Employee Absenteeism

Saurabh Shrivastava

28-07-2020

Contents

Chapter 1

Introduction

1.1 Problem Statement

XYZ is a courier company. As we appreciate that human capital plays an important role

in collection, transportation and delivery. The company is passing through genuine

issue of Absenteeism. The company has shared it dataset and requested to have an

answer on the following areas:

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

   As per our analysis we found below points, which can help company to reduce the number of absenteeism

   a. Employee having distance between 10-30 are more absent, company can make new policy.
   b. Employee having transportation expenses between 200 to 300 are more absent, company can provide them transportation facility
   c. Employee between age 30-40 are more absent, company need to bring policy to monitor
   d. Employee of 40+ age are less absent
   e. Employee having No Child, Single Child or 2 children are more absent, company can make new policy for those parent(for eg alternate office, or flexible shift timing)
   f. Employee having more than 2 children are less absent

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

Below tables shows that monthly loss percentage we can project for 2011, if same absenteeism trend continues

Month of absence	Predicted_Absenteeism in hours	monthly_loss_percentage
1	40.129662	0.633360
2	68.961575	1.088409
3	154.448857	2.437640
4	73.239864	1.155932
5	45.982372	0.725732
6	100.445174	1.585309
7	140.578734	2.218730
8	57.779103	0.911918
9	74.032353	1.168440
10	78.040713	1.231703
11	114.562468	1.808120
12	67.389058	1.063590

1.2 Data

Total Number of Attributes: 21

Missing Values : Yes

Below are the detail of the independent and dependent variable , we will analyse for solving the problem. Our task will be to analyse the each variable distribution, if required scale them, study the correlation between independent and target variable .

Using graphical representation , we will study the dependency of independent variable with dependent variable to make our conclusion about what factor can be improved to minimise the absenteeism.

After study, analysing and conclusion made from the given data, we have to predict the loss for year 2011, if same trend continues.

Below are independent variable we have :

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

1 Individual identification (ID)

2. Reason for absence (ICD)
   - 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).

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 (target)

Below are first 5 observations of given data:

Table a. (Column 1 to 6)

ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense
11	26	7	3	1	289
36	0	7	3	1	118
3	23	7	4	1	179
7	7	7	5	1	279
11	23	7	5	1	289

Table b. (Column 7 to 12)

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

Table c. (Column 13 to 21)

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

Chapter 2

Methodology

2.1 Pre Processing

Data pre-processing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we pre process our data before feeding it into our model. This includes multiples process described below. At very first step it involve missing value analysis and checking invalid data which can affect our model accuracy. After which It followed by feature selection, features scaling.

We will also check the graphical representation of data in terms of bar graph, histogram, heatmap and other to visualise the data for conclude the characteristics of the data, dependency of the data on target variable.

All above process, on very high level , termed as Exploratory Data Analysis.

2.1.1 Missing Value (KNN Imputation)

Here in data frame, we found missing value , which we converted into percentage for better analysis. Refer to below table. We will find the missing value, analyse them before selecting the right method to impute the missing value.

For imputing the missing value, we have many methods. Out of those majorly we have 3 methods, which are mean method, median method and KNN Imputation. Here in our current project , we found KNN imputation to be more effective. Hence we used KNN Imputation for imputing missing value.

Missing Value Percentage - wise

ID	0.000000
Reason for absence	0.426136
Month of absence	0.142045
Day of the week	0.000000
Seasons	0.000000
Transportation expense	0.994318
Distance from Residence to Work	0.426136
Service time	0.426136
Age	0.426136
Work load Average/day	1.420455
Hit target	0.852273
Disciplinary failure	0.852273
Education	1.420455
Son	0.852273
Social drinker	0.426136
Social smoker	0.568182
Pet	0.284091
Weight	0.142045
Height	1.846591
Body mass index	4.119318
Absenteeism time in hours	3.125000

2.2.2 Invalid values analysis

After missing value imputing, we checked our variable for invalid data, found that, few categorical variable has got invalid value from above imputation. Hence we check and removed those variable. We also changed the required data type for variable.

2.1.3 Feature Selection

Before performing any type of modelling, 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.

In this process, we first check the distribution of variables. Refer to Fig 1.1. From there we concluded mainly 3 points which are :

- Age<40 are more in numbers, similarly zero disciplinary failure are less.
- Transportation expenses is ranging majorly from 180-300.
- Non Smoker are less as compared to smoker. Similarly Drinkers are more.

After that we studied the heatmap(Fig 1.2) of variable for analysing the correlation of the variable. From where we concluded that Body mass index and weight are highly corelated, hence we drop Body mass index. Similarly we found Age and Service time are also corelated.

Next by Fig 1.3, we analysed the correlation between continuous variable with Target variable.

From Fig 1.4, we study the effect 'Age' variable on Absenteeism hour of employee, which gives us Employee between 30 to 40 years of age are more absent. Employee with age more than 40 years are less absent.

From Fig 1.5, we analysed that effect of transportation expense on absenteeism, where we found those having expenses between 200-300 are more absent

From Fig 1.6, we check the work distance dependency on absenteeism, and found employee having work distance between 10KM to 30 KM are more absent.

From Fig 1.7, we check no. of child dependency on absenteeism, and we found that employee having more than 2 children are less absent, whereas employee having below 2 Children are more absent

2.1.4 Feature Scaling

Feature scaling in machine learning is one of the most critical steps during the pre-processing of data before creating a machine learning model. Scaling can make a difference between a weak machine learning model and a better one.

The most common techniques of feature scaling are Normalization and Standardization.

Here after all dependency analysis, we will scale our data to a proper range. Here our variable like ["Transportation expense","Distance from Residence to Work","Age","Work load Average/day ","Hit target","Weight","Height"] are having higher range value, comparing to other variable which can affect our model performance. Hence we will perform standardisation for scaling the data, to make them in close range with respect to other variables.

2.1.5 Principle Component Analysis (PCA)

Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

Here in our case, we found that 9 components are retaining the 95% of variance of independent variable. Hence we chosen 9 component for dimensionality reduction.

2.2 Modeling

2.2.1 Model Selection

In our early stages of analysis we found that our target variable is continuous variable, hence we will adopt regression model for making our final model. We will opt for Linear Regression and Random forest regression, and compare their root mean square error to conclude the best model.

2.2.2 Linear Regression

Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable.

```
#Using linear regression (in python)

lrm_reg = LinearRegression()

lrm_reg.fit(X_train, y_train)

Y_predict_lrm =lrm_reg.predict(X_test)
```

2.2.3 Random Forest Regression

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees.

```
#using random forest Regressor (in python)

rf_model = RandomForestRegressor()

rf_model.fit(X_train_scaled_pca, y_train)

y_pred_pca = rf_model.predict(X_test_scaled_pca)
```

Chapter 3

Conclusion

3.1 Model Evaluation

For model evaluation in terms of accuracy and for the analytical factors we have certain ways. Out of which the confusion matrix is used to evaluate a classification problem's accuracy. On the other hand, mean squared error (MSE), mean absolute error (MAE) and root mean square error (RMSE) are used to evaluate the regression problem's accuracy. We have used RMSE to evaluate our model

3.1.1 Root Mean Square Error

RMSE is the standard deviation of the errors which occur when a prediction is made on a dataset. This is the same as MSE (Mean Squared Error) but the root of the value is considered while determining the accuracy of the model.

In RMSE, the errors are squared before they are averaged. This basically implies that RMSE assigns a higher weight to larger errors. This indicates that RMSE is much more useful when large errors are present and they drastically affect the model's performance. In this metric also, lower the value, better is the performance of the model.

3.2 Model Selection

We can see that both models perform comparatively on average and therefore we can select either of the two models without any loss of information. But if we deeply analyse, the RMSE for linear regression model is better than the random forest .

```
RMSE for Linear Regression 0.0141929179459719
RMSE for Random Forest Regression 0.015114174246574805
```

Hence we will consider the Linear Regression model.

3.3 Predicting loss for year 2011

For predicting the loss of year 2011, we will add age and service time of employee by 1. Keeping the increase value in our data frame, we will predict the absenteeism from our above selected Linear Regression model.

Once predicted we will sum the absenteeism for each month, as below table :

Month of absence	Predicted_Absenteeism in hours
1	40.129662
2	68.961575
3	154.448857
4	73.239864
5	45.982372
6	100.445174
7	140.578734
8	57.779103
9	74.032353
10	78.040713
11	114.562468
12	67.389058

Now let say in a month excluding weekend 22 days are working days. total working hours of 36 employees will be 22 x 8 x 36.

And total losses percentage = (absent_hours / Total_Hours)*100

Total_Monthly_hours = 22 x 8 x 36

'monthly_loss_percentage'= (Predicted_Absenteeism in hours / total_Monthly_hours) x 100

Which gives below loss detail. Below tables shows the loss percentages.

