EMPLOYEE ABSENTEESIM

Problem Statement

We have receive the data of XYZ courier company where the organization want to know the trend & reason of their employee absent & reduce the absenteeism in future as it effect the organization growth both in generating profit & production hours. This is a regression problem & we have design the model according to that.

Data:

1. Individual identification (ID).
2. Reason for absence (ICD).
3. Month of absence
4. Day of the week (Monday (2), Tuesday (3), Wednesday (4), Thursday (5), Friday (6))
5. Seasons (summer (1), autumn (2), winter (3), spring (4)
6. Transportation expense
7. Distance from Residence to Work (kilometers)
8. Service time
9. Age
10. Work load Average/day
11. Hit target
12. Disciplinary failure (yes=1; no=0)
13. Education (high school (1), graduate (2), postgraduate (3), master and doctor (4))
14. Son (number of children)
15. Social drinker (yes=1; no=0)
16. Social smoker (yes=1; no=0)
17. Pet (number of pet)
18. Weight
19. Height
20. Body mass index
21. Absenteeism time in hours

Here the target variable is Absenteeism time in hours.

Reason for absence(ICD)

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.

Exploratory Data Analysis

As the data has missing value ,we can’t feed the incomplete data to the model. So before going to ML model we have to clean & process the data & impute missing value, check for outliers & do Feature Selection & Feature Scaling.

Missing Value Analysis

Columns	Missing Percentage	No. of missing Values
Body Mass Index	4.18	31
Absenteeism in hours	2.97	22
Height	1.89	14
Workday Average/day	1.35	10
Education	1.35	10

Transportation Expense	0.94	7
Hit Target	0.81	6
Disciplinary failure	0.81	6
Son	0.81	6
Social Smoker	0.54	4
Reason for absence	0.40	3
Distance from home to work	0.40	3
Service time	0.40	3
Age	0.40	3
Social Drinker	0.40	3
Pet	0.27	2
Month of absence	0.13	1
Weight	0.13	1
ID	0	0
Day of week	0	0
Seasons	0	0

We have to first check the correct method for imputation between mean ,median & KNN,by replacing any present value with NA & then imputing using above method & compare the imputed value with original value .

Sample R Code

```
#Absentism Time in hours

#reference NA to check best method for fare

edf[3,21]=NA

#Mean method

edf$`Absenteeism time in hours`[is.na(edf$`Absenteeism time in hours`)]=mean(edf$`Absenteeism time in hours`,na.rm =T)

#Actual=2

#Analysis=6.98

#Median method

edf$`Absenteeism time in hours`[is.na(edf$`Absenteeism time in hours`)]=median(edf$`Absenteeism time in hours`,na.rm =T)

#Actual=2

#Analysis=3

library("VIM")
```

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=2

#Analysis=2

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=2

#Analysis=4

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=2

#Analysis=4

ANALYSIS MISSING VALUE

Columns	Actual	Mean	Median	K=3	K=5	K=7
Body Mass Index	25	26.7	25	25	25	25
Absenteeism in hours	2	6.98	3	2	4	4
Height	170	172	170	170	170	170
Workday Average/day	239554	271232	264249	265615	249797	249797
Education	3	NA	NA	3	3	3
Transportation Expense	179	221	225	179	179	179
Hit Target	91	94.6	95	97	93	97
Disciplinary failure	0	NA	NA	0	0	0
Son	2	2	1	2	2	2
Social Smoker	1	NA	NA	1	1	1
Reason for absence	7	NA	NA	14	14	14
Distance from home to work	36	29.7	26	36	36	36
Service time	11	12.6	13	11	11	11
Age	39	36.4	37	39	39	43
Social Drinker	1	NA	NA	1	1	1
Pet	2	NA	NA	2	2	2
Month of absence	7	NA	NA	3	7	7
Weight	90	79	83	90	90	90

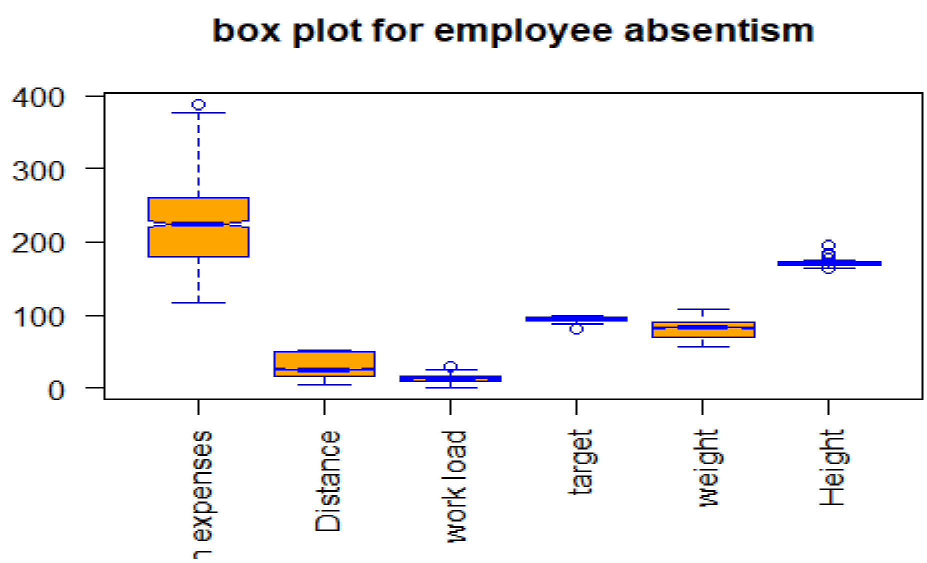
Since Education, disciplinary failure, Social Smoker, Reason for Absence, Social Drinker, Pet, Month of Absence are categorical variable we are using only KNN Imputation.

KNN Imputation with k=5 give the nearest or exact value in most of the cases, so we use this method to finally impute the missing values.

