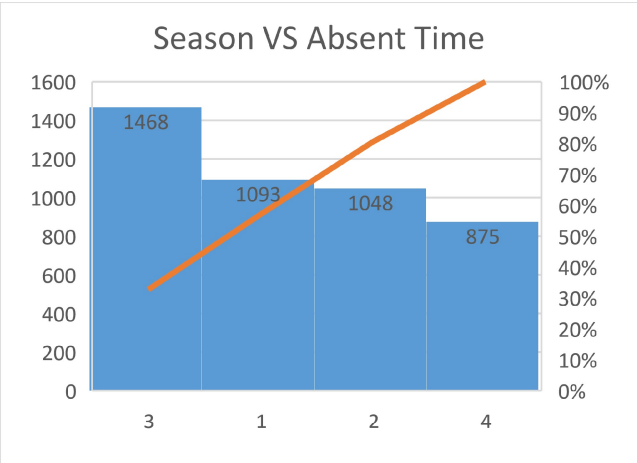
Around half of the employee of the organisation is suffering from "*diseases of the musculoskeletal system and connective tissue*" (Reason of Absent: 13), which leads towards the highest absenteeism rate. Moreover, should also look after the employees how has "*injury, poisoning and certain other consequences of external causes*" (Reason of Absent 19) which is affecting 16 employees of the organisation.

Fig 3.3.3: Reason of Absent with respect to Absenteeism



Employees tend to take more absent leaves in the third season, there might be some employees has a season health issue which is affecting the employees. Moreover, as per the data set employees are taking more absently leave in the month of March and July.

Fig 3.3.4: Seasonal and monthly representation with respect to absenteeism



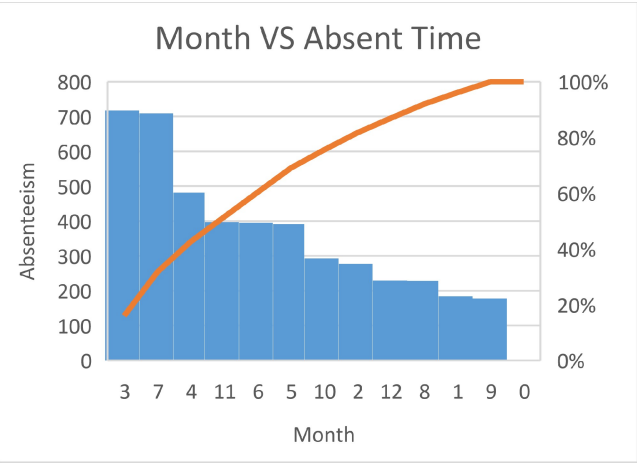
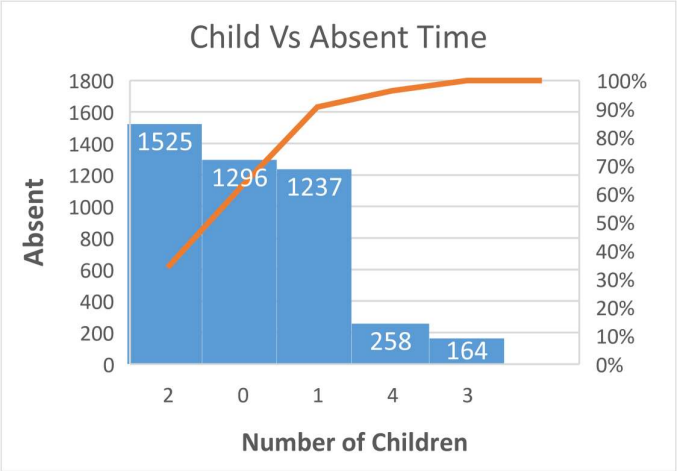


Fig 3.3.5: Absenteeism with respect to Children



The company should emphasis on the employees who has 1 or 2 children, there might be a chance that the increase in absent is due to the employee children.

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

Some employees on the company have a series health condition which should be taken care off. Company management should focus on the employee with no child, as there might be chance of high health risk. Moreover, employee ID 3 and 11 should be taken into consideration as these employees have taken the maximum absentee and also having some serious health issue which leads to the high absenteeism.

"*Diseases of the musculoskeletal system and connective tissue*" and "*Injury, poisoning and certain other consequences of external causes*" are not a common disease, there might be an environmental issue. There might a chance that due to some environmental factor, maximum employees are affecting due to this disease. These are the serious issue which should be taken into consideration as this can also affect other employees of an organisation. Around 60% of the workforce is affected by this disease.