Month of absence	Predicted_Absenteeism in hours	monthly_loss_percentage
1	40.129662	0.633360
2	68.961575	1.088409
3	154.448857	2.437640
4	73.239864	1.155932
5	45.982372	0.725732
6	100.445174	1.585309
7	140.578734	2.218730
8	57.779103	0.911918
9	74.032353	1.168440
10	78.040713	1.231703
11	114.562468	1.808120
12	67.389058	1.063590

Appendix A - Extra Figures

Fig 1.1 Graph representing the distribution of each variable:

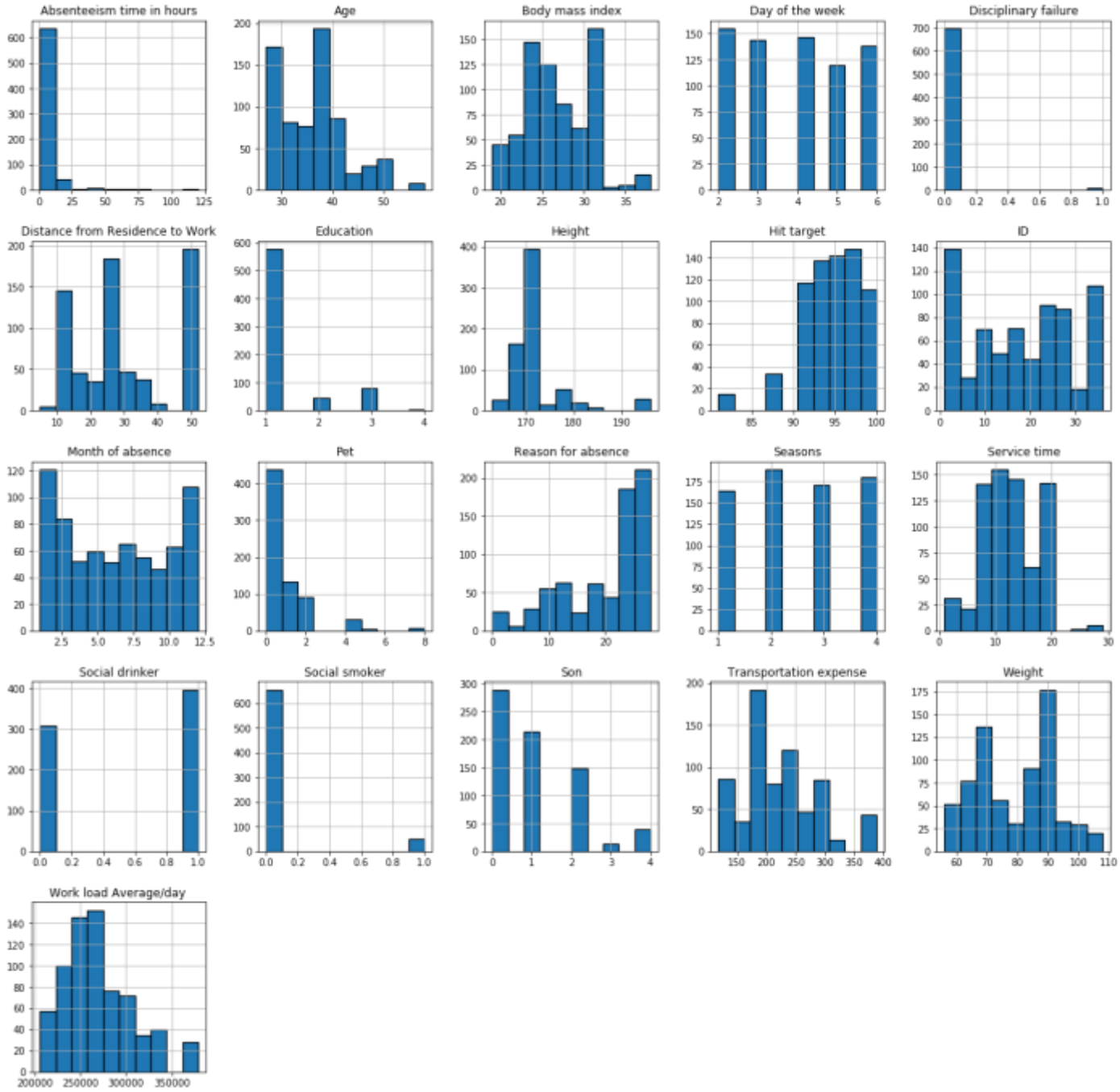


Fig 1.2 Heatmap of variables for studying the correlation

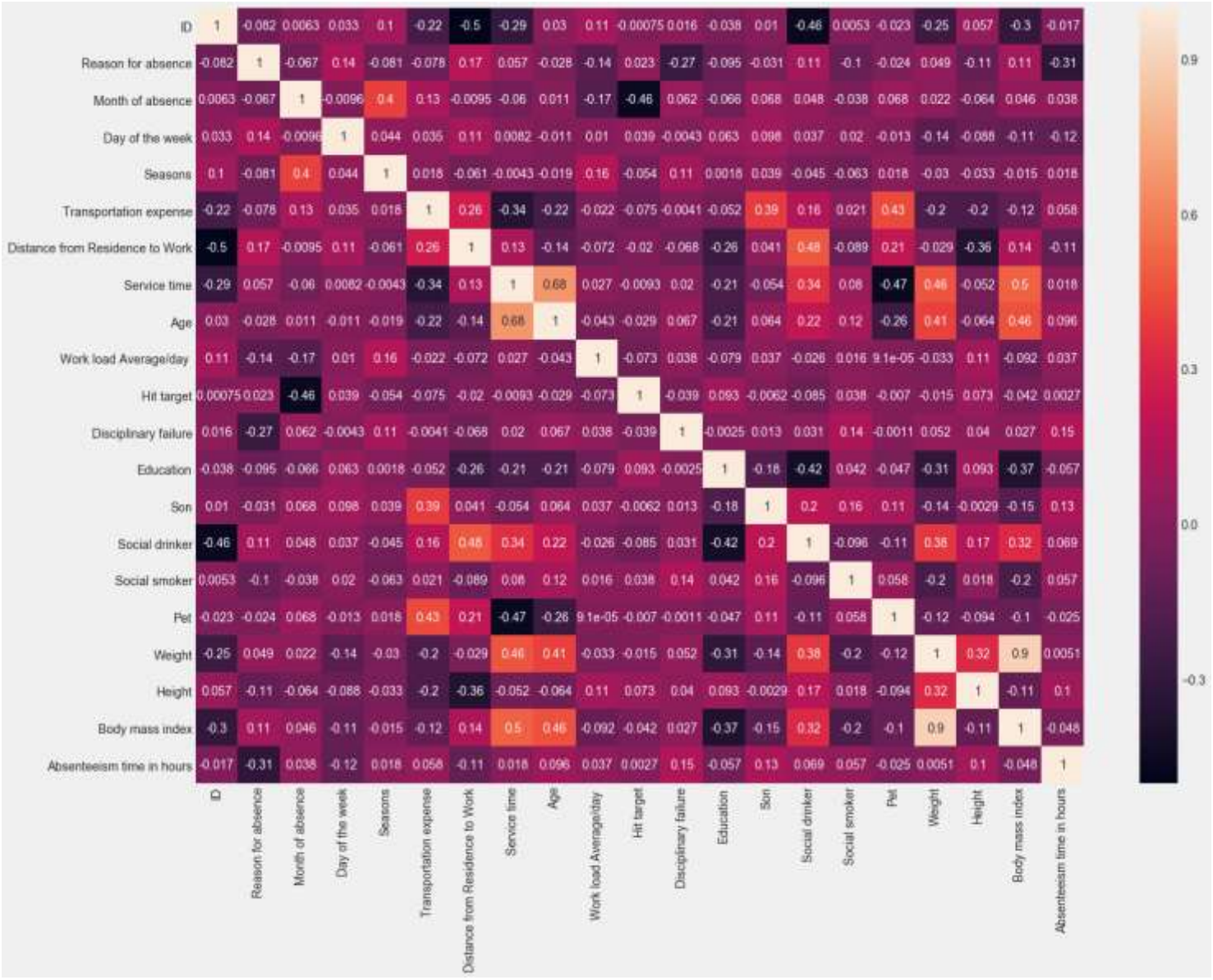


Fig 1.3 Correlation of independent variable with dependent variable

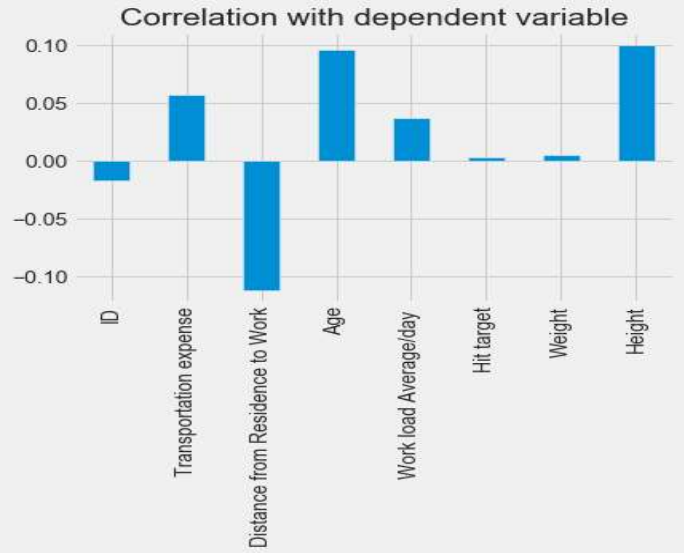


Fig 1.4 Effect of 'Age' on 'Absenteeism

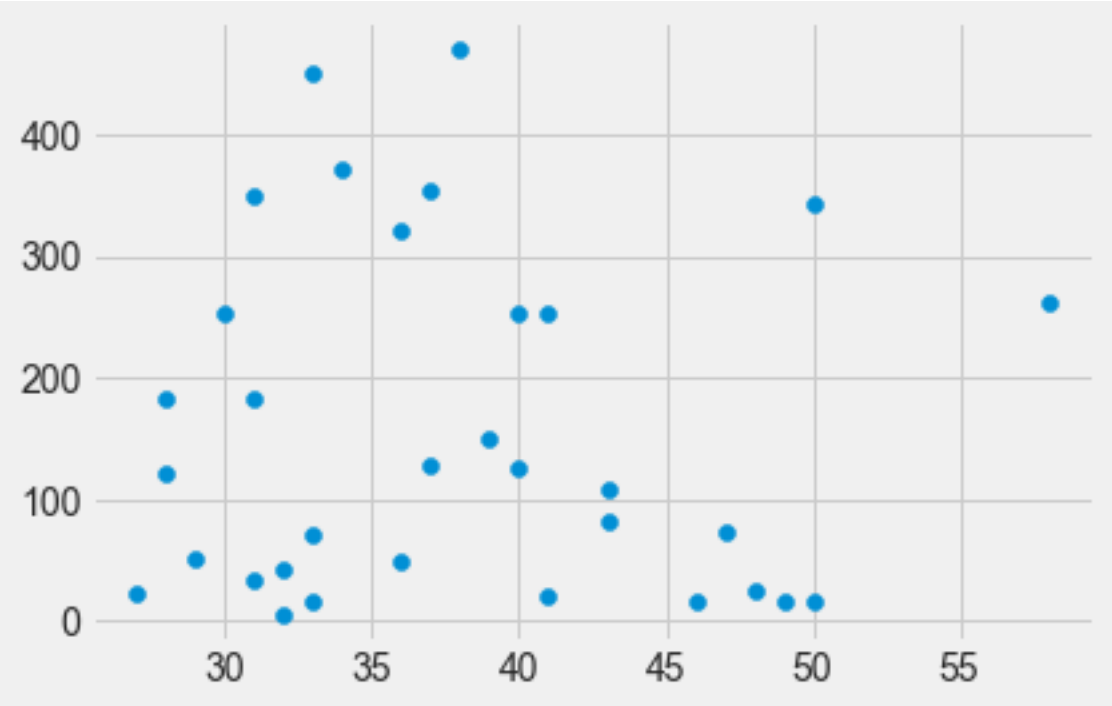


Fig 1.5 Effect of 'Transportation_Expense' on Total Absenteeism hours

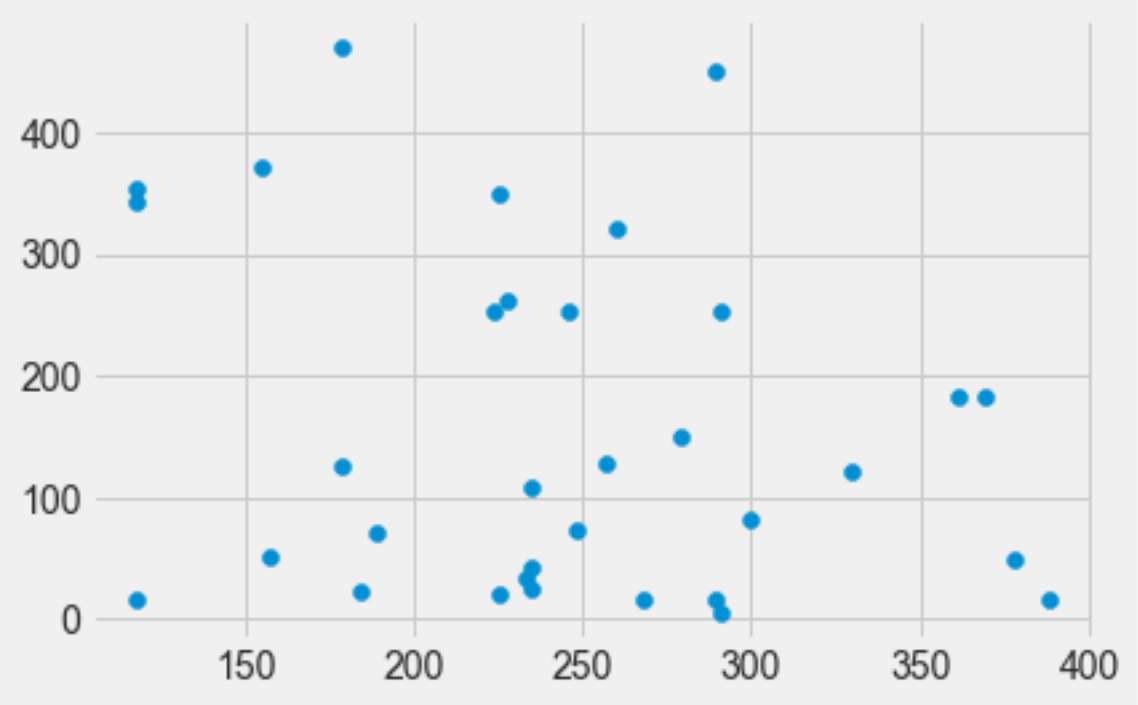


Fig 1.6 Effect of 'Work_Distance' and by total hours of absence

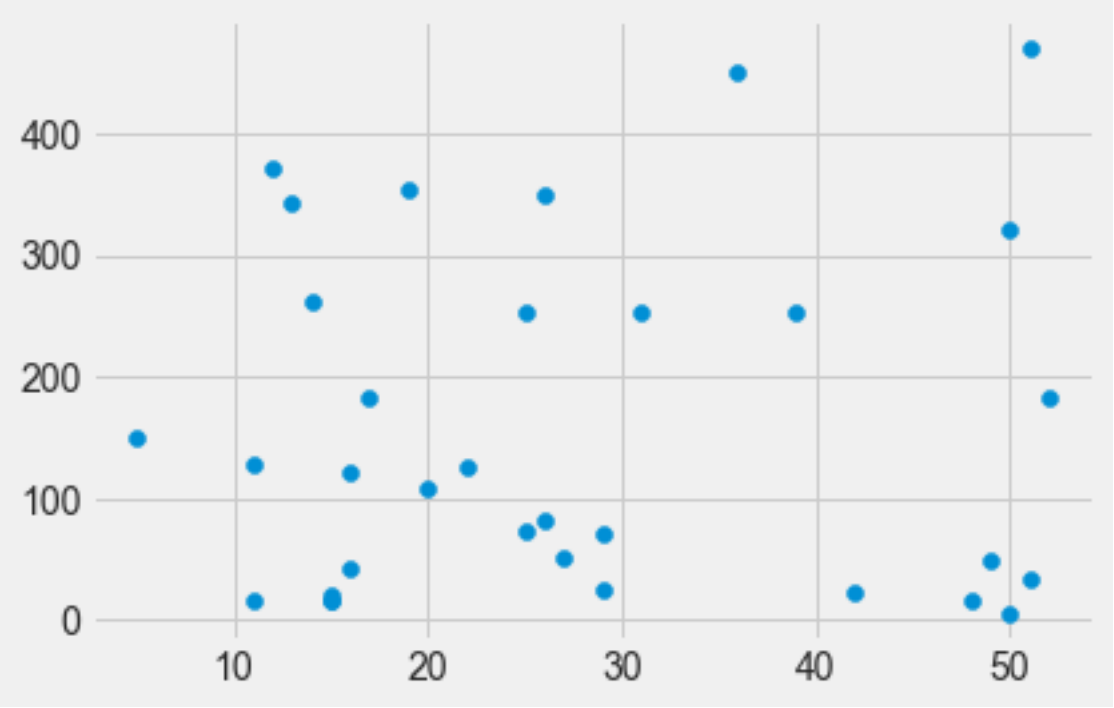
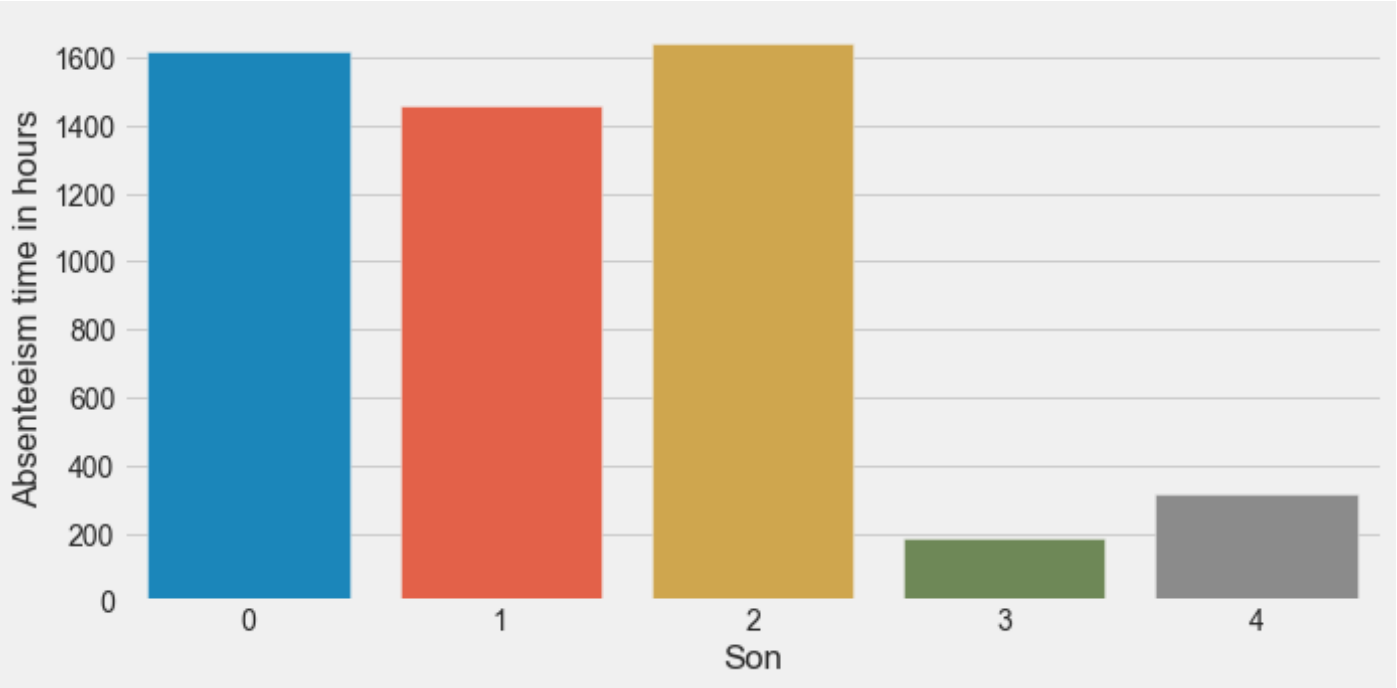


Fig 1.7 Absence dependency on number of kids



Appendix B – Code

Python Code :

```
import pandas as pd
import os
import numpy as np
import math

#------ for model evaluation -----
from sklearn.metrics import mean_squared_error

#----- for preprocessing
from sklearn.preprocessing import Normalizer
from sklearn.preprocessing import StandardScaler

#---- for model building
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor

#---- for visualization---
import matplotlib.pyplot as plt
import seaborn as sns

#----Train Test Split
from sklearn.model_selection import train_test_split