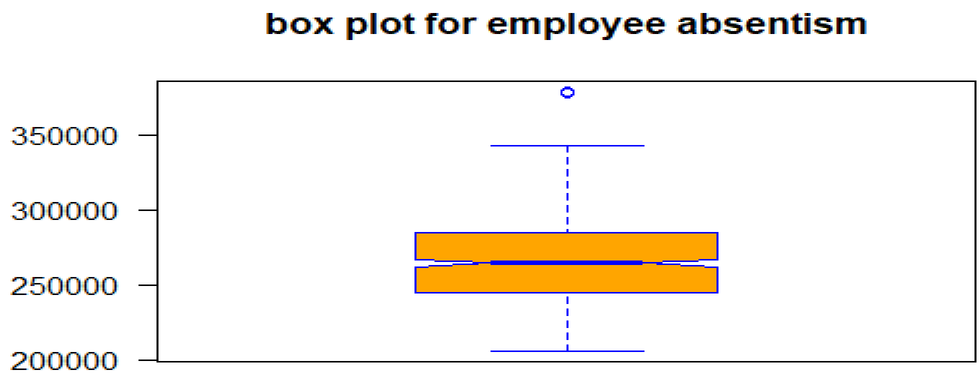
Outlier Analysis

Outlier analysis is a part of data cleaning where we have find the outliers(which fall away from the dataset) & if we got maximum outliers for a particular variable, we can either impute the outliers or can drop the outliers.

The best method to find the outliers is to design the box plot



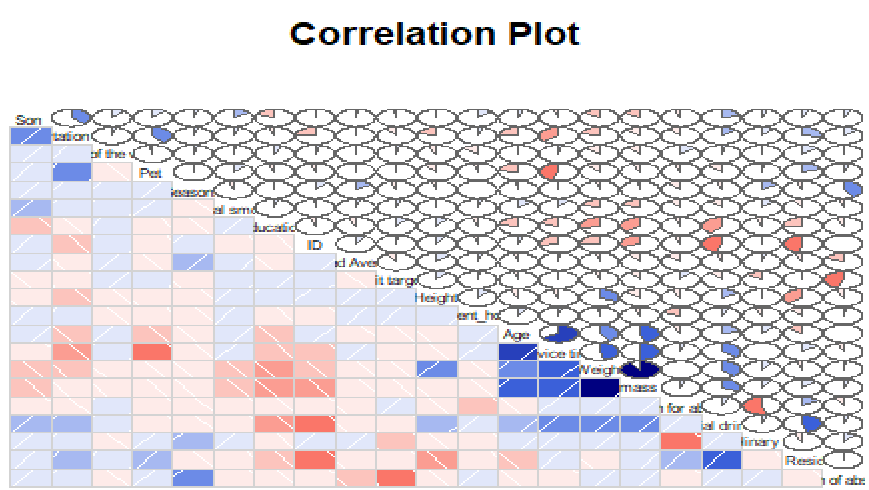
Box plot for work load.As



As we can visualize that there are not much outliers in the above box plot we don't required to drop any variable or do imputaion.

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 variable are highly negatively or positively correlated then we can drop any one variable as taking two variable to model will give no meaning rather increase the complexity of model.



We can see body mass & weight are highly correlated, Body Mass is the ratio of weight & square of height. So we can drop two variable **height & weight.**

Feature Scaling

Some variable have different range, unit by which there may not be proper scale between two variables. Feature Scaling is a technique where we can limit the range of the variables that can compete in common ground.

Sample Python code.(Normalization)

```
cnames1=["Transport_expense","Distance","Service_time","Work_load","Hit_target"]

for j in cnames1:
print(j)
edf[j]=(edf[j]-min(edf[j]))/(max(edf[j])-min(edf[j]))
```

Here we are scaling 5 variable i.e
'Transport_expense','Distance','Service_time','Work_load','Hit_target'.

MODELLING

Linear Regression

It is a light weight & statistical model. It describe the relationship between the variables.

Python code

```
#LINEAR REGRESSION

model = sm.OLS(train.iloc[:,18], train.iloc[:,0:17]).fit()

predictions_LR = model.predict(test.iloc[:,0:17])
```

```
Call:
lm(formula = Absent_hours ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max 
-31.961  -5.101  -1.796   1.822 106.704 

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)    
(Intercept)         34.13278    8.41823   4.055 5.82e-05 ***
ID                  -0.19960    0.08015  -2.490  0.01309 *  
Absent_reason       -0.50849    0.08486  -5.992 3.97e-09 ***
Absent_month         0.20893    0.23012   0.908  0.36436    
```

```
Absent_day          -1.13309    0.43851  -2.584  0.01005 *
Seasons             -0.54885    0.60658  -0.905  0.36599
Transport_expense    1.23627    3.19874   0.386  0.69930
Distance            -6.92349    2.60507  -2.658  0.00812 **
Service_time        -1.98291    7.18259  -0.276  0.78261
Age                  0.36705    0.14982   2.450  0.01463 *
Work_load           -0.84736    2.87684  -0.295  0.76846
Hit_target           4.34002    3.44425   1.260  0.20823
Discipline_failure -19.00519    3.33583  -5.697 2.08e-08 ***
Education           -2.58653    1.17065  -2.209  0.02759 *
Son                  0.86630    0.64364   1.346  0.17893
Social_drinker       0.71342    1.66637   0.428  0.66874
Social_smoker       -6.10322    2.47894  -2.462  0.01415 *
Pet                 -0.21339    0.56870  -0.375  0.70766
Body_mass           -0.65743    0.20693  -3.177  0.00158 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.24 on 499 degrees of freedom
Multiple R-squared:  0.1695,   Adjusted R-squared:  0.1395
F-statistic: 5.658 on 18 and 499 DF,  p-value: 2.595e-12
```

The variable which has 3 star or maximum no. of star gives the maximum contribution to the model & describing the target variables.

Here Adjusted R-squared is 0.1395 which means this model will help 13.9% in predicting the target value.

