Final_Project_Notebook

December 5, 2018

Data Analaysis

1 Preparing and importing the mandatory librareis for data analaysis

The researcher has performed the follwing tasks the prepare to data for further analysis. * Exploring : conduct a prelimunary analasys and understand the nature of the data * Pre-processing: cleaning ,integrate , and packaging

1.1 Importing the Necessary Libraries

```
In [1]: import pandas as pd
        from sklearn.tree import DecisionTreeRegressor
        from sklearn.linear_model import LinearRegression
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import mean_squared_error
        from math import sqrt
        import matplotlib.pyplot as plt
        import numpy as np
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.metrics import accuracy_score
```

1.2 Data Engineering

1.2.1 Acquire The Dataset :Identify , Retrieve and Query Data

- Identify database was created with records of absenteeism at work from July 2007 to July 2010 at a courier company in Brazil.
- Retrieve query All The attributes

```
In [2]: df=pd.read_excel("./Absenteeism_at_work.xls")
```

1.2.2 Prepare The Dataset

Exploring Dataset (Data Ingestion)

```
In [3]: df.head()
```

Out[3]:

	ID	Reason for absence	Month of absence	Day of the week	Seasons
0	11	26	7	3	1
1	36	0	7	3	1
2	3	23	7	4	1
3	7	7	7	5	1
4	11	23	7	5	1

	Transportation expense	Distance from Residence to Work	Service time	Age
0	289	36	13	33
1	118	13	18	50
2	179	51	18	38
3	279	5	14	39
4	289	36	13	33

	Work_load_ Average_day	...	Disciplinary failure
0	239554	...	0
1	239554	...	1
2	239554	...	0
3	239554	...	0
4	239554	...	0

	Education	Son	Social drinker	Social smoker	Pet	Weight	Height
0	1	2	1	0	1	90	172
1	1	1	1	0	0	98	178
2	1	0	1	0	0	89	170
3	1	2	1	1	0	68	168
4	1	2	1	0	1	90	172

	Body mass index	Absenteeism time in hours
0	30	4
1	31	0
2	31	2
3	24	4
4	30	2

[5 rows x 21 columns]

Explore The Attributes of the dataset

In [4]:
```
df.columns
```

Out[4]:
```
Index(['ID', 'Reason for absence', 'Month of absence', 'Day of the week',
       'Seasons', 'Transportation expense', 'Distance from Residence to Work',
       'Service time', 'Age', 'Work_load_ Average_day', 'Hit target',
       'Disciplinary failure', 'Education', 'Son', 'Social drinker',
       'Social smoker', 'Pet', 'Weight', 'Height', 'Body mass index',
       'Absenteeism time in hours'],
      dtype='object')
```

Explore The Size of The Dataset

```
In [5]: df.shape
```

```
Out[5]: (740, 21)
```

1.2.3 Preprocessing Data: Cleaning and Transform

Data Munging , Data Wrangling and prepprocessing

- Cleaning any empty or uncomplete emplyee absenteeism record
- clear any empty or uncomplte attribute
- orginize only social drinkers data
- Filtering and classify required attribute to study the absenteeism hours of social drinkers

Check if there is any null value in the dataset

```
In [7]: df.isnull().any().any()
```

```
Out[7]: False
```

The About result indicated that there is null or empty record within the dataset.

```
In [10]: df['Absenteeism time in hours'].describe()
```

```
Out[10]: count    740.000000
         mean       6.924324
         std       13.330998
         min        0.000000
         25%        2.000000
         50%        3.000000
         75%        8.000000
         max      120.000000
         Name: Absenteeism time in hours, dtype: float64
```

```
In [4]: Total_mean=df['Absenteeism time in hours'].mean()
        Total_mean
```

```
Out[4]: 6.924324324324324
```

Classfiyy the only social drinkers

```
In [11]: Social_drinkers_data=df[df['Social drinker']==1]
```

Classfiyy the non social drinkers

```
In [13]: Non_Social_drinker=df[df['Social drinker']==0]
```

Explore The Socail Drinkers Dataset

```
In [9]: Social_drinkers_data.shape
```

```
Out[9]: (420, 21)
```