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

Fig 3.3.6 is projecting the forecasting with respect to the season, if this trend of absenteeism continued, there is a chance that the company will suffer the average of 6 (216 hours per season) per season. Fig 3.3.7 also shows the monthly forecasting with 90% confidential interval.

Fig 3.3.6: Seasonal Absenteeism Regression

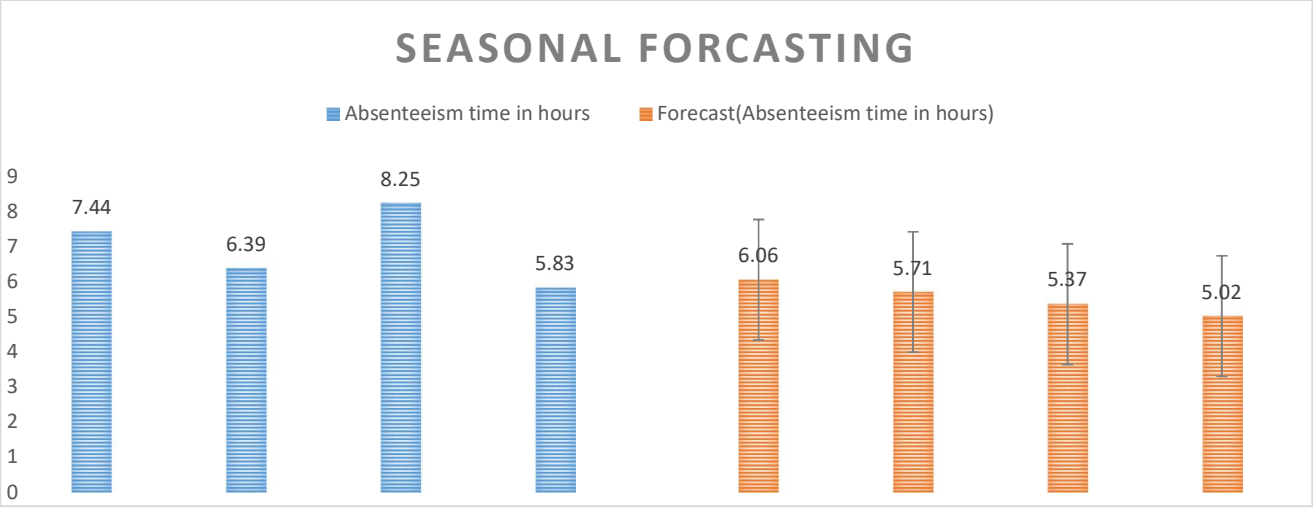
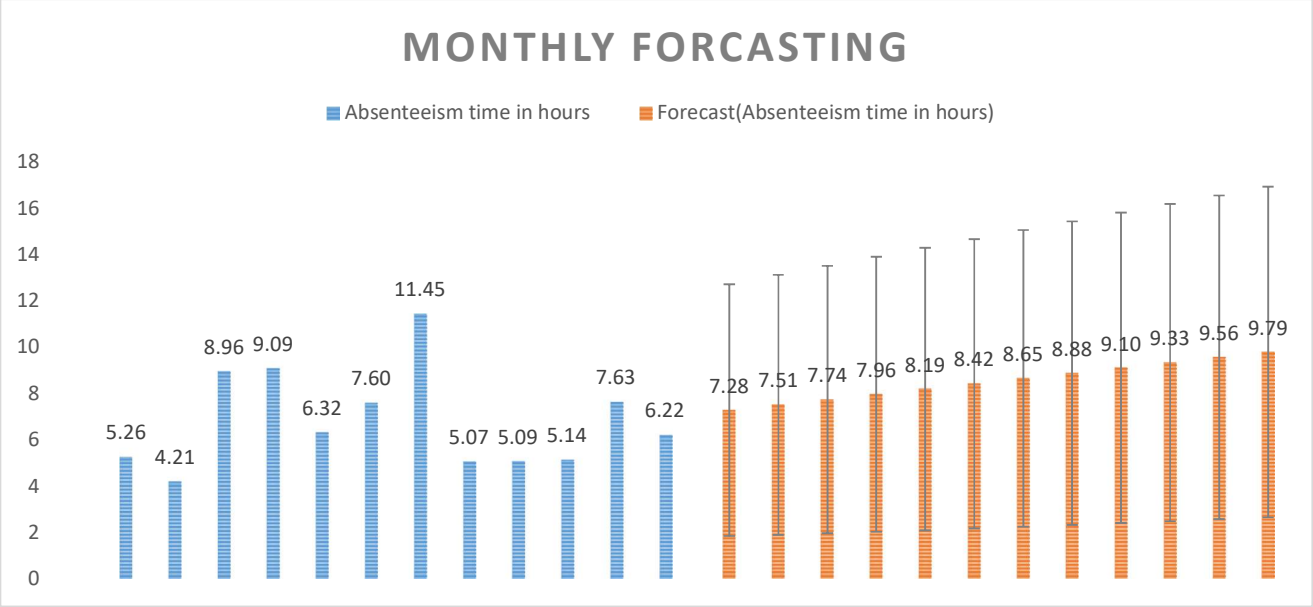