```

In [2]:

```
#Setting default location
os.chdir("C:/Users/SAURABH SHRIVASTAVA/OneDrive/Data Scientist/Edwisor project")
os.getcwd()
```

Out[2]:

```
'C:\\Users\\SAURABH SHRIVASTAVA\\OneDrive\\Data Scientist\\Edwisor project'
```

In [3]:

```
#Loading Data
emp = pd.read_excel("Absenteeism_at_work_Project final.xls")
print(emp.shape)
emp.head(3)
```

```
(740, 21)
```

Out[3]:

In [4]:

```
#Checking invalid data
#Absenteeism Hour contains zero, so we will drop that
emp=emp[emp["Absenteeism time in hours"]!=0]

#check for missing value in percentage
(emp.isnull().sum()/emp.shape[0])*100
```

Out[4]:

```
ID                                0.000000
Reason for absence                0.426136
Month of absence                  0.142045
Day of the week                   0.000000
Seasons                           0.000000
Transportation expense            0.994318
Distance from Residence to Work   0.426136
```

```
Service time                      0.426136
Age                               0.426136
Work load Average/day             1.420455
Hit target                        0.852273
Disciplinary failure              0.852273
Education                         1.420455
Son                               0.852273
Social drinker                    0.426136
Social smoker                     0.568182
Pet                               0.284091
Weight                            0.142045
Height                            1.846591
Body mass index                   4.119318
Absenteeism time in hours         3.125000
dtype: float64
```

In [5]:

```
#Imputing Missing Value