1.3 Describe The main reason for absenteeism in work for All type of Employees

```
In [14]: Reason_absences=Non_Social_drinker['Reason for absence'].value_counts()
         Reason_absences=Reason_absences.to_frame(name='counts_non')
         Reason_absences.reset_index(inplace=True)
         Reason_absences.rename(columns={'index':'Reason_code'}, inplace=True)
         Reason_absences.head(10)
```

Out[14]:

	Reason_code	counts_non
0	23	82
1	27	31
2	28	25
3	25	25
4	13	21
5	11	15
6	0	14
7	18	12
8	22	12
9	19	11

Reason Code Descrption : The above result indicated the top ten(10) reason for absenceenteeism dental consultation (28), medical consultation (23), physiotherapy (27),XIII Diseases of the musculoskeletal system and connective tissue (13),no reason(0),XIX Injury, poisoning and certain other consequences of external causes ,patient follow-up (22), unjustified absence (26), X Diseases of the respiratory system,XIV Diseases of the genitourinary system

1.3.1 Basic Statstics of employee Absenteeism using .describe() method.

- **count:** The number of rows in the dataset, which were filtered to only Social drinkers/Non Drinkers and Total employees.
- **mean:** the average absent time in hour.
- **std:** the standard deviation.
- **min:** the shortest absent hour in the dataset.
- **25%:** the 25th percentile. 25% of absent hours were lower than .
- **50%:** the 50th percentile, or the median. 50% of absent hours were lower than .
- **75%:** the 75th percentile. 75% of absent hours were lower than .
- **max:** the longest hours in the absenteeism dataset:

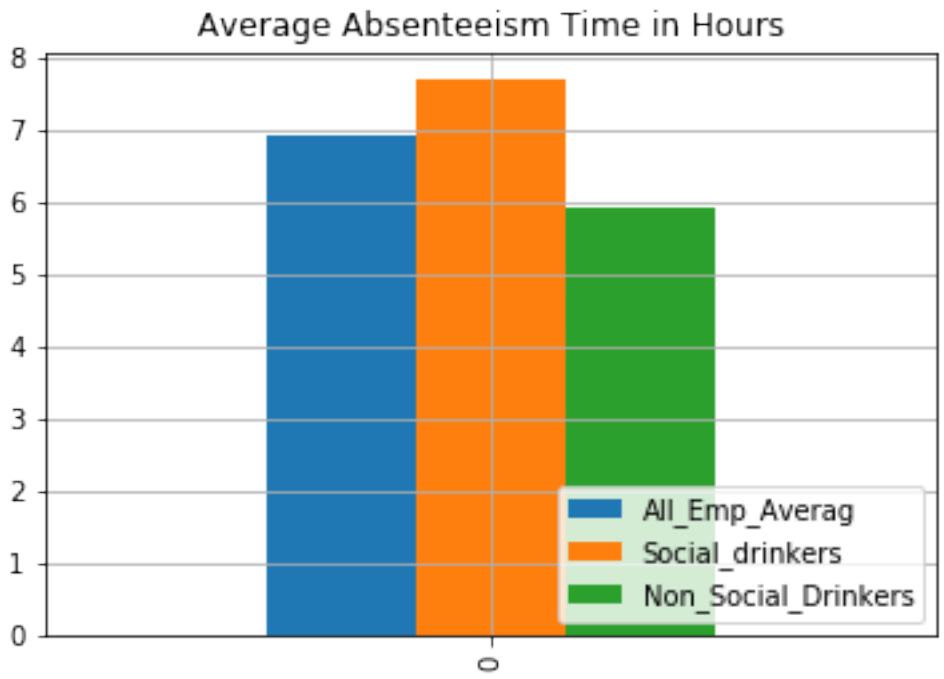
```
In [23]: Non_Social_drinker['Absenteeism time in hours'].describe()
```

```
Out[23]: count    320.000000
         mean       5.931250
         std       12.736353
         min        0.000000
         25%        2.000000
         50%        3.000000
         75%        8.000000
         max      120.000000
         Name: Absenteeism time in hours, dtype: float64
```

```
In [8]: Non_Social_mean=Non_Social_drinker['Absenteeism time in hours'].mean()
        Non_Social_mean

Out[8]: 5.93125

In [24]: Social_drinkers_data['Absenteeism time in hours'].describe()

Out[24]: count    420.000000
         mean       7.680952
         std       13.733680
         min        0.000000
         25%        2.000000
         50%        4.000000
         75%        8.000000
         max      120.000000
         Name: Absenteeism time in hours, dtype: float64

In [11]: Socail_Dr_mean=Social_drinkers_data['Absenteeism time in hours'].mean()
         Socail_Dr_mean

Out[11]: 7.680952380952381
```