Decision Tree

It is a predictive model based on a branching series of Boolean tests(from past experience).This model can be used for both classification & regression. It is a rule & each branch connect with "and" & multiple branch are connected by "OR".

Python code

```
#DECISIONS TREE

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeRegressor

fit_DT = DecisionTreeRegressor(max_depth=3).fit(train.iloc[:,0:17], train.iloc[:,18])

predictions_DT = fit_DT.predict(test.iloc[:,0:17])
```

Random Forest

Random Forest is an ensemble that consists of many decision trees. The method combine Breiman's "bagging" idea(feeding error to next tree) & the random selection of features. Random forest use CART algorithm which use Gini Index.

Python Code:

```
from sklearn.ensemble import RandomForestRegressor

RF_model = RandomForestRegressor(n_estimators = 700).fit(train.iloc[:,0:17], train.iloc[:,18])

predictions_RF = RF_model.predict(test.iloc[:,0:17])
```

In the above set of code we change n_estimators(300,500,700,50),& lock the code which give less error.

KNN METHOD

K-nearest Neighbour predicts the value by checking the distance with from other feature with respect to the Kth value.

Applying the K-nearest Neighbours on our data set with n_neighbors=3(7,5,9). Checking which kth value fit best for the respective data set.

Python Code

```
from  sklearn.neighbors   import  KNeighborsRegressor

KNN_model=KNeighborsRegressor(n_neighbors=9).fit(train.iloc[:,0:17], train.iloc[:,18])

KNN_Predictions=KNN_model.predict(test.iloc[:,0:17])
```

ERROR METRICS

Here RMSE & MAE is used as an error metrics to check which model gives more accurate result.

For R

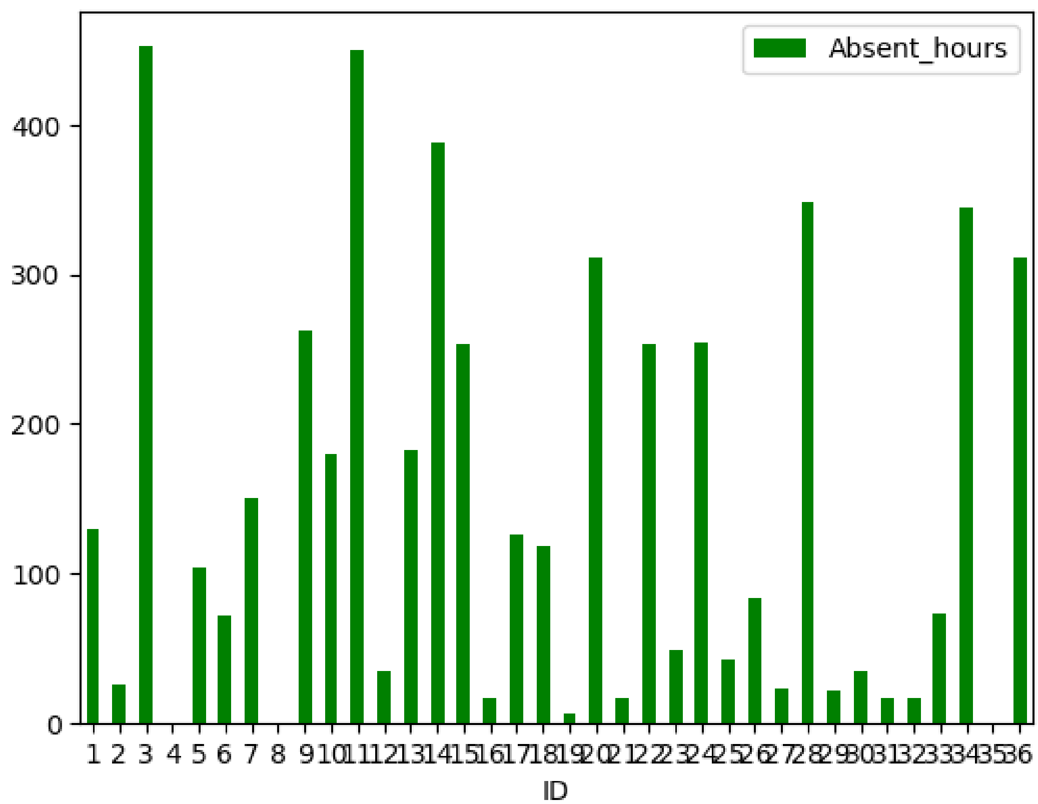
	Linear Regression	Decision Tree	Random Forest	KNN
MAE	5.85	5.38	5.01	4.38
RMSE	10.46	11.31	12.66	10.37

For Python

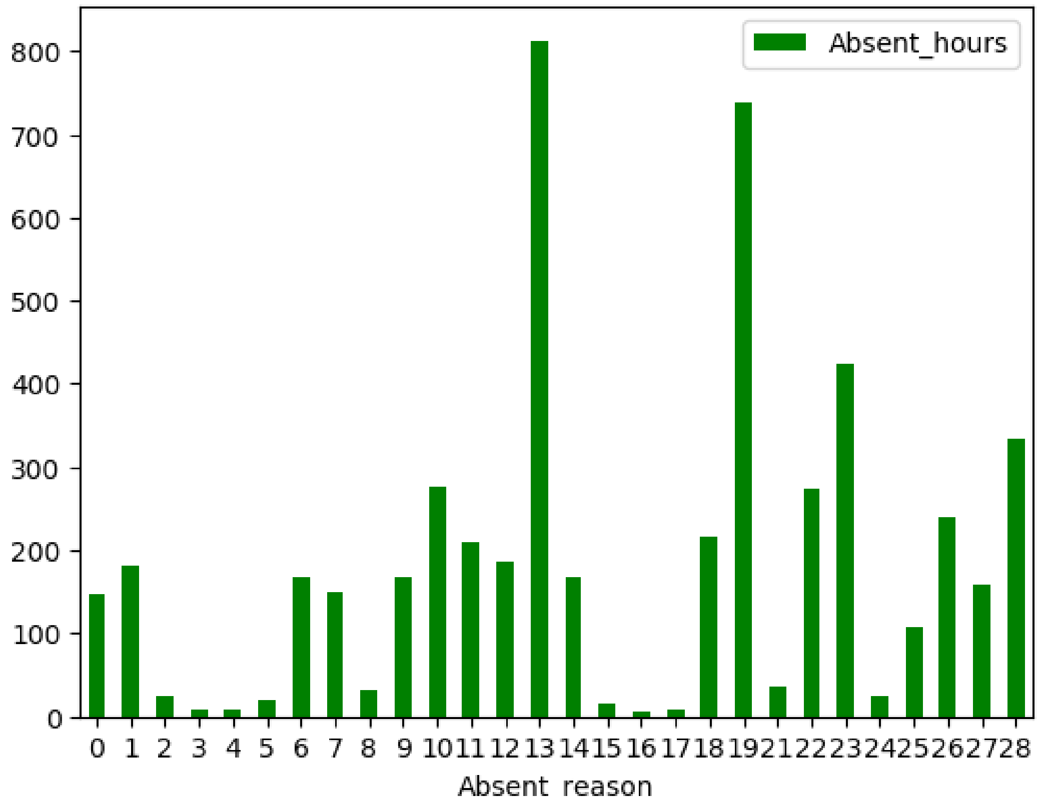
	Linear Regression	Decision Tree	Random Forest	KNN
MAE	5.23	4.69	4.97	4.77
RMSE	7.52	8.78	10.26	7.84