#Applying KNN Imputation for missing value impute

from sklearn.impute import KNNImputer
imputer=KNNImputer()
emp = pd.DataFrame(imputer.fit_transform(emp), columns=emp.columns)

#Re-check for missing value in percentage
(emp.isnull().sum()/emp.shape[0])*100
```

Out[5]:

```
ID                                0.0
Reason for absence                0.0
Month of absence                  0.0
Day of the week                   0.0
Seasons                           0.0
Transportation expense            0.0
Distance from Residence to Work   0.0
Service time                      0.0
Age                               0.0
Work load Average/day             0.0
Hit target                        0.0
Disciplinary failure              0.0
Education                         0.0
Son                               0.0
Social drinker                    0.0
Social smoker                     0.0
Pet                               0.0
Weight                            0.0
Height                            0.0
Body mass index                   0.0
Absenteeism time in hours         0.0
dtype: float64
```

In [6]:

```
#Rechecking of invalid values
emp["Month of absence"].value_counts()
```

Out[6]:

```
3.0     84
2.0     72
7.0     65
10.0    63
5.0     59
11.0    59
8.0     54
4.0     52
6.0     51
1.0     49
12.0    49
9.0     46
8.2      1
Name: Month of absence, dtype: int64
```

In [7]:

```
#Rechecking of invalid values
emp["Pet"].value_counts()
```

Out[7]:

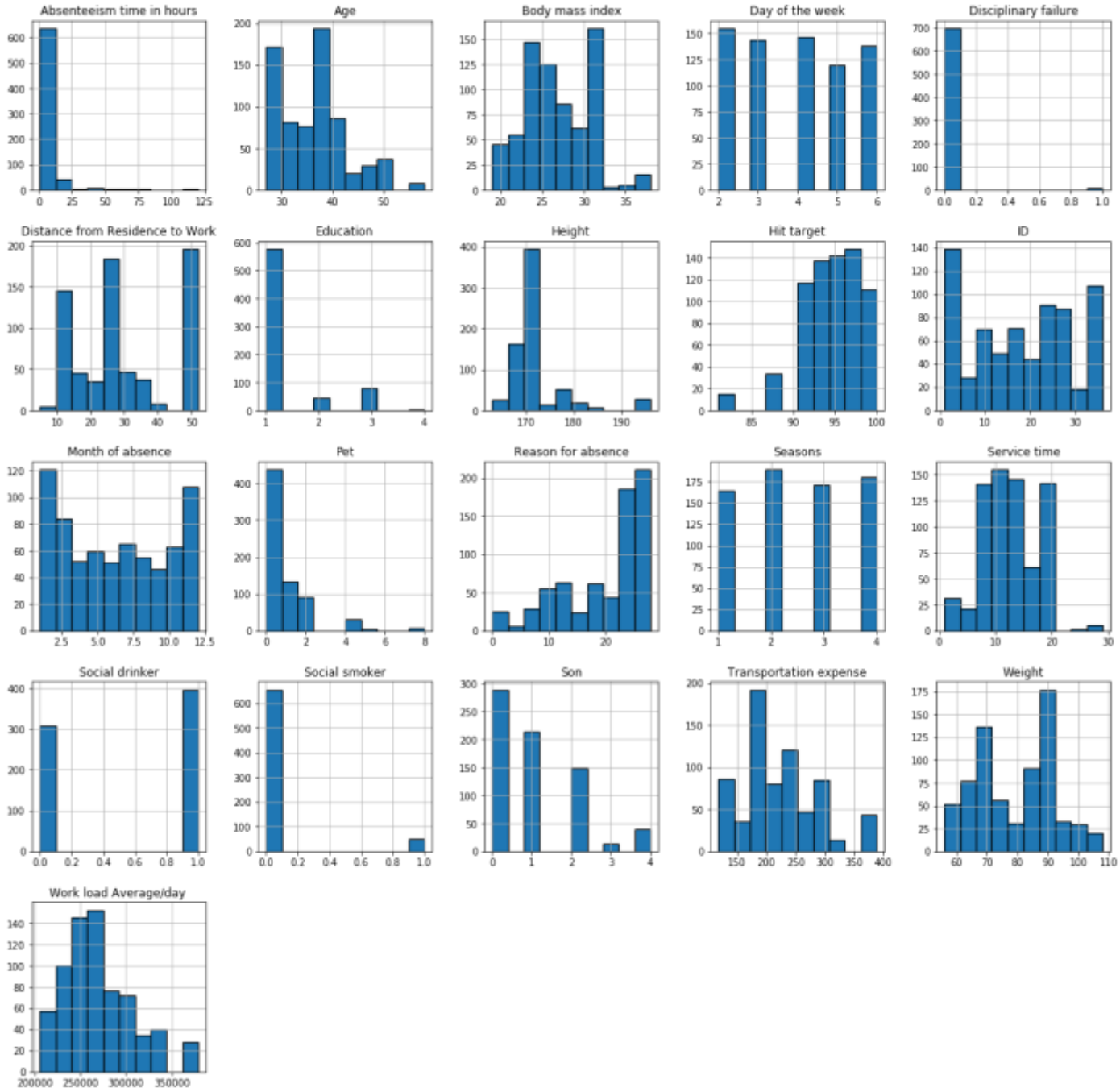
```
0.0    438
1.0    131
2.0     92
4.0     30
8.0      7
5.0      4
0.8      2
Name: Pet, dtype: int64
```

