Absenteeism Prediction

Preprocessing

> Preprocessing of data like dealing with categorical data,grouping of categories,changing data format and etc.

Import libraries

In [3]:
```
import numpy as np
import pandas as pd
```

Load the dataset

In [4]:
```
raw_data = pd.read_csv("Absenteeism_data.csv")
data = raw_data.copy()
data = data.drop(columns=["ID"],axis=1)

#SHOW FULL DATASET
# pd.options.display.max_rows = None
# pd.options.display.max_columns = None
# display(data)

#SUMMARIZE DATASET IN SHORT
# data.info()

# data.describe(include="all")

data.head(10)
```

Out[4]:

	Reason for Absence	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours
0	26	07/07/2015	289	36	33	239.554	30	1	2	1	4
1	0	14/07/2015	118	13	50	239.554	31	1	1	0	0
2	23	15/07/2015	179	51	38	239.554	31	1	0	0	2
3	7	16/07/2015	279	5	39	239.554	24	1	2	0	4
4	23	23/07/2015	289	36	33	239.554	30	1	2	1	2
5	23	10/07/2015	179	51	38	239.554	31	1	0	0	2
6	22	17/07/2015	361	52	28	239.554	27	1	1	4	8
7	23	24/07/2015	260	50	36	239.554	23	1	4	0	4
8	19	06/07/2015	155	12	34	239.554	25	1	2	0	40
9	22	13/07/2015	235	11	37	239.554	29	3	1	1	8

In [6]:
```
data.describe(include="all")
```

Out[6]:

	Reason for Absence	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	A
count	700.000000	700	700.000000	700.000000	700.000000	700.000000	700.000000	700.000000	700.000000	700.000000	
unique	NaN	432	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
top	NaN	17/08/2015	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
freq	NaN	5	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
mean	19.411429	NaN	222.347143	29.892857	36.417143	271.801774	26.737143	1.282857	1.021429	0.687143	
std	8.356292	NaN	66.312960	14.804446	6.379083	40.021804	4.254701	0.668090	1.112215	1.166095	
min	0.000000	NaN	118.000000	5.000000	27.000000	205.917000	19.000000	1.000000	0.000000	0.000000	
25%	13.000000	NaN	179.000000	16.000000	31.000000	241.476000	24.000000	1.000000	0.000000	0.000000	
50%	23.000000	NaN	225.000000	26.000000	37.000000	264.249000	25.000000	1.000000	1.000000	0.000000	
75%	27.000000	NaN	260.000000	50.000000	40.000000	294.217000	31.000000	1.000000	2.000000	1.000000	
max	28.000000	NaN	388.000000	52.000000	58.000000	378.884000	38.000000	4.000000	4.000000	8.000000	

Processing Reason for Absence

In [ ]:
```
# DROPPING THE FIRST COLUMNS TO AVOID MULTICOLLINEARITY(SINCE d1+d2+d3=1 , IF WE KNOW d1,d2 THEN d3 IS ALREAD
Y DEFINED)
reasons_columns = pd.get_dummies(data["Reason for Absence"],drop_first=True)
data = data.drop(["Reason for Absence"],axis=1)
reasons_columns
```

Out[ ]:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	21	22	23	24	25	26	27	28
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
695	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
696	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
697	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
698	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
699	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

700 rows × 27 columns

Group the Reasons for Absence

In [ ]:
```
# LOC METHOD HELPS TO EXTRACT ROWS and COLUMNS WITH labels(HERE LABELS ARE USED NOT INDEXES, IN CASE OF INDEX
ES USE ILOC METHOD)
# DATAFRAME.MAX(axis = 1) find max along rows

reason_1 = reasons_columns.loc[:,1:14].max(axis=1)
reason_2 = reasons_columns.loc[:,15:17].max(axis=1)
reason_3 = reasons_columns.loc[:,18:21].max(axis=1)
reason_4 = reasons_columns.loc[:,22:].max(axis=1)
```

Concatenate dataframes

In [ ]:
```
# ADDING COLUMNS THORUGH CONCATE METHOD
# data = pd.concate([data,reason_type1,reason_type2,reason_type3,reason_type4],axis=1)
# columns_names = ['Date', 'Transportation Expense', 'Distance to Work', 'Age',
#        'Daily Work Load Average', 'Body Mass Index', 'Education',
#        'Children', 'Pets', 'Absenteeism Time in Hours', 'Reason_1', 'Reason_2', 'Reason_3', 'Reason_4']

# ADDING COLUMNS MY WAY
data[["Reason_1","Reason_2","Reason_3","Reason_4"]] = np.column_stack([reason_1,reason_2,reason_3,reason_4])
data.head()
```

Out[ ]:

	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Reason_1	Reason_2	Reason_3
0	07/07/2015	289	36	33	239.554	30	1	2	1	4	0	0	(
1	14/07/2015	118	13	50	239.554	31	1	1	0	0	0	0	(
2	15/07/2015	179	51	38	239.554	31	1	0	0	2	0	0	(
3	16/07/2015	279	5	39	239.554	24	1	2	0	4	1	0	(
4	23/07/2015	289	36	33	239.554	30	1	2	1	2	0	0	(