KNN Model gives less error rate & will give much accurate result for the target variable(Absenteeism in hours).So,we will use KNN model for predictions for further data.

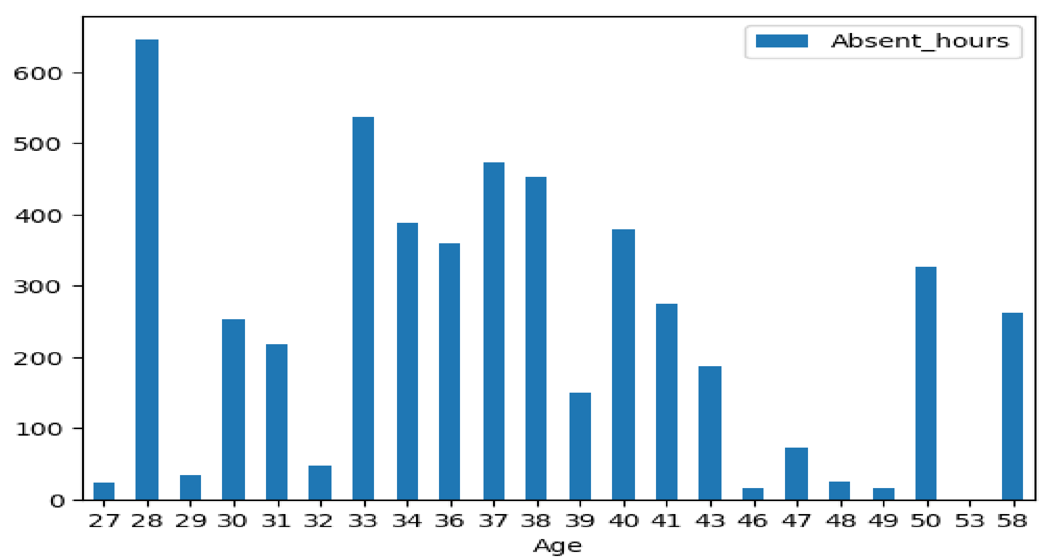
SUMMARY



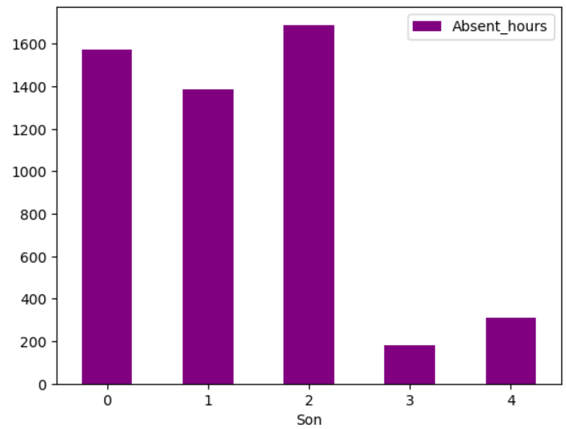
In the above ID vs Absent_hours bar graph we can visualize that Employee ID 3 &11 have maximum no. of absent_hours 450 & 453 respectively.

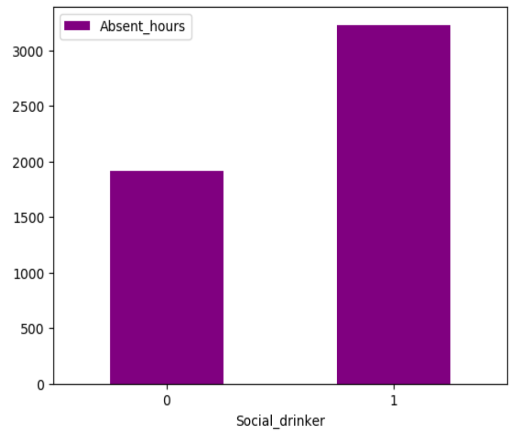


In the above graph we can visualize that people has suffered mostly from **DISEASES OF THE MUSCULOSKELETAL SYSTEM AND CONNECTIVE TISSUE** & **INJURY, POISONING AND CERTAIN OTHER CONSEQUENCES OF EXTERNAL CAUSES .** which together contribute 1550 loss of hours.



The above is a graph for Age vs Absent hours which shows that people of age 28,32,37,38 has maximum no. of absent hours. Employee ID 3 & 11 has age of 38 & 33 respectively & they have several health issue & ID 11 has 1 pet also.





The graph shows that the employee who drinks contributing more to the absent hours & when it comes to no. of children, the employee which has maximum no. of children are contributing less to absent_hours.