In [8]:

```
#After imputation , we found we found many independent variable has got invalid value,
#hence we will round of those value to make it valid

#Rounding up variables and converting into int format
for j in emp.columns:
    for i in range(len(emp[j])):
        emp[j].loc[i]=(round(emp[j].loc[i]))

emp=emp.astype(int) #changing float to int
```

In [9]:

```
#############Feature Selection#############################################################
#########

#Analysing each variables for its distribution, for our understanding of data how it is
distributed on total.
emp.hist(edgecolor='black', linewidth=1.2, figsize=(20, 20));
```

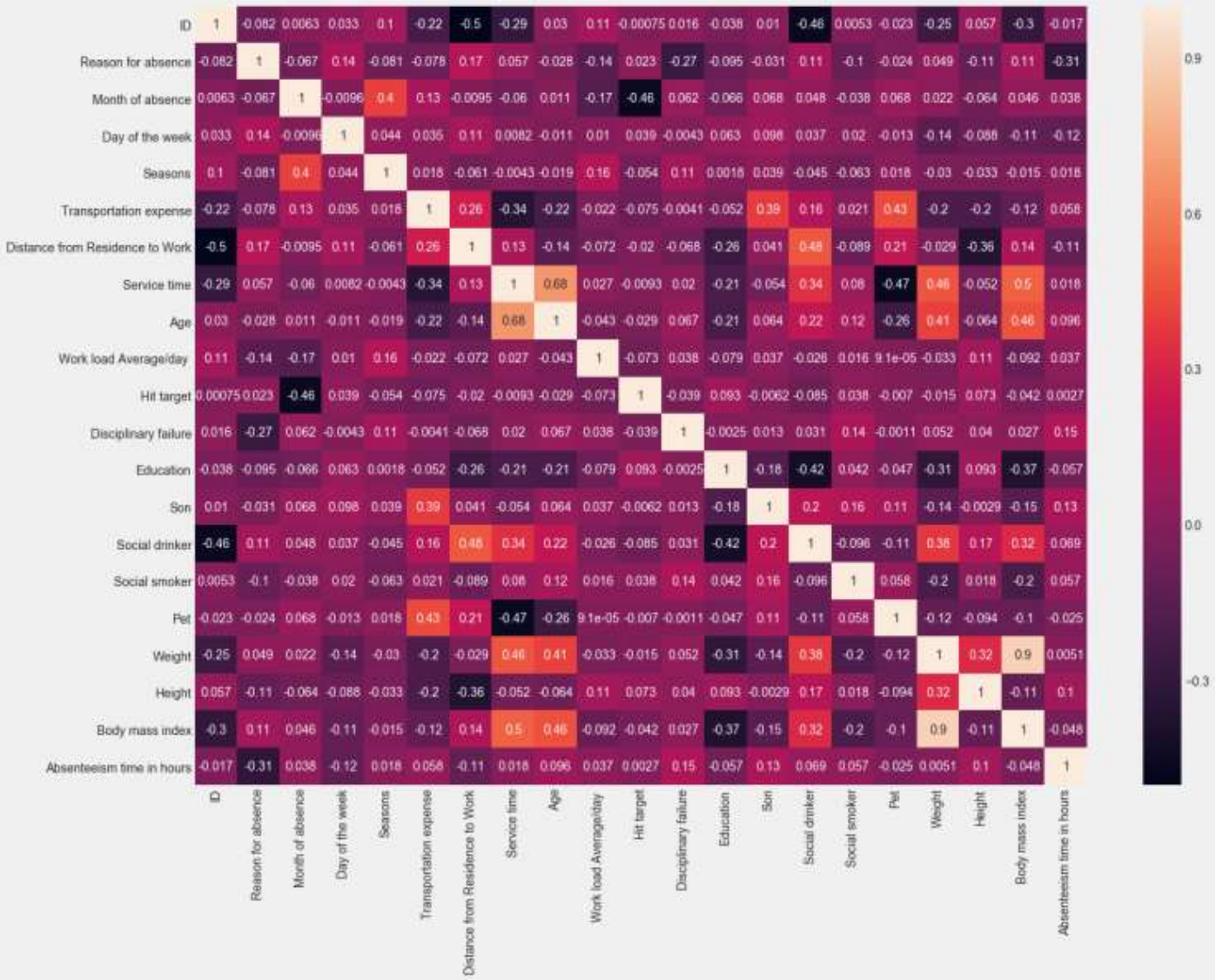


In [10]:

```
#As from above graph we analysed that Age<40 are more in numbers, similarly zero discip
linary failure are less.
#Transportation expenses is ranging majorly from 180-300.
#Non Smoker are less as compared to smoker. Similarly Drinkers are more.

#Analysing Correlation between different variables from heat map
import seaborn as sns
%matplotlib inline
sns.set_style("whitegrid")
plt.style.use("fivethirtyeight")
plt.figure(figsize=(20, 15))
sns.heatmap(emp.corr(), annot=True)
```

Out[10]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x20b4e621c18>
```



In [11]:

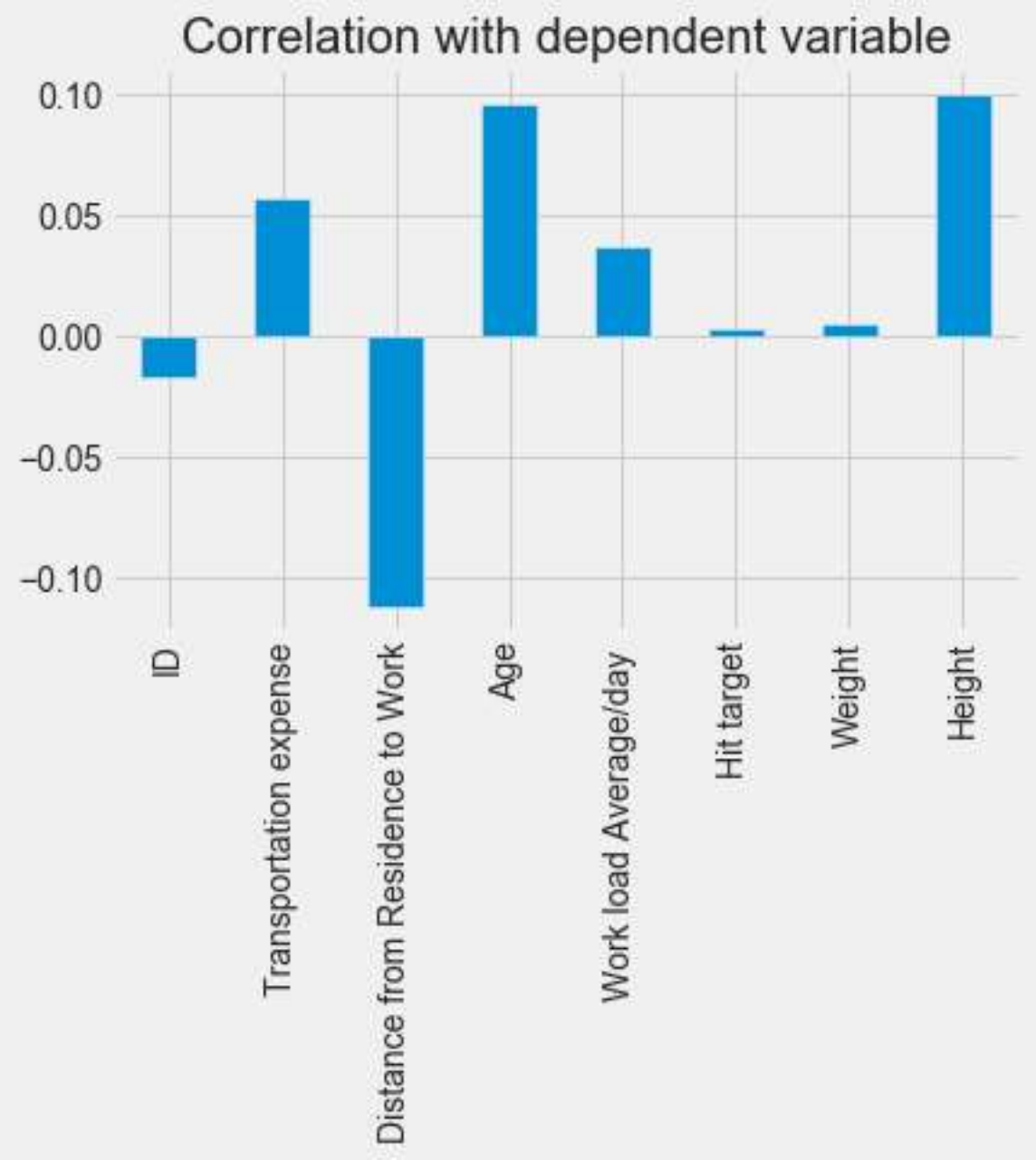
```
#From above heat map, we can see, weight and body mass index are highly corelated,
#similary Service time and Age is also corelated . So we will drop below column