1. Checkpoint

In [ ]:
```
data_reason_processed = data.copy()
data_reason_processed.head(20)
```

Out[ ]:

	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Reason_1	Reason_2	Reason_
0	07/07/2015	289	36	33	239.554	30	1	2	1	4	0	0	
1	14/07/2015	118	13	50	239.554	31	1	1	0	0	0	0	
2	15/07/2015	179	51	38	239.554	31	1	0	0	2	0	0	
3	16/07/2015	279	5	39	239.554	24	1	2	0	4	1	0	
4	23/07/2015	289	36	33	239.554	30	1	2	1	2	0	0	
5	10/07/2015	179	51	38	239.554	31	1	0	0	2	0	0	
6	17/07/2015	361	52	28	239.554	27	1	1	4	8	0	0	
7	24/07/2015	260	50	36	239.554	23	1	4	0	4	0	0	
8	06/07/2015	155	12	34	239.554	25	1	2	0	40	0	0	
9	13/07/2015	235	11	37	239.554	29	3	1	1	8	0	0	
10	20/07/2015	260	50	36	239.554	23	1	4	0	8	1	0	
11	14/07/2015	260	50	36	239.554	23	1	4	0	8	1	0	
12	15/07/2015	260	50	36	239.554	23	1	4	0	8	1	0	
13	15/07/2015	179	51	38	239.554	31	1	0	0	1	1	0	
14	15/07/2015	179	51	38	239.554	31	1	0	0	4	0	0	
15	17/07/2015	246	25	41	239.554	23	1	0	0	8	1	0	
16	17/07/2015	179	51	38	239.554	31	1	0	0	2	0	0	
17	27/07/2015	179	51	38	239.554	31	1	0	0	8	0	0	
18	30/07/2015	189	29	33	239.554	25	1	2	2	8	1	0	
19	05/08/2015	248	25	47	205.917	32	1	2	1	2	0	0	

Processing Dates

In [ ]:
```
# IT'S IMPORTANT TO PASS THE FORMAT TO CONVERT DATES INTO TIMESTAMP DATA STRUCTURE , OTHERWISE DATES WILL BE
 CONVERTED WRONGLY
data_reason_processed["Date"] = pd.to_datetime(data_reason_processed["Date"], format=("%d/%m/%Y"))
data_reason_processed.head(50)
```

Out[ ]:

	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Reason_1	Reason_2	Reason_3	R
0	2015-07-07	289	36	33	239.554	30	1	2	1	4	0	0	0	
1	2015-07-14	118	13	50	239.554	31	1	1	0	0	0	0	0	
2	2015-07-15	179	51	38	239.554	31	1	0	0	2	0	0	0	
3	2015-07-16	279	5	39	239.554	24	1	2	0	4	1	0	0	
4	2015-07-23	289	36	33	239.554	30	1	2	1	2	0	0	0	
5	2015-07-10	179	51	38	239.554	31	1	0	0	2	0	0	0	
6	2015-07-17	361	52	28	239.554	27	1	1	4	8	0	0	0	
7	2015-07-24	260	50	36	239.554	23	1	4	0	4	0	0	0	
8	2015-07-06	155	12	34	239.554	25	1	2	0	40	0	0	1	
9	2015-07-13	235	11	37	239.554	29	3	1	1	8	0	0	0	
10	2015-07-20	260	50	36	239.554	23	1	4	0	8	1	0	0	
11	2015-07-14	260	50	36	239.554	23	1	4	0	8	1	0	0	
12	2015-07-15	260	50	36	239.554	23	1	4	0	8	1	0	0	
13	2015-07-15	179	51	38	239.554	31	1	0	0	1	1	0	0	
14	2015-07-15	179	51	38	239.554	31	1	0	0	4	0	0	0	
15	2015-07-17	246	25	41	239.554	23	1	0	0	8	1	0	0	

	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Reason_1	Reason_2	Reason_3	R
16	2015-07-17	179	51	38	239.554	31	1	0	0	2	0	0	0	
17	2015-07-27	179	51	38	239.554	31	1	0	0	8	0	0	1	
18	2015-07-30	189	29	33	239.554	25	1	2	2	8	1	0	0	
19	2015-08-05	248	25	47	205.917	32	1	2	1	2	0	0	0	
20	2015-08-12	330	16	28	205.917	25	2	0	0	8	1	0	0	
21	2015-08-03	179	51	38	205.917	31	1	0	0	1	1	0	0	
22	2015-08-10	361	52	28	205.917	27	1	1	4	40	1	0	0	
23	2015-08-14	260	50	36	205.917	23	1	4	0	4	0	0	0	
24	2015-08-17	289	36	33	205.917	30	1	2	1	8	0	0	1	
25	2015-08-24	361	52	28	205.917	27	1	1	4	7	0	0	0	
26	2015-08-04	289	36	33	205.917	30	1	2	1	1	0	0	0	
27	2015-08-12	157	27	29	205.917	22	1	0	0	4	0	0	0	
28	2015-08-19	289	36	33	205.917	30	1	2	1	8	0	0	1	
29	2015-08-28	179	51	38	205.917	31	1	0	0	2	0	0	0	
30	2015-08-17	179	51	38	205.917	31	1	0	0	8	0	0	1	
31	2015-08-27	235	29	48	205.917	33	1	1	5	8	0	0	1	