Fig 3.3.7: Monthly Absenteeism Regression



Appendix A – Python Code

```
# IMPORTING ALL THE LIBRARY
import os
import random
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
from fancyimpute import KNN, NuclearNormMinimization, SoftImpute, BiScaler
```

```
#Importing the data
os.chdir('D:/Data Science/EDWISOR/2_PORTFOLIO/project 1')
data = pd.read_excel('Absenteeism_at_work_Project.xls', sheet_name=
'Absenteeism_at_work' ,header=0,
                     na_values= ['',' ','NA','na','Na','N/A','N/a','n/a'])
################################################################################
#Defining feature as categorical
Categorical_col = ['Reason for absence', 'Month of absence', 'Day of the
week',
       'Seasons', 'Disciplinary failure', 'Education', 'Social drinker',
       'Social smoker']
Non_Categorical_col = [i for i in data.columns if i not in Categorical_col]
for i in data.columns:
    if i in Categorical_col: data[i] = pd.Categorical(data[i])
```

```
# Self Defined function
################################################################################
# Return the percentage of missing data in the original dataset
def PerOfMissing(d1,d2):# d1--data by droping the NAN value d2--Original data
    percent_of_missing_data = round( 100 - ((len(d1)/len(d2))*100), 2)
    percent_of_missing_data = str(percent_of_missing_data) + '% of data has
Missing value'
    return percent_of_missing_data

################################################################################
# Return MAE, MRSE, R², Adjusted R²
def reg_acc(y_true, y_pre):
    from math import sqrt
    from sklearn.metrics import mean_absolute_error
    from sklearn.metrics import mean_squared_error
    from sklearn.metrics import r2_score
    print ("MSE of data: ", mean_squared_error(y_true,y_pre))
    print ("***RMSE of data: ", sqrt(mean_squared_error(y_true,y_pre)),'***')
    print ('Other Parameters:')
    print ("R² : ", r2_score(y_true,y_pre))
    print ('MAE:',mean_absolute_error(y_true,y_pre))
```

```
#Creating list of columns name on basis of NAN value
col_with_nan, col_without_nan = [],[]
for i in data.columns:
    if data[i].isnull().sum() > 0:
        col_with_nan.append(i)
    else :col_without_nan.append(i)

#Creating the table which columns has how much missing value
missing_val = pd.DataFrame(data.isnull().sum())
missing_val = missing_val.reset_index()
missing_val = missing_val.rename(columns={'index': 'Variables', 0:
'Missing_Val'})
missing_val['Missing_per'] =
round((missing_val['Missing_Val']/len(data))*100,2)
missing_val = missing_val.sort_values('Missing_per', ascending=
False).reset_index(drop = True)
missing_val
```

```
#random Number
index_NO_nan = data.dropna().index
random_index = []
for i in range(len(col_with_nan)):
    random_index.append(random.choice(index_NO_nan))
##############################################################################
#replacing data with nan
dum = data.copy()
for i in range(len(col_with_nan)):
    dum[col_with_nan[i]].loc[random_index[i]] = np.nan
##############################################################################
#created table
actV,meanV,medianV = [],[],[]
for i in range(len(col_with_nan)):
    actV.append(data[col_with_nan[i]].loc[random_index[i]])

for i in range(len(col_with_nan)):
    if col_with_nan[i] in Categorical_col:
        meanV.append(np.nan)#Categorical data has no mean(Error occur )
        medianV.append(np.nan)#Categorical data has no median
    else:
        meanV.append(data[col_with_nan[i]].mean())
        medianV.append(data[col_with_nan[i]].mean())

frame ={'Actual Value': actV,'Mean': meanV, 'Median': medianV}
data_frame = pd.DataFrame(data = frame, index = col_with_nan, dtype =int)
```

```
# Applying the KNN imputation over the data (k = 3)
process_data = KNN(k=3).complete(data)
process_data = pd.DataFrame(data = process_data, columns=data.columns)