emp=emp.drop(['Body mass index'], axis=1)

#Segregating variable into Categorical and continous
categorical_var = ['Reason for absence','Month of absence','Day of the week','Seasons',
'Disciplinary failure','Education','Son','Social drinker','Social smoker','Pet']
continous_var = ['ID','Transportation expense','Distance from Residence to Work','Age',
'Work load Average/day ','Hit target','Weight','Height']
```

In [12]:

```
##Explore the correlation btwn the independent continous features with target variabe
corr=emp[continous_var].corrwith(emp["Absenteeism time in hours"])
corr.plot.bar(figsize=(6,4), title='Correlation with dependent variable', grid=True, le
gend=False, style=None, fontsize=None, colormap=None, label=None)
```

Out[12]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x20b4f01d390>
```

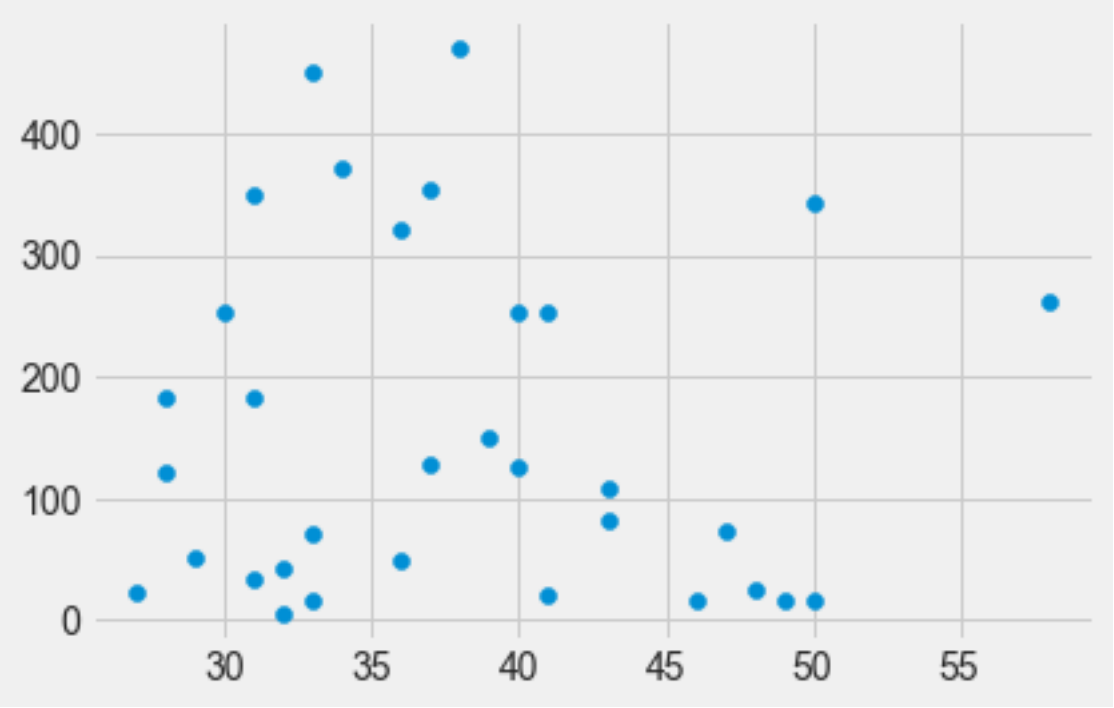


In [13]:

```
############Checking the effect of 'Age' on 'Absence'
#Analysing 'Age' with respect to total hours of absence
emp_hours = emp[['ID','Absenteeism time in hours']].groupby('ID').sum().reset_index()
emp_age = emp[['ID','Age']].groupby('ID').max().reset_index()
absence_by_age = emp_hours.merge(emp_age, how='inner',left_on='ID', right_on='ID')

plt.scatter('Age', 'Absenteeism time in hours', data=absence_by_age)
```

Out[13]:

```
<matplotlib.collections.PathCollection at 0x20b4db95198>
```

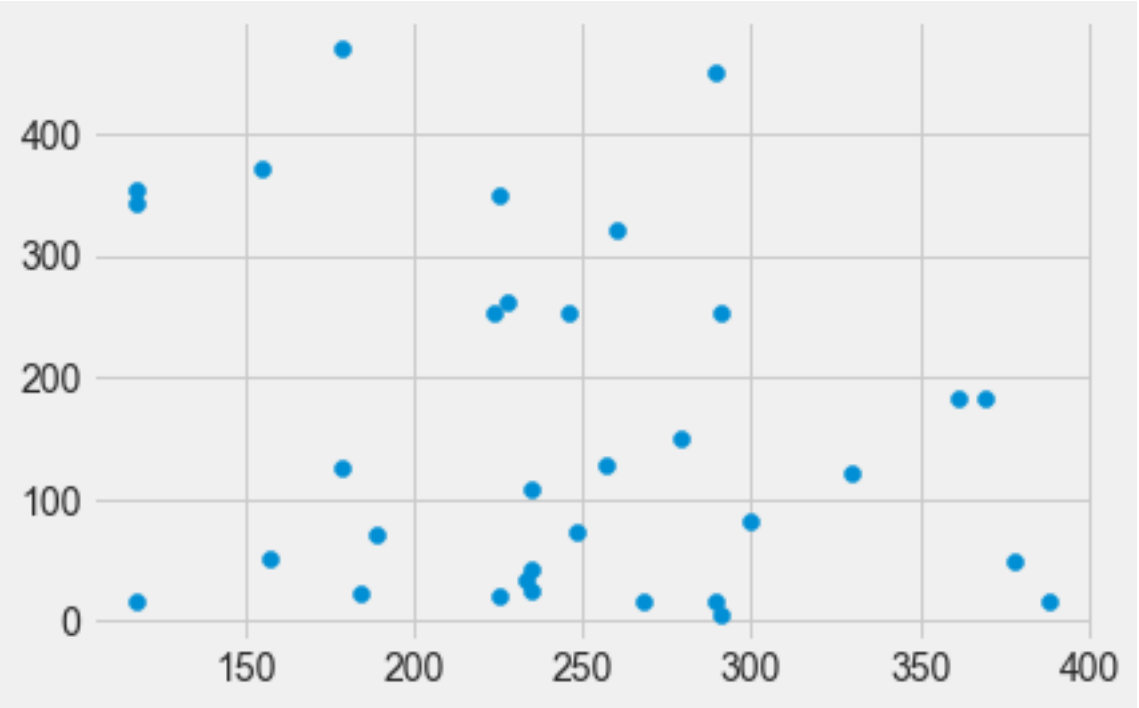


In [14]:

```
########From above we conclude age 30-40 are more absent and 40+ age are less absent

#Checking the effect of 'Transportation_Expense' on Total Absenteeism hours

emp_transport = emp[['ID','Transportation expense']].groupby('ID').max().reset_index()
absence_by_transport = emp_hours.merge(emp_transport, how='inner',left_on='ID', right_o
n='ID')
plt.scatter('Transportation expense', 'Absenteeism time in hours', data=absence_by_tran
sport)
```