1.4 Visualization

```
In [165]: d={'All_Emp_Averag':[Total_mean],'Social_drinkers':[Socail_Dr_mean],'Non_Social_Drinl
          Average_df=pd.DataFrame(data=d)
          plt.figure(figsize=(8,8))
          Average_df.plot(kind='bar')
          plt.title('Average Absenteeism Time in Hours')
          plt.grid(True)
          plt.legend(loc='lower right')
          plt.savefig('C:/DataScienceUCSD/Projects/Final_Project/Average_Abs.png',bbox_inches=

<Figure size 576x576 with 0 Axes>
```




```
In [9]: Top_Reasons=['dental consultation','medical consultation','physiotherapy','Diseases of
```

```
In [10]: Fabs_R['Description']=Top_Reasons
         Fabs_R
```

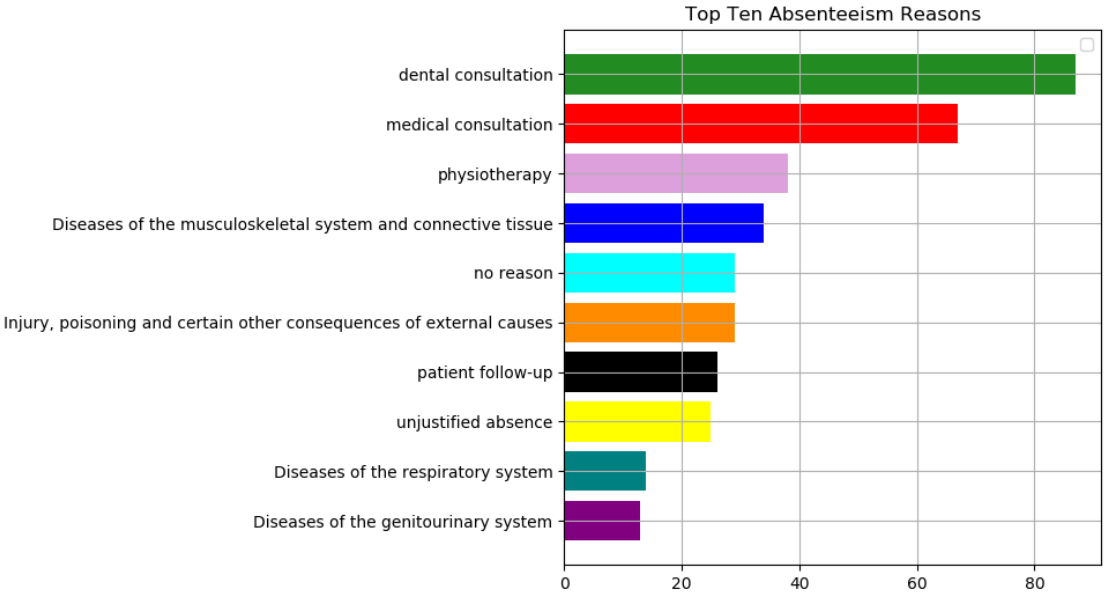
```
C:\Users\asfetu\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  """Entry point for launching an IPython kernel.
```

Out[10]:

	Reason_code	counts	Description
0	28	87	dental consultation
1	23	67	medical consultation
2	27	38	physiotherapy
3	13	34	Diseases of the musculoskeletal system and con...
4	0	29	no reason
5	19	29	Injury, poisoning and certain other consequenc...
6	22	26	patient follow-up
7	26	25	unjustified absence
8	10	14	Diseases of the respiratory system
9	14	13	Diseases of the genitourinary system