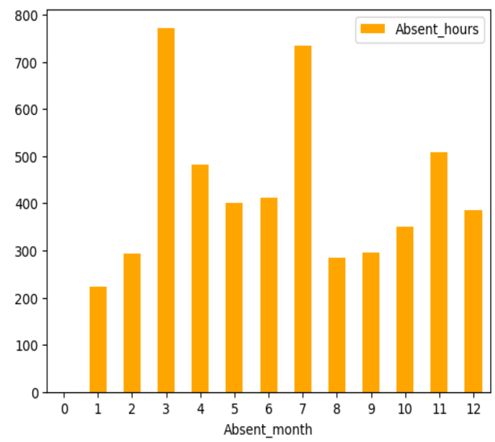
1. **What changes company should bring to reduce the number of absenteeism?**
   Ans:
   - ➔ Employee ID 3 & 11 should be taken into consideration as they have more absent hours.
   - ➔ There are some serious health issue suffered by employees like
     1) DISEASES OF THE MUSCULOSKELETAL SYSTEM AND CONNECTIVE TISSUE
     2) INJURY, POISONING AND CERTAIN OTHER CONSEQUENCES OF EXTERNAL CAUSES
     3) DISEASES OF THE RESPIRATORY SYSTEM

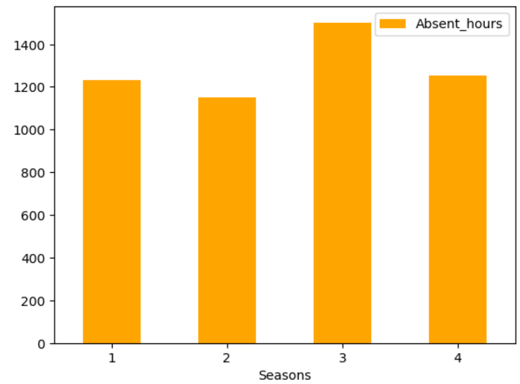
        There are more absent_hours of drinkers & injury due to external cause. Organization should take care of their employee health & give more attention towards Social_drinkers.

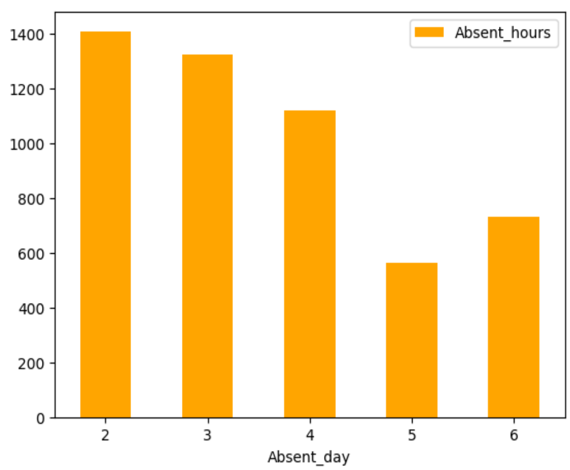
   - ➔ There are 240 hours of absent where reason are un justified, the company should look forward to find out the reason.
   - ➔ DISEASES OF THE RESPIRATORY SYSTEM & DISEASES OF THE MUSCULOSKELETAL SYSTEM AND CONNECTIVE TISSUE might be occurring because of some envorinmenal effect or pollution.Organisation should take this as consideration.

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

Ans:







As we can see there are more no. of absent_hours in winter, so the company should expect a maximum no. or absent_hours in 2011 winter & more no. of absent hours in Monday Company should expect a maximum no. of hours loss in the month of march & July.

R CODE

```
#clear the envorinment

rm(list=ls())

#set working directory

setwd("E:/data science and machine learning/Employee Absentism project 2/R Code")

#current working directory

getwd()

library(readxl)

#loading file

edf=read_excel("Absentism.xls")

#Getting colnames

colnames(edf)

#getting datatype of each variable

str(edf)

#loading some of the libraries

x=c("ggplot2","Corrgram","DMwR","Caret","randomForest","unbalanced","C50","dummies","e10","
MASS","rpart","gbm","ROSE")

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

library("VIM")

#finding out number of missing value

missing_val=data.frame(apply(edf,2,function(x){sum(is.na(x))}))

#giving names in dataframe

missing_val$Columns=row.names(missing_val)

row.names(missing_val)=NULL

names(missing_val)[1]="Missing_Percentage"

#converting to percentage
```

```
missing_val$Missing_Percentage=(missing_val$Missing_Percentage/nrow(edf))*100

#Arranging in descending order

missing_val=missing_val[order(-missing_val$Missing_Percentage),]

#MISSING VALUE ANALYSIS(checking correct method)

#Body Mass Index

#reference NA to check best method for fare

edf[9,20]=NA

#Mean method

edf$`Body mass index`[is.na(edf$`Body mass index`)]=mean(edf$`Body mass index`,na.rm =T)

#Actual=25

#Analysis=26.7

#Median method

edf$`Body mass index`[is.na(edf$`Body mass index`)]=median(edf$`Body mass index`,na.rm =T)

#Actual=25

#Analysis=25

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=25

#Analysis=25

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=25

#Analysis=25

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=25
```

```
#Analysis=25

#Absentism Time in hours

#reference NA to check best method for fare

edf[3,21]=NA

#Mean method

edf$`Absenteeism time in hours`[is.na(edf$`Absenteeism time in hours`)]=mean(edf$`Absenteeism time in hours`,na.rm =T)

#Actual=2

#Analysis=6.98

#Median method

edf$`Absenteeism time in hours`[is.na(edf$`Absenteeism time in hours`)]=median(edf$`Absenteeism time in hours`,na.rm =T)

#Actual=2

#Analysis=3

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=2

#Analysis=2

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=2

#Analysis=4

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=2

#Analysis=4
```

```
#Height

#reference NA to check best method for fare

edf[3,19]=NA

#Mean method

edf$Height[is.na(edf$Height)]=mean(edf$Height,na.rm =T)

#Actual=170

#Analysis=172

#Median method

edf$Height[is.na(edf$Height)]=median(edf$Height,na.rm =T)

#Actual=170

#Analysis=170

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=170

#Analysis=170

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=170

#Analysis=170

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=170

#Analysis=170
```