Out[14]:

```
<matplotlib.collections.PathCollection at 0x20b4e5c1400>
```



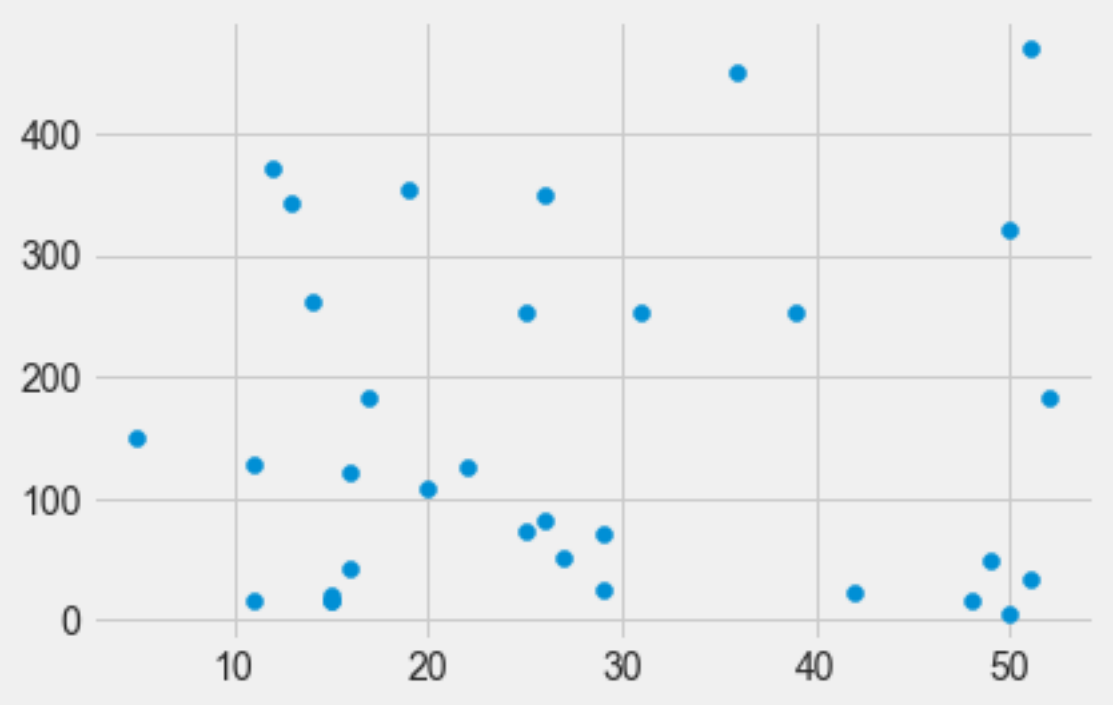
In [15]:

```
##########Above concludes that expenses between 200 to 300 are more absenteeism hours

#checking the effect of 'Work_Distance' and by total hours of absence
```

```
emp_distance = emp[['ID','Distance from Residence to Work']].groupby('ID').max().reset_
index()
absence_by_distance = emp_hours.merge(emp_distance, how='inner',left_on='ID', right_on=
'ID')
plt.scatter('Distance from Residence to Work', 'Absenteeism time in hours', data=absenc
e_by_distance)
```

Out[15]:

```
<matplotlib.collections.PathCollection at 0x20b4dc10748>
```

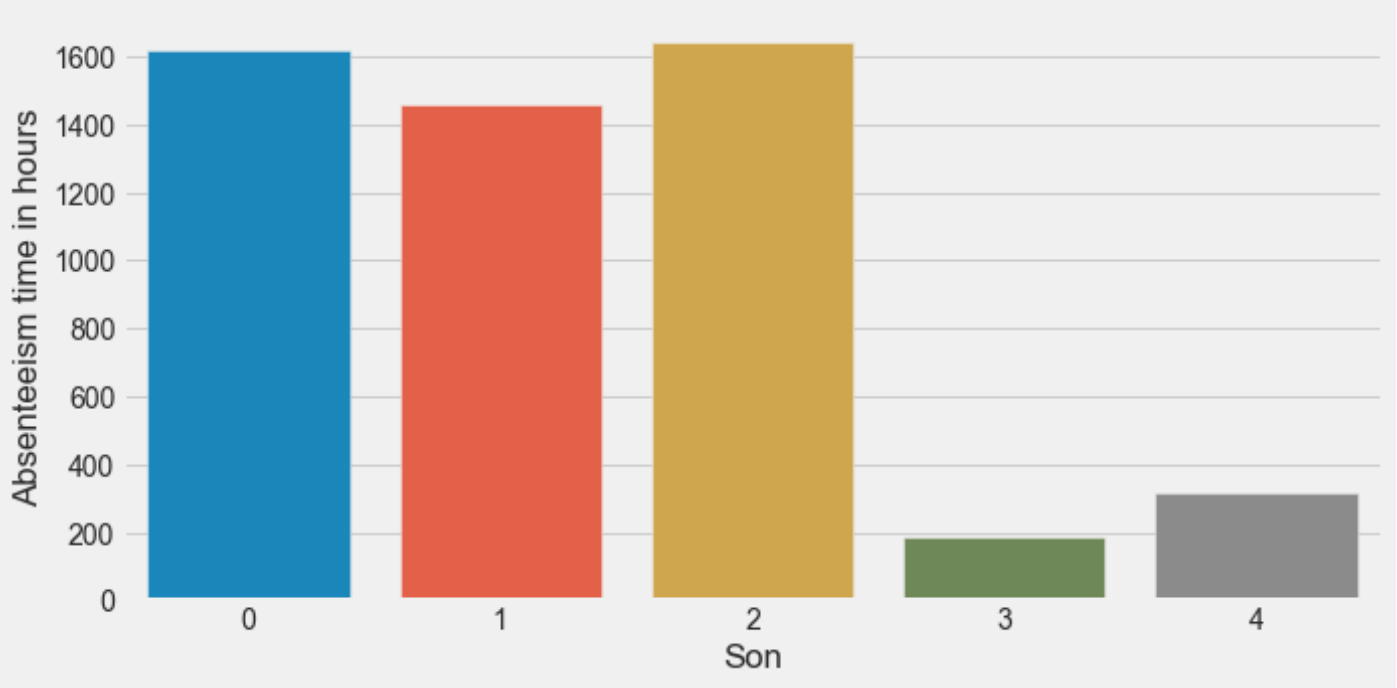


In [16]:

```
#########3#above shows that employee having distance between 10-30 are more absent

#Analyzing absence dependency of no of kids
emp_son_tot = emp[['Son','Absenteeism time in hours']].groupby('Son').sum().sort_values
('Absenteeism time in hours').reset_index()
fig,ax = plt.subplots(nrows=1,ncols=1)
fig.set_size_inches(10,5)
sns.barplot(x='Son', y='Absenteeism time in hours', hue=None, data=emp_son_tot, order=N
one, hue_order=None, units=None, orient=None, color=None, palette=None,errcolor='.26',
ax=ax)
```

Out[16]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x20b4dc02438>
```



In [17]:

```
######Above shows that employee having 0-2 childs are more absent and those having more
than 2 childs are less absent
```

In [18]:

```
#Train Test Split using sklear library and scaling dependent variable
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(emp.iloc[:,0:19], emp.iloc[:,19], t
est_size=0.2, random_state=20)
y_train=y_train/1000
y_test=y_test/1000
```

In [19]:

```
contCol = ["Transportation expense","Distance from Residence to Work","Age","Work load
Average/day ","Hit target","Weight","Height"]

# Feature Scaling
from sklearn.preprocessing import StandardScaler

X_train_scaled = X_train
X_test_scaled = X_test

# apply standardization on numerical features
for i in contCol:

    # fit on training data column
    sc = StandardScaler().fit(X_train_scaled[[i]])

    # transform the training data column
    X_train_scaled[i] = sc.transform(X_train_scaled[[i]])

    # transform the testing data column
    X_test_scaled[i] = sc.transform(X_test_scaled[[i]])

X_train_scaled.head()
```

Out[19]:

In [20]:

```
### Applying PCA for Dimensionality reduction,

from sklearn.decomposition import PCA
pca = PCA(n_components=9)
pca.fit(X_train_scaled)
X_train_scaled_pca = pca.transform(X_train_scaled)
X_test_scaled_pca = pca.transform(X_test_scaled)
explained_variance = pca.explained_variance_ratio_
explained_variance
```

Out[20]:

```
array([0.56207107, 0.24431879, 0.08494647, 0.05565959, 0.0104569 ,
       0.00774492, 0.00611165, 0.00558609, 0.00519288])
```

In [21]:

```
#Now time for making our model and evaluating the model, since target variable is conti
nous, we will use Regression model