In [74]:
```
fig=plt.figure(figsize=(6,6))
plt.barh(Fabs_R['Description'],Fabs_R['counts'],color=['forestgreen', 'red', 'plum',
plt.title('Top Ten Absenteeism Reasons')
plt.gca().invert_yaxis()
plt.legend()
plt.grid(True)
fig.show()
plt.savefig('C:/DataScienceUCSD/Projects/Final_Project/Top_Reason.png',bbox_inches='t
```

```
No handles with labels found to put in legend.
C:\Users\asfetu\Anaconda3\lib\site-packages\matplotlib\figure.py:459: UserWarning: matplotlib
  "matplotlib is currently using a non-GUI backend, "
```



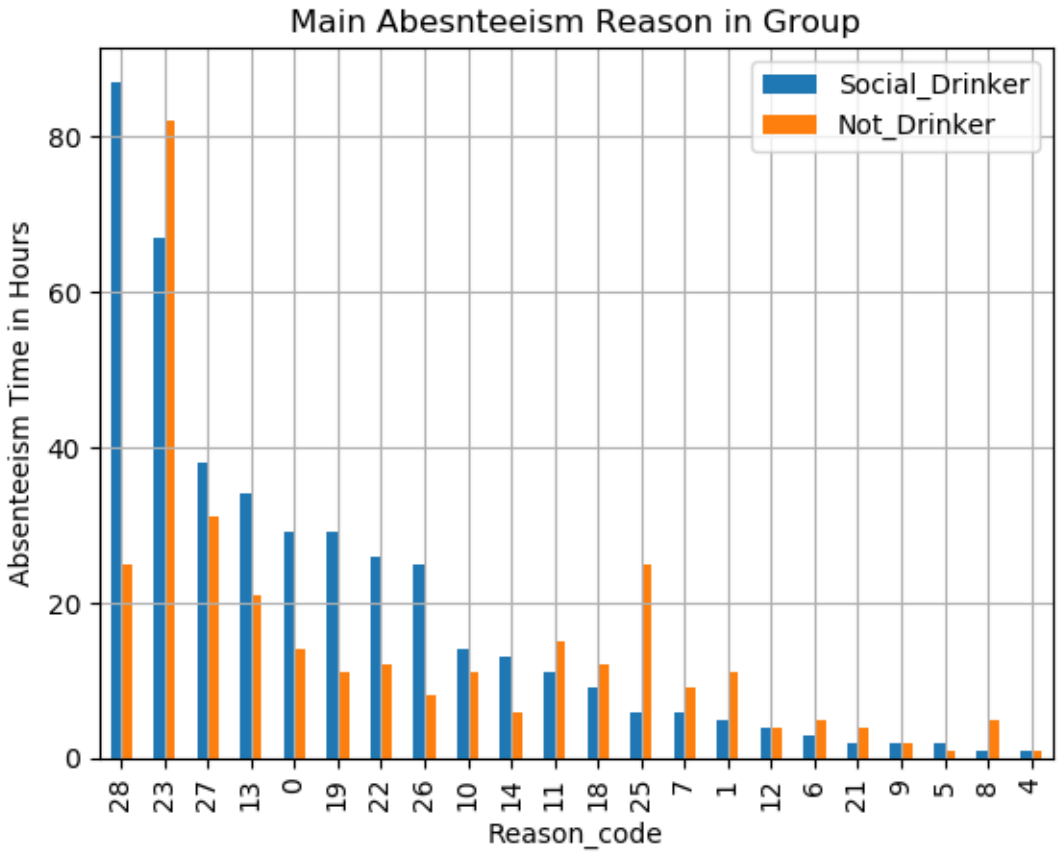
In [11]:
```
both_table=Reason_absence.merge(Reason_absences,on='Reason_code')
both_table.rename(columns={'counts':'Social_Drinker','counts_non':'Not_Drinker'}, inpl
both_table[:3]
```

Out[11]:

	Reason_code	Social_Drinker	Not_Drinker
0	28	87	25
1	23	67	82
2	27	38	31

In [75]:
```
both_table.plot(x='Reason_code', kind='bar')
plt.title('Main Abesnteeism Reason in Group')
plt.ylabel('Absenteeism Time in Hours')
plt.grid(True)
fig.show()
plt.savefig('C:/DataScienceUCSD/Projects/Final_Project/Groupcomparison.png',bbox_inch
```

```
C:\Users\asfetu\Anaconda3\lib\site-packages\matplotlib\figure.py:459: UserWarning: matplotlib i
  "matplotlib is currently using a non-GUI backend, "
```