	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Reason_1	Reason_2	Reason_3	R
32	2015-08-27	235	11	37	205.917	29	3	1	1	4	0	0	0	
33	2015-08-17	235	29	48	205.917	33	1	1	5	8	0	0	1	
34	2015-08-17	179	51	38	205.917	31	1	0	0	2	0	0	0	
35	2015-08-17	361	52	28	205.917	27	1	1	4	1	0	0	0	
36	2015-08-04	289	36	33	205.917	30	1	2	1	8	0	0	0	
37	2015-08-20	291	50	32	205.917	23	1	0	0	4	1	0	0	
38	2015-08-21	235	29	48	205.917	33	1	1	5	8	0	0	0	
39	2015-08-28	260	50	36	205.917	23	1	4	0	4	0	0	0	
40	2015-09-01	184	42	27	241.476	21	1	0	0	2	0	0	0	
41	2015-09-07	118	10	37	241.476	28	1	0	0	4	0	0	0	
42	2015-09-01	179	51	38	241.476	31	1	0	0	4	0	0	0	
43	2015-09-08	235	20	43	241.476	38	1	1	0	8	0	0	1	
44	2015-09-09	155	12	34	241.476	25	1	2	0	2	0	0	0	
45	2015-09-13	118	10	37	241.476	28	1	0	0	3	0	0	0	
46	2015-09-14	179	51	38	241.476	31	1	0	0	3	0	0	0	
47	2015-09-24	291	31	40	241.476	25	1	1	1	4	0	0	0	

	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Reason_1	Reason_2	Reason_3	R
48	2015-09-04	260	50	36	241.476	23	1	4	0	8	0	0	0	
49	2015-09-14	291	31	40	241.476	25	1	1	1	32	1	0	0	

In [ ]:
```
# GETTING DAY AND MONTH FROM DATE AND ADDING THEM AS COLUMN

data_reason_processed["Month Value"] = data_reason_processed["Date"].apply(lambda x : x.month)
data_reason_processed["Day of the Week"] = data_reason_processed["Date"].apply(lambda x: x.weekday())

data_reason_processed = data_reason_processed.drop(["Date"],axis=1)
data_reason_processed.head()
```

Out[ ]:

	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Reason_1	Reason_2	Reason_3	Reason_4
0	289	36	33	239.554	30	1	2	1	4	0	0	0	1
1	118	13	50	239.554	31	1	1	0	0	0	0	0	0
2	179	51	38	239.554	31	1	0	0	2	0	0	0	1
3	279	5	39	239.554	24	1	2	0	4	1	0	0	0
4	289	36	33	239.554	30	1	2	1	2	0	0	0	1

In [ ]:
```
columns_names_ordered = ['Reason_1', 'Reason_2', 'Reason_3', 'Reason_4',
                          'Month Value','Day of the Week', 'Transportation Expense', 'Distance to Work', 'Ag
e',
       'Daily Work Load Average', 'Body Mass Index', 'Education',
       'Children', 'Pets', 'Absenteeism Time in Hours']

data_reason_processed = data_reason_processed[columns_names_ordered]
```

2. Checkpoint

In [ ]:
```
data_reason_date_processed = data_reason_processed.copy()
data_reason_date_processed
```

Out[ ]:

	Reason_1	Reason_2	Reason_3	Reason_4	Month Value	Day of the Week	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	P
0	0	0	0	1	7	1	289	36	33	239.554	30	1	2	
1	0	0	0	0	7	1	118	13	50	239.554	31	1	1	
2	0	0	0	1	7	2	179	51	38	239.554	31	1	0	
3	1	0	0	0	7	3	279	5	39	239.554	24	1	2	
4	0	0	0	1	7	3	289	36	33	239.554	30	1	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
695	1	0	0	0	5	2	179	22	40	237.656	22	2	2	
696	1	0	0	0	5	2	225	26	28	237.656	24	1	1	
697	1	0	0	0	5	3	330	16	28	237.656	25	2	0	
698	0	0	0	1	5	3	235	16	32	237.656	25	3	0	
699	0	0	0	1	5	3	291	31	40	237.656	25	1	1	

700 rows × 15 columns

Processing Education

In [ ]:
```
data_reason_date_processed['Education'] = data_reason_date_processed["Education"].map(lambda x: 0 if x==1 els
e 1)
```

In [ ]:
```
# VALUE_COUNTS counts  NUMBER OF UNIQUE CHARACTERS
data_reason_date_processed["Education"].value_counts()
```

Out[ ]:
```
0    583
1    117
Name: Education, dtype: int64
```

Saving the preprocessed data

In [ ]:
```
df_processed = data_reason_date_processed.copy()
df_processed.to_csv("Absenteeism_preprocessed_data.csv",index=False)
```