#Work load average/day

#reference NA to check best method for fare

edf[1,10]=NA

#Mean method

edf$`Work load Average/day`[is.na(edf$`Work load Average/day`)]=mean(edf$`Work load Average/day`,na.rm =T)

#Actual=239554

#Analysis=271232

#Median method

edf$`Work load Average/day`[is.na(edf$`Work load Average/day`)]=median(edf$`Work load Average/day`,na.rm =T)

#Actual=239554

#Analysis=264249

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=239554

#Analysis=265615

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=239554

#Analysis=249797

#Knn method(7)

edf=kNN(edf,k=7)

```
#Actual=239554

#Analysis=249797

#Education

#reference NA to check best method for fare

edf[505,13]=NA

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=3

#Analysis=3

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=3

#Analysis=3

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=3

#Analysis=3

#Transportation expenses

#reference NA to check best method for fare

edf[6,6]=NA

#Mean method

edf$`Transportation expense`[is.na(edf$`Transportation expense`)]=mean(edf$`Transportation expense`,na.rm =T)

#Actual=179

#Analysis=221
```

```
#Median method

edf$`Transportation expense`[is.na(edf$`Transportation expense`)]=median(edf$`Transportation expense`,na.rm =T)

#Actual=179

#Analysis=225

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=179

#Analysis=179

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=179

#Analysis=179

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=179

#Analysis=179

#Hit target

#reference NA to check best method for fare

edf[530,11]=NA

#Mean method

edf$`Hit target`[is.na(edf$`Hit target`)]=mean(edf$`Hit target`,na.rm =T)

#Actual=91

#Analysis=94.6
```

```
#Median method

edf$`Hit target`[is.na(edf$`Hit target`)]=median(edf$`Hit target`,na.rm =T)

#Actual=91

#Analysis=95

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=91

#Analysis=97

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=91

#Analysis=93

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=91

#Analysis=97

#Disciplinary failure

#reference NA to check best method for fare

edf[1,12]=NA

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=0

#Analysis=0
```

```
#Knn method(5)

edf=kNN(edf,k=5)

#Actual=0

#Analysis=0

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=0

#Analysis=0

#Son

#reference NA to check best method for fare

edf[1,14]=NA

#Mean method

edf$Son[is.na(edf$Son)]=mean(edf$Son,na.rm =T)

#Actual=2

#Analysis=2

#Median method

edf$Son[is.na(edf$Son)]=median(edf$Son,na.rm =T)

#Actual=2

#Analysis=1

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=2

#Analysis=2
```

```
#Knn method(5)

edf=kNN(edf,k=5)

#Actual=2

#Analysis=2

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=2

#Analysis=2

#Social Smoker

#reference NA to check best method for fare

edf[4,16]=NA

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=1

#Analysis=1

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=1

#Analysis=1

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=170

#Analysis=170
```

#Reason for absence

#reference NA to check best method for fare

edf[4,2]=NA

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=7

#Analysis=14

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=7

#Analysis=14

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=7

#Analysis=14

#Distance from residence to work

#reference NA to check best method for fare

edf[5,7]=NA

#Mean method

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

#Actual=36

#Analysis=29.7

#Median method

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

```
#Actual=36

#Analysis=26

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=36

#Analysis=36

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=36

#Analysis=36

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=36

#Analysis=36

#Service time

#reference NA to check best method for fare

edf[8,8]=NA

#Mean method

edf$`Service time`[is.na(edf$`Service time`)]=mean(edf$`Service time`,na.rm =T)

#Actual=11

#Analysis=12.6

#Median method

edf$`Service time`[is.na(edf$`Service time`)]=median(edf$`Service time`,na.rm =T)

#Actual=11

#Analysis=13
```

```
library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=11

#Analysis=11

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=11

#Analysis=11

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=11

#Analysis=11

#Social Drinker

#reference NA to check best method for fare

edf[4,9]=NA

#Mean method

edf$Age[is.na(edf$Age)]=mean(edf$Age,na.rm =T)

#Actual=39

#Analysis=36.4

#Median method

edf$Age[is.na(edf$Age)]=median(edf$Age,na.rm =T)

#Actual=39

#Analysis=37

library("VIM")
```

```
#Knn method(3)

edf=kNN(edf,k=3)

#Actual=39

#Analysis=97

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=91

#Analysis=93

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=170

#Analysis=170

#Hit target

#reference NA to check best method for fare

edf[4,15]=NA

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=1

#Analysis=1

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=1

#Analysis=1

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=1
```

#Analysis=1

#Pet

#reference NA to check best method for fare

edf[182,17]=NA

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=2

#Analysis=2

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=2

#Analysis=2

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=2

#Analysis=2

#month of absence

#reference NA to check best method for fare

edf[4,3]=NA

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=7

#Analysis=3

#Knn method(5)