1.5 Absenteeism Rate in Age Group (per Age per Person)

In [15]:
```
#Age_Group_total_hours
Age_count=df.Age.value_counts()
Age_count=Age_count.to_frame(name='counts').reset_index().rename(columns={'index':'Ag
Age_Absent_Rate=Age_Group_total_hours.merge(Age_count,on='Age')
Age_Absent_Rate['Rate']=Age_Absent_Rate['Absenteeism time in hours']/Age_Absent_Rate[
Age_Absent_Rate.sort_values(by='Rate', ascending=False, inplace=True)
Age_Absent_Rate=Age_Absent_Rate.reset_index()
Age_Absent_Rate.index+=1
del Age_Absent_Rate['index']
Age_Absent_Rate
```

Out[15]:

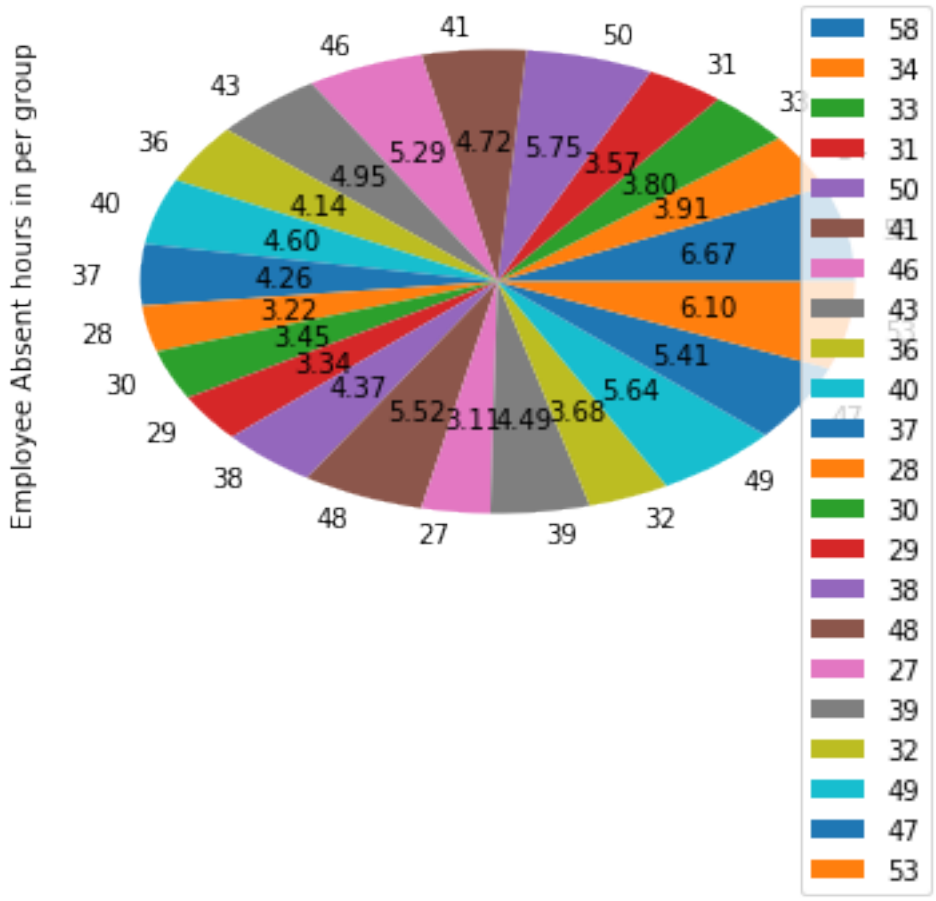
	Age	Absenteeism time in hours	counts	Rate
1	58	262	8	32.750000
2	34	476	29	16.413793