##############################################################################
#############
# So, now on we will proccess further in KNN_pre_data
##############################################################################
#############
# converting all the non categorical value in integer

for i in process_data.columns:
    process_data[i]= process_data[i].astype('int')

for i in Categorical_col:
    process_data[i] = pd.Categorical(process_data[i])
```

```
#Generating the correlation heatmap
plt.figure(figsize = (15, 10))
heat = sns.heatmap(process_data.drop(['ID'],axis=1).corr(), annot = True)
heat.figure.savefig('Heatmap.png')
```

```
# As per the heat map Weight is highly correlated with Body mass index.
Moreover, weight divides with square of an height
#  is equal body mass index so weight and height column will be drop from the
data set
process_data = process_data.drop(['Height','Weight'],axis=1)
```

```
# All the numerical data will be scaled
scaling_col = ['Transportation expense', 'Distance from Residence to
Work','Service time',
               'Work load Average/day ', 'Hit target','Body mass index']

for i in scaling_col:
    process_data[i]=(process_data[i]-
min(process_data[i]))/(max(process_data[i])-min(process_data[i]))
    print(i,': Scaling Done')
```

```
# Spliting the data set in train-test
from sklearn.model_selection import train_test_split
Pro_X = process_data.drop(['Absenteeism time in hours'], axis=1)
Pro_Y = process_data['Absenteeism time in hours']
X_TRAIN,X_TEST,Y_TRAIN,Y_TEST = train_test_split(Pro_X,Pro_Y, test_size=0.2)
```

```
# Applying KNN Machine learnig algorithm
from sklearn.neighbors import KNeighborsRegressor
Pro_KNN_model = KNeighborsRegressor(n_neighbors=3).fit(X_TRAIN,Y_TRAIN)
Final_KNN_predict = Pro_KNN_model.predict(X_TEST)
reg_acc(Y_TEST,Final_KNN_predict)
```

```
#Applying linear regression machine learning algorithm
from sklearn.linear_model import LinearRegression
linReg = LinearRegression().fit(X_TRAIN,Y_TRAIN)
linear_predict = linReg.predict(X_TEST)
reg_acc(Y_TEST,linear_predict)
print('Intercept :',linReg.intercept_)
print('Coefficient')
for i,coef in zip(process_data.columns,linReg.coef_):
    print (i,"-->",coef)
```

```
#Applying all the different machine learning algorithm to check which produce
the best result
from sklearn import tree
Deci_tree_pre =
tree.DecisionTreeRegressor().fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
print('Decision Tree ML')
reg_acc(Deci_tree_pre,Y_TEST)
print('######END########')
##################################################################
from sklearn.ensemble import RandomForestRegressor
Ran_for_pre =
RandomForestRegressor(n_estimators=50).fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
```

```
print('Random Forest ML')
reg_acc(Ran_for_pre,Y_TEST)
print('######END########')
#################################################################
from sklearn.naive_bayes import GaussianNB
print('Naive Bayes')
NB_pre = GaussianNB().fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
reg_acc(NB_pre,Y_TEST)
```

```
list_mae,list_mse,list_r2,num=[],[],[],[]
for i in range(50):
    if i%2!=0:
        dummy_KNN =
KNeighborsRegressor(n_neighbors=i).fit(X_TRAIN,Y_TRAIN).predict(X_TEST)
        num.append(i)
        list_mae.append(mean_absolute_error(dummy_KNN,Y_TEST))
        list_mse.append(mean_squared_error(dummy_KNN,Y_TEST))
        list_r2.append(r2_score(dummy_KNN,Y_TEST))


sns.lineplot(x=num,y=list_mae)
print('********** M A E **********')
print('Kth Value:',num[list_mae.index(min(list_mae))])
print('Lowest MAE Value:',min(list_mae))



sns.lineplot(x=num,y=list_mse)
print('********** M S E **********')
print('Kth Value:',num[list_mse.index(min(list_mse))])
print('Lowest MSE Value:',min(list_mse))


sns.lineplot(x=num,y=list_r2)
print('********** R2 Value **********')
print('Kth Value',num[list_r2.index(max(list_r2))])
print('Lowest R2 Value:',max(list_r2))
```