edf=kNN(edf,k=5)

```
#Actual=7

#Analysis=7

#Knn method(7)

edf=kNN(edf,k=7)

#Actual=7

#Analysis=7

#Weight

#reference NA to check best method for fare

edf[1,18]=NA

#Mean method

edf$Weight[is.na(edf$Weight)]=mean(edf$Weight,na.rm =T)

#Actual=90

#Analysis=79

#Median method

edf$Weight[is.na(edf$Weight)]=median(edf$Weight,na.rm =T)

#Actual=90

#Analysis=83

library("VIM")

#Knn method(3)

edf=kNN(edf,k=3)

#Actual=90

#Analysis=90

#Knn method(5)

edf=kNN(edf,k=5)

#Actual=90

#Analysis=90
```

```
#Knn method(7)

edf=kNN(edf,k=7)

#Actual=90

#Analysis=90


#MISSING VALUE ANALYSIS(Locking the method)

#Knn method(5)

edf=kNN(edf,k=5)

edf=edf[,c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21)]

write.csv(edf,"Absent.csv",row.names = T)

#finding out number of missing value

missing_val=data.frame(apply(edf,2,function(x){sum(is.na(x))}))

#giving names in dataframe

missing_val$Columns=row.names(missing_val)

row.names(missing_val)=NULL

names(missing_val)[1]="Missing_Percentage"

#converting to percentage

missing_val$Missing_Percentage=(missing_val$Missing_Percentage/nrow(edf))*100

colnames(edf)[colnames(edf)=="Absenteeism time in hours"]="Absent_hours"

colnames(edf)[colnames(edf)=="Transportation expense"]="Transport_expense"

colnames(edf)[colnames(edf)=="Distance from Residence to Work"]="Distance"

colnames(edf)[colnames(edf)=="Service time"]="Service_time"

colnames(edf)[colnames(edf)=="Work load Average/day"]="Work_load"

colnames(edf)[colnames(edf)=="Hit target"]="Hit_target"

colnames(edf)[colnames(edf)=="Reason for absence"]="Absent_reason"
```

```
colnames(edf)[colnames(edf)=="Month of absence"]="Absent_month"

colnames(edf)[colnames(edf)=="Day of the week"]="Absent_day"

colnames(edf)[colnames(edf)=="Disciplinary failure"]="Discipline_failure"

colnames(edf)[colnames(edf)=="Social drinker"]="Social_drinker"

colnames(edf)[colnames(edf)=="Social smoker"]="Social_smoker"

colnames(edf)[colnames(edf)=="Body mass index"]="Body_mass"


#OUTLIER ANALYSIS


p1=edf$`Transportation expense`

p2=edf$`Distance from Residence to Work`

p3=edf$`Service time`

p4=edf$`Work load Average/day`

p5=edf$`Hit target`

p6=edf$Weight

p7=edf$Height


boxplot(p1,p2,p3,p5,p6,p7,

    main="box plot for employee absentism",

    at=c(1,2,3,4,5,6),

    names=c("Transportation expenses","Distance","work load","target","weight","Height"),

    las=2,

    col="Orange",

    border="blue",

    notch=TRUE

    )
```

```
boxplot(p4,
       main="box plot for employee absentism",
       at=c(1),
       names=c("Work load/day"),
       las=2,
       col="Orange",
       border="blue",
       notch=TRUE
)

#FEATURE SELECTION

numeric_index=sapply(edf,is.numeric)
numeric_data=edf[,numeric_index]
cnames=colnames(numeric_data)

install.packages("corrgram",dependencies = TRUE)
library(knitr)
library(fpca)
library(corrgram)

corrgram(edf[,numeric_index],order=TRUE,upper.panel=panel.pie,text.panel=panel.txt,main="Correlation Plot")

library(ggplot2)
```

```
library(scales)

library(gplots)

library(psych)



#BAR GRAPH

ggplot(data=edf,aes(x=edf$Height,y=edf$`Body mass index`))+

  geom_col(position=position_dodge())


#DROPING HEIGHT & WEIGHT VARIABLE

library(dplyr)

edf=select(edf,-Height,-Weight)


#FEATURE SCALING


cnames=c("Transport_expense","Distance","Service_time","Work_load","Hit_target")

for(i in cnames)

{

   print(i)

   edf[,i]=(edf[,i]-min(edf[,i]))/(max(edf[,i]-min(edf[,i])))

}



#SAMPLING

train_index=sample(1:nrow(edf),0.7*nrow(edf))

train = edf[train_index,]
```

```
test = edf[-train_index,]

#LINEAR REGRESSION

library(rpart)

library(MASS)

library(DMwR)


#checking multicollinearity

library(usdm)

vif(edf[ ,-19])


#checking correlation with thershold 90%

vifcor(edf[,-19], th = 0.9)


#Linear regression model

lm_model = lm(Absent_hours ~., data = train)


#Summary of the model

summary(lm_model)

library(ie2misc)


#Predict

predictions_LR = predict(lm_model, test[,1:18])

rmse_LR=sqrt(mean((test$Absent_hours-predictions_LR)^2))

#mae=mean(abs((test$Absent_hours - predictions_LR)/test$Absent_hours))

mae=mean(abs(test$Absent_hours - predictions_LR))
```

```
#RMSE=10.46

#MAE=5.85


#KNN PREDICTION


library(class)

##predict test data

KNN_predictions=knn(train[,-19],test[,-19],train$Absent_hours,k=5)

str(KNN_predictions)

KNN_predictions=as.numeric(as.character(KNN_predictions))

rmse_KNN=sqrt(mean((test$Absent_hours-KNN_predictions)^2))

mae=mean(abs(test$Absent_hours - KNN_predictions))


#RMSE=10.37

#MAE=4.38


#DECISION TREE


#rpart for regression

fit = rpart(Absent_hours ~ ., data = train, method = "anova")


#Predict for new test cases

predictions_DT = predict(fit, test[,-19])

rmse_DT=sqrt(mean((test$Absent_hours-predictions_DT)^2))

mae=mean(abs(test$Absent_hours - predictions_DT))
```

```
#RMSE=11.31

#MAE=5.38


#RANDOM FOREST

library(randomForest)

library(RRF)

library(inTrees)


RF_Model=randomForest(Absent_hours~.,train,importance=TRUE,ntree=100)


#extract rules

treelist=RF2List(RF_Model)


exec=extractRules(treelist,train[,-1])

#visualize some rules

exec[1:2,]


#Make rules more readable

readableRules=presentRules(exec,colnames(train))

readableRules[1:4,]


#get rule metrics

ruleMetric=getRuleMetric(exec,train[,-1],train$Absent_hours)


ruleMetric[1:2,]
```

```
#prediction of test data using RF Model

RF_Prediction=predict(RF_Model,test[,-19])

rmse_RF=sqrt(mean((test$Absent_hours-RF_Prediction)^2))

mae=mean(abs(test$Absent_hours - RF_Prediction))


#RMSE=12.66

#MAE=5.01
```