3	33	538	51	10.549020
4	31	217	22	9.863636
5	50	327	37	8.837838
6	41	275	34	8.088235
7	46	16	2	8.000000
8	43	187	24	7.791667
9	36	346	50	6.920000
10	40	379	58	6.534483
11	37	465	78	5.961538
12	28	651	117	5.564103
13	30	253	46	5.500000
14	29	31	7	4.428571
15	38	482	113	4.265487
16	48	25	6	4.166667
17	27	27	7	3.857143
18	39	30	8	3.750000
19	32	48	13	3.692308
20	49	16	5	3.200000
21	47	73	24	3.041667
22	53	0	1	0.000000

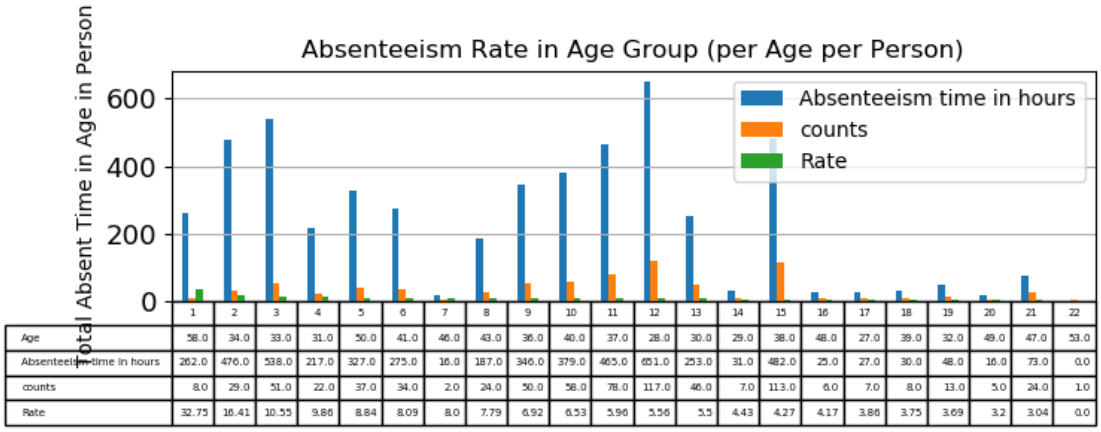
Discussion: The Above data shows that the absenteeism per employee per age .An emplyee at age 58 has most records. Almost an employee at age 58 was absent for 33 hours becuase of the above reasons.

```
In [17]: fig=plt.figure()
         Display= pd.Series(Age_Absent_Rate['Age'], index=Age_Absent_Rate.index, name='Employe
         Display.plot.pie(labels=Age_Absent_Rate['Age'],legend=True, autopct='%.2f', subplots=
         fig.show()
         plt.savefig('C:/DataScienceUCSD/Projects/Final_Project/rating_per_age.jpg')
```

```
C:\Users\asfetu\Anaconda3\lib\site-packages\matplotlib\figure.py:459: UserWarning: matplotlib
  "matplotlib is currently using a non-GUI backend, "
```




```
In [88]: fig, ax = plt.subplots(1, 1)
         ax.get_xaxis().set_visible(False)
         Age_Absent_Rate.plot(x='Age',kind='bar',table=np.round(Age_Absent_Rate.T, 2), ax=ax, i
         plt.title('Absenteeism Rate in Age Group (per Age per Person)')
         plt.ylabel('Total Absent Time in Age in Person')
         plt.grid(True)
         plt.savefig('C:/DataScienceUCSD/Projects/Final_Project/Abs_Age_group.png',bbox_inches=
```



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Age	58.0	34.0	33.0	31.0	50.0	41.0	46.0	43.0	36.0	40.0	37.0	28.0	30.0	29.0	38.0	48.0	27.0	39.0	32.0	49.0	47.0	53.0
Absenteeism time in hours	262.0	476.0	538.0	217.0	327.0	275.0	16.0	187.0	346.0	379.0	465.0	651.0	253.0	31.0	482.0	25.0	27.0	30.0	48.0	16.0	73.0	0.0
counts	8.0	29.0	51.0	22.0	37.0	34.0	2.0	24.0	50.0	58.0	78.0	117.0	46.0	7.0	113.0	6.0	7.0	8.0	13.0	5.0	24.0	1.0
Rate	32.75	16.41	10.55	9.86	8.84	8.09	8.0	7.79	6.92	6.53	5.96	5.56	5.5	4.43	4.27	4.17	3.86	3.75	3.69	3.2	3.04	0.0

2 Test The Accuracy level of Classificatio Model: Decision Tree Classification Model

2.1 Check whether there is Null Value

```
In [16]: df.isnull().any().any()
```

```
Out[16]: False
```

2.2 Convert to Classification Task

According to The catagory classified in the orginal Data source .