Appendix B – R Code

```
###################Setting directory and files###################
setwd('D:/Data Science/EDWISOR/2_PORTFOLIO/project 1')
x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "unbalanced",
"C50", "dummies", "e1071", "Information",
      "MASS", "rpart", "gbm", "ROSE", 'sampling', 'DataCombine',
'inTrees','readxl')
lapply(x, require, character.only = TRUE)
absent_data = read_xls('Absenteeism_at_work_Project.xls',sheet = 1,col_names
= TRUE)
anyNA(absent_data)
absent_data= as.data.frame(absent_data)
col_names = colnames(absent_data)
categorical_col = c('Reason for absence', 'Month of absence', 'Day of the
week','Seasons', 'Disciplinary failure', 'Education', 'Social drinker',
'Social smoker')
```

```
####################KNN imputation###################
knn_process_data=knnImputation(absent_data,k = 3)
knn_process_data=as.data.frame(knn_process_data)
#Created an knn_process_data by applying KNN imputation algorithm

for(i in colnames(knn_process_data)){
  knn_process_data[,i]=as.integer(knn_process_data[,i])
}

for (i in categorical_col){
  knn_process_data[,i]=as.factor(knn_process_data[,i])
}
###################Feature Selection###################
#corrgram(knn_process_data[,numeric_index],order = FALSE,
#         upper.panel = panel.pie,text.panel = panel.txt,main= 'Correlation
Plot')

knn_process_data = subset(knn_process_data,select = -c(Weight,Height))
knn_process_data= as.data.frame(knn_process_data)
```

```
###################Featrue Scaling###################

scaling_col = c("Transportation expense", "Distance from Residence to
Work","Service time",
                "Work load Average/day" ,"Hit target","Body mass index")
for (i in scaling_col){
  print(i)
  knn_process_data[,i]= (knn_process_data[,i]-
min(knn_process_data[,i]))/(max(knn_process_data[,i])-
min(knn_process_data[,i]))
}
```

```
####################Modelling###################
rmExcept(c("knn_process_data","absent_data"))
set.seed(121)
train_index=createDataPartition(knn_process_data$`Absenteeism time in
hours`,p = 0.8,list = FALSE)
train=knn_process_data[train_index,]
test=knn_process_data[-train_index,]
train$`Absenteeism time in hours`=as.factor(train$`Absenteeism time in
hours`)
test$`Absenteeism time in hours`=as.factor(test$`Absenteeism time in hours`)



#KNN Algo
library(class)
train$`Absenteeism time in hours`=as.numeric(train$`Absenteeism time in
hours`)
knn_model = knnreg(train[,-19],train$`Absenteeism time in hours`,k = 3)
predict_knn =predict(knn_model,test[,-19])

measureRAE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_knn))
measureRSQ(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_knn))
measureRMSE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_knn))
summary(knn_model)



#Linear Regession
col = colnames(train)
pre = col[19]
col=col[-19]
train$`Absenteeism time in hours`=as.numeric(train$`Absenteeism time in
hours`)
linreg =  lm(`Absenteeism time in hours`~ .,data = train)
predict_linreg = predict(linreg,test[,1:18])

measureMAE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_linreg))
measureMSE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_linreg))
measureRMSE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_linreg))




#Decision Tree
Dec_model = rpart(`Absenteeism time in hours`~ .,data = train)
summary(Dec_model)
predict_Dec = predict(Dec_model,test[,-19])

measureMAE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_Dec))
measureRSQ(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_Dec))
measureRMSE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_Dec))
```

```
#Random Forest
rf_model = randomForest(x=train[,-19],y=train$`Absenteeism time in
hours`,importance = TRUE,ntree = 500)
summary(rf_model)
predict_rf = predict(rf_model,test[,-19])
RMSE(pred = as.numeric(predict_rf), obs = as.numeric(test$`Absenteeism time
in hours`))
importance(rf_model,type = 1)
measureRAE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_rf))
measureRSQ(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_rf))
measureRMSE(as.numeric(test$`Absenteeism time in
hours`),as.numeric(predict_rf))
```