PYTHON CODE

```
#IMPORTING REQUIRED LIBRARY

import os

import pandas as pd

import numpy as np

import matplotlib as plt

import datetime as dt

import seaborn as sns

import import_ipynb

import matplotlib.pyplot as plt1

%matplotlib inline

import sklearn
```

```
from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeRegressor

from sklearn.ensemble import RandomForestRegressor

import statsmodels.api as sm

from sklearn.neighbors import KNeighborsRegressor

#SETTING WORKING DIRECTORY

os.chdir("E:/data science and machine learning/Employee Absentism project 2/Python")

os.getcwd()

#GETTING THE FILE FROM HDD

edf=pd.read_csv("Absent.csv",sep=',')

type(edf)

edf.columns

edf.dtypes

missing_val=pd.DataFrame(edf.isnull().sum())

missing_val

del edf['Unnamed: 0']

edf.columns

#BOX PLOT TO CHECK OUTLIERS OF EVERY VARIABLE

plt.boxplot(edf['Absent_hours'])

cnames=['ID', 'Absent_reason', 'Absent_month', 'Absent_day', 'Seasons',

    'Transport_expense', 'Distance', 'Service_time', 'Age', 'Work_load',

    'Hit_target', 'Discipline_failure', 'Education', 'Son',

    'Social_drinker', 'Social_smoker', 'Pet', 'Weight', 'Height',

    'Body_mass']

plt.boxplot(edf['Distance'])

plt.boxplot(edf['Transport_expense'])
```

```
plt.boxplot(edf['Service_time'])

plt.boxplot(edf['Age'])

plt.boxplot(edf['Hit_target'])

plt.boxplot(edf['Body_mass'])

#FEATURE SELECTION

edf_corr=edf.loc[:,cnames]


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

corr=edf_corr.corr()

ax = sns.heatmap(corr)


del edf['Height']

del edf['Weight']

cnames1=["Transport_expense","Distance","Service_time","Work_load","Hit_target"]

edf

edf.dtypes

#NORMALIZATION

for j in cnames1:

    print(j)

    edf[j]=(edf[j]-min(edf[j]))/(max(edf[j])-min(edf[j]))

#SAMPLING

train, test = train_test_split(edf, test_size=0.3)

#LINEAR REGRESSION

model = sm.OLS(train.iloc[:,18], train.iloc[:,0:17]).fit()

predictions_LR = model.predict(test.iloc[:,0:17])

predictions_LR
```

```
#ERROR METRICS

from sklearn.metrics import mean_squared_error

from sklearn.metrics import mean_absolute_error

from math import sqrt


rms_LR = sqrt(mean_squared_error(test['Absent_hours'],predictions_LR))

mae_LR = mean_absolute_error(test.iloc[:,18],predictions_LR)

#DECISIONS TREE

fit_DT = DecisionTreeRegressor(max_depth=3).fit(train.iloc[:,0:17], train.iloc[:,18])

predictions_DT = fit_DT.predict(test.iloc[:,0:17])

predictions_DT

rms_DT = sqrt(mean_squared_error(test['Absent_hours'],predictions_DT))

mae_DT = mean_absolute_error(test.iloc[:,18],predictions_DT)


#RANDOM FOREST

RF_model = RandomForestRegressor(n_estimators = 700).fit(train.iloc[:,0:17], train.iloc[:,18])

predictions_RF = RF_model.predict(test.iloc[:,0:17])

predictions_RF

rms_RF = sqrt(mean_squared_error(test['Absent_hours'],predictions_RF))

mae_RF = mean_absolute_error(test.iloc[:,18],predictions_RF)

#KNN

KNN_model=KNeighborsRegressor(n_neighbors=9).fit(train.iloc[:,0:17], train.iloc[:,18])

KNN_Predictions=KNN_model.predict(test.iloc[:,0:17])

KNN_Predictions

rms_KNN = sqrt(mean_squared_error(test['Absent_hours'],KNN_Predictions))

mae_KNN = mean_absolute_error(test.iloc[:,18],KNN_Predictions)
```

```
plt1.rcdefaults()

plt1.bar(edf['ID'],edf['Absent_hours'])

edf['ID'].nunique()

edf['ID'].value_counts()

#PLOTTING & GROUPING

edfID=edf.groupby('ID',).sum()[['Absent_hours']]

edfID

edfID.plot.bar(rot=0,color='green')

edfAge=edf.groupby('Age',as_index=True).sum()[['Absent_hours']]

edfAge

edfAge.plot.bar(rot=0)

edfAbs=edf.groupby('Absent_reason',as_index=True).sum()[['Absent_hours']]

edfAbs

edfAbs.plot.bar(rot=0,color='green')

edfSon=edf.groupby('Son',as_index=True).sum()[['Absent_hours']]

edfSon

edfSon.plot.bar(rot=0,color='purple')

edfSD=edf.groupby('Social_drinker',as_index=True).sum()[['Absent_hours']]

edfSD

edfSD.plot.bar(rot=0,color='purple')

edfSS=edf.groupby('Social_smoker',as_index=True).sum()[['Absent_hours']]

edfSS

edfSS.plot.bar(rot=0,color='purple')

edfPet=edf.groupby('Pet',as_index=True).sum()[['Absent_hours']]

edfPet
```

```
edfPet.plot.bar(rot=0,color='purple')

edfmonth=edf.groupby('Absent_month',as_index=True).sum()[['Absent_hours']]

edfmonth

edfmonth.plot.bar(rot=0,color='orange')

edfmonth

edfseason=edf.groupby('Seasons',as_index=True).sum()[['Absent_hours']]

edfseason

edfseason.plot.bar(rot=0,color='orange')

edfday=edf.groupby('Absent_day',as_index=True).sum()[['Absent_hours']]

edfday

edfday.plot.bar(rot=0,color='orange')
```

-Thank You
Sujeet Biswal