Catagory 1 White Absenteeism: Based on ICD * 21 type of Absenteeism Reason **Catagory 2** Black Absenteeism: 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).

```
In [17]: df['Non_ICD']=((df['Reason for absence']==22)*1 )| ((df['Reason for absence']==23)*1)
         #df['Non_ICD']=((df['Reason for absence']==22) | (df['Reason for absence']==23))*1
         df.head()
```

```
Out[17]:    ID  Reason for absence  Month of absence  Day of the week  Seasons  \
         0  11                  26                 7                3        1
         1  36                   0                 7                3        1
         2   3                  23                 7                4        1
         3   7                   7                 7                5        1
         4  11                  23                 7                5        1

            Transportation expense  Distance from Residence to Work  Service time  Age  \
         0                     289                               36            13   33
         1                     118                               13            18   50
         2                     179                               51            18   38
```

```
3                    279                                  5            14   39
4                    289                                 36            13   33

   Work_load_ Average_day   ...     Education  Son  Social drinker  \
0                  239554   ...             1    2               1
1                  239554   ...             1    1               1
2                  239554   ...             1    0               1
3                  239554   ...             1    2               1
4                  239554   ...             1    2               1

   Social smoker  Pet  Weight  Height  Body mass index  \
0              0    1      90     172               30
1              0    0      98     178               31
2              0    0      89     170               31
3              1    0      68     168               24
4              0    1      90     172               30

   Absenteeism time in hours  Non_ICD
0                          4        1
1                          0        0
2                          2        1
3                          4        0
4                          2        1

[5 rows x 22 columns]
```

```
In [21]: Final_data=df.copy()
         type(Final_data)
```

```
Out[21]: pandas.core.frame.DataFrame
```

2.3 Attributes (Features and Target) are saved in X and Y respectively

2.3.1 Target saved in y

```
In [22]: y=Final_data[['Non_ICD']].copy()
```

```
In [23]: type(y)
```

```
Out[23]: pandas.core.frame.DataFrame
```

2.3.2 Features are saved in X to predict the Black or Grey abesenteeism

```
In [24]: Features=['Month of absence','Day of the week',
                'Seasons','Transportation expense','Distance from Residence to Work',
                'Service time','Age','Work_load_ Average_day','Hit target',
                'Disciplinary failure', 'Education','Son','Social drinker',
                'Social smoker','Pet','Weight','Height','Body mass index']
```

```
In [25]: X=Final_data[Features].copy()
```

```
In [26]: type(X)

Out[26]: pandas.core.frame.DataFrame
```

2.4 Perform Test and Train Split

```
In [27]: X_train, X_test, y_train, y_test=train_test_split(X, y, test_size=0.33, random_state=
```

```
In [31]: Black_Absenteeism_Classifier=DecisionTreeClassifier(max_leaf_nodes=20, random_state=0)
         Black_Absenteeism_Classifier.fit(X_train, y_train)

Out[31]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                     max_features=None, max_leaf_nodes=20,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, presort=False, random_state=0,
                     splitter='best')
```

2.4.1 Fit on Train Set

```
In [30]: type(Black_Absenteeism_Classifier)

Out[30]: sklearn.tree.tree.DecisionTreeClassifier
```

2.4.2 Predict on Test Set

```
In [29]: Black_Absent_Prediction=Black_Absenteeism_Classifier.predict(X_test)
```

```
In [32]: Black_Absent_Prediction[:10]

Out[32]: array([1, 1, 1, 1, 1, 0, 1, 1, 1, 0], dtype=int32)
```

```
In [33]: y_test.columns

Out[33]: Index(['Non_ICD'], dtype='object')
```

2.5 Measure The Accuracy of The Classifier

```
In [34]: accuracy_score(y_true=y_test, y_pred=Black_Absent_Prediction)

Out[34]: 0.6612244897959184
```