#using random forest Regressor
rf_model = RandomForestRegressor()
rf_model.fit(X_train_scaled_pca, y_train)
y_pred_pca = rf_model.predict(X_test_scaled_pca)

#Using linear regression
lrm_reg = LinearRegression()
lrm_reg.fit(X_train, y_train)
Y_predict_lrm =lrm_reg.predict(X_test)
```

In [ ]:

In [22]:

```
#finding Root Mean Square Error : RMSE for both model
rmse_rForest = math.sqrt(mean_squared_error(y_test, y_pred_pca))
rmse_linear = math.sqrt(mean_squared_error(y_test, Y_predict_lrm))


print("RMSE for Linear Regression {}".format(rmse_linear))
print("RMSE for Random Forest Regression {}".format(rmse_rForest))
```

```
RMSE for Linear Regression 0.0141929179459719
RMSE for Random Forest Regression 0.015114174246574805
```

In [23]:

```
#For 2011 prediction
#service and age will be added by 1

emp_pred_2011 = emp
emp_pred_2011["Service time"] = emp["Service time"] + 1
emp_pred_2011["Age"] = emp["Age"] + 1
emp_pred_2011.head()
```

Out[23]:

In [24]:

```
#emp_pred_2011= emp_pred_2011.drop(columns = ['Body mass index'], axis=1)
```

In [25]:

```
emp_pred_2011.shape
```

Out[25]:

```
(704, 20)
```

In [26]:

```
#-------- Standardise the scale, before passing the input to model
contCol = ["Transportation expense","Distance from Residence to Work","Age","Work load
Average/day ","Hit target","Weight","Height"]

#from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(emp_pred_2011.iloc[:,0:19], emp_pre
d_2011.iloc[:,19], test_size=0.2, random_state=20)
y_train=y_train/1000
y_test=y_test/1000

# Feature Scaling
from sklearn.preprocessing import StandardScaler

X_train_scaled = X_train
X_test_scaled = X_test


# apply standardization on numerical features
for i in contCol:

    # fit on training data column
    sc = StandardScaler().fit(X_train_scaled[[i]])

    # transform the training data column
    X_train_scaled[i] = sc.transform(X_train_scaled[[i]])

    # transform the testing data column
    X_test_scaled[i] = sc.transform(X_test_scaled[[i]])
#X_test = sc.transform(X_test)

#emp.describe()
##########TRY USING SCALED AND UNSCALED DATA for MODELLING OR BY CHANGING SCALLING COL
UMN
X_train_scaled.head()
```

Out[26]:

In [27]:

```
#predicting using linear regression from above model
predict_2011_absent =lrm_reg.predict(X_test)
absence_predict_2011=X_test
absence_predict_2011['Predicted_Absenteeism in hours'] = predict_2011_absent*1000
absence_predict_2011.head()
```

Out[27]:

In [28]:

```
#adding total absenteeism with respect to month of absent
monthly_absence= absence_predict_2011.groupby('Month of absence').sum().reset_index()[[
'Month of absence','Predicted_Absenteeism in hours']]
monthly_absence
```

Out[28]:

	Month of absence	Predicted_Absenteeism in hours
0	1	40.129662
1	2	68.961575
2	3	154.448857
3	4	73.239864
4	5	45.982372
5	6	100.445174
6	7	140.578734
7	8	57.779103
8	9	74.032353
9	10	78.040713
10	11	114.562468
11	12	67.389058

In [29]:

```
#lets say in a month excluding weekend 22 days are working days. total working hours of
36 employees will be 22*8*36.
# total losses % = (absent_hours / Total_Hours)*100
tot_Monthly_hours = 22*8*36
monthly_absence['monthly_loss_percentage'] = (monthly_absence['Predicted_Absenteeism in
hours']/tot_Monthly_hours) * 100
```

In [30]:

```
#Loss Predicted For 2011/Month -----------
monthly_absence
```

Out[30]:

	Month of absence	Predicted_Absenteeism in hours	monthly_loss_percentage
0	1	40.129662	0.633360
1	2	68.961575	1.088409
2	3	154.448857	2.437640

	Month of absence	Predicted_Absenteeism in hours	monthly_loss_percentage
3	4	73.239864	1.155932
4	5	45.982372	0.725732
5	6	100.445174	1.585309
6	7	140.578734	2.218730
7	8	57.779103	0.911918
8	9	74.032353	1.168440
9	10	78.040713	1.231703
10	11	114.562468	1.808120
11	12	67.389058	1.063590

R Code:

```
library("readxl")
library("DMwR")
library("ggplot2")
library("randomForest")
library("corrgram")
library("rpart")
library("plyr")
library("caret")
library("vctrs")
library(CSTools)
library(Metrics)
library(e1071)



# Set working directory
setwd("C:/Users/SAURABH SHRIVASTAVA/OneDrive/Data Scientist/Edwisor project")

#loading data
emp = read_excel('Absenteeism_at_work_Project final.xls',sheet = 1,col_names = TRUE)

emp = as.data.frame(emp)
head(emp)

#getting all column names from dataframe

col_names = colnames(emp)

#separating categorical variable names in a variable
col_categ = c('Month of absence', 'Day of the week',
        'Seasons', 'Disciplinary failure', 'Education', 'Social drinker',
        'Social smoker')
```

```
############Missing value check################

#checking whether missing value present
table(is.na(emp))

###########Imputing missing value from KNN ########

emp_knn=knnImputation(emp,k = 3)
emp_knn=as.data.frame(emp_knn)

############Converting required datatype################

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

for (i in col_categ){
  emp_knn[,i]=as.factor(emp_knn[,i])
}

###################Feature Selection###################
corrgram(emp_knn[,],order = FALSE,upper.panel = panel.pie,text.panel = panel.txt,main= 'Correlation Plot')

#from corelation plot we understood weight and body mass index is highly corelated,
#hence we drop any one of this. Similarly we will do for Service time and age
#emp_knn = subset(emp_knn,select = -c(Weight))

dim(emp_knn)
emp_knn = subset(emp_knn,select = -c(`Body mass index`, `Service time`))

###################Renaming Column###################
names(emp_knn)[names(emp_knn) == "Reason for absence"] <- "Reason_for_absence"
names(emp_knn)[names(emp_knn) == "Month of absence"] <- "Month_of_absence"
```

```
names(emp_knn)[names(emp_knn) == "Day of the week"] <- "Day_of_the_week"
names(emp_knn)[names(emp_knn) == "Transportation expense"] <- "Transportation_expense"
names(emp_knn)[names(emp_knn) == "Distance from Residence to Work"] <- "Work_Distance"
names(emp_knn)[names(emp_knn) == "Work load Average/day"] <- "Average_Workload_per_day"
names(emp_knn)[names(emp_knn) == "Hit target"] <- "Hit_target"
names(emp_knn)[names(emp_knn) == "Disciplinary failure"] <- "Disciplinary_failure"
names(emp_knn)[names(emp_knn) == "Social drinker"] <- "Social_drinker"
names(emp_knn)[names(emp_knn) == "Social smoker"] <- "Social_smoker"
names(emp_knn)[names(emp_knn) == "Absenteeism time in hours"] <- "Absenteeism_time_in_hours"

####################Featrue Scaling###################

scaling_col = c("Transportation_expense", "Work_Distance", "Average_Workload_per_day" ,"Hit_target")
for (i in scaling_col){
  emp_knn[,i]= (emp_knn[,i]-min(emp_knn[,i]))/(max(emp_knn[,i])-min(emp_knn[,i]))
}

##########Checking normality for our target variable#######
#hist(emp_knn$`Absenteeism_time_in_hours`)

#################Train test split###################
#set.seed(1)
#train_index = sample(1:nrow(emp_knn), 0.9*nrow(emp_knn))
#train = emp_knn[train_index,]
#test = emp_knn[-train_index,]

train_index=createDataPartition(emp_knn$`Absenteeism_time_in_hours`,p = 0.9,list = FALSE)
train=emp_knn[train_index,]
test=emp_knn[-train_index,]
train$`Absenteeism_time_in_hours`=as.factor(train$`Absenteeism_time_in_hours`)
test$`Absenteeism_time_in_hours`=as.factor(test$`Absenteeism_time_in_hours`)
#####################################################

###################Linear Regession###############
```

```
#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[,1:19])

linear_rmse = rmse(as.numeric(test$`Absenteeism_time_in_hours`),as.numeric(predict_linreg))

#################Random Forest ###################

rf_model = randomForest(`Absenteeism_time_in_hours`~., data=train, ntree=1000)

predict_rf = predict(rf_model,test[,-19])

rForest_rmse = RMSE(pred = as.numeric(predict_rf), obs = as.numeric(test$`Absenteeism_time_in_hours`))

###############comparing RMSE for model evaluation######
print("RMSE for Linear Regression is :")
linear_rmse

print("RMSE for random forest is :")
rForest_rmse
```