In [ ]:

```
Assignment 10 <br>

Referral id: SIRSS1088 <br>
NAME: ONASVEE BANARSE <br>
EMAIL : 2obanarse@gmail.com <br>
COLLEGE: AISSMS IOIT <br>
GitHub : https://github.com/ORION-22/RegexSoftware_ASSIGNMENT.git
```

In [ ]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:

```
df_train=pd.read_csv('../input/heart-attack-analysis-prediction-dataset/heart.csv')
df_o2saturation=pd.read_csv('../input/heart-attack-analysis-prediction-dataset/o2Saturation.csv')
```

In [ ]:

```
df_train['o2saturation']=df_o2scturation
```

In [ ]:

```
df_train
```

In [ ]:

```
df_train.info()
```

In [ ]:

```
cols_order=['age','sex','cp','trtbps','chol','fbs','restecg','thalachh','exng','oldpeak','slp','caa','thall','o2s
df_train=df_train.reindex(columns=cols_order)
```

In [ ]:

```
df_train.head(5)
```

In [ ]:

```
df_train.shape
```

In [ ]:

```
df_train.describe()
```

In [ ]:

```
df_train.info()
```

- age: The person's age in years
- sex: The person's sex (1 = male, 0 = female)
- cp: The chest pain experienced (Value 1: typical angina, Value 2: atypical angina, Value 3:nonanginal pain, Value 4: asymptomatic)
- trtbps: The person's resting blood pressure (mm Hg on admission to the hospital)
- chol: The person's cholesterol measurement in mg/dl
- fbs: The person's fasting blood sugar (> 120 mg/dl, 1 = true; 0 = false)
- restecg: Resting electrocardiographic measurement (0 = normal, 1 = having ST-T wave abnormality, 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria)
- thalachh: The person's maximum heart rate achieved
- exng: Exercise induced angina (1 = yes; 0 = no)
- oldpeak: ST depression induced by exercise relative to rest ('ST' relates to positions on the ECG plot. See more here)
- slp: the slope of the peak exercise ST segment (Value 1: upsloping, Value 2: flat, Value 3: downsloping)
- caa: The number of major vessels (0-3)
- thall: A blood disorder called thalassemia (3 = normal; 6 = fixed defect; 7 = reversable defect)
- output: Heart disease (0 = no, 1 = yes)

In [ ]:

```
df_train.isnull().sum()
```

In [ ]:

```
df_train.corr()
```

In [ ]:

```
sns.heatmap(data=df_train.corr())
```

In [ ]:

```
df_train.loc[df_train['age']<=20,'age']=0
df_train.loc[(df_train['age']>20 )& (df_train['age']<=40),'age'] =1
df_train.loc[(df_train['age']>40)&(df_train['age']<=60),'age'] =2
df_train.loc[df_train['age']>60,'age'] =3
```

In [ ]:

```
sns.countplot(x='sex',hue='age',data=df_train)
```

In [ ]:

```
df_train.plot(figsize=(15,8))
```

In [ ]:

```
list_plot=['age','sex','cp','fbs','restecg','exng','oldpeak','slp','caa','thall']
df_train[list_plot].plot(figsize=(15,8))
```

In [ ]:

```
X_train=df_train.drop(['output'],axis =1).values
Y_train=df_train['output'].values
```

In [ ]:

```
X_train
```

In [ ]:

```
Y_train
```

In [ ]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X_train, Y_train, test_size = 0.20,random_state=5)
```

In [ ]:

```
print(x_train)
```

In [ ]:

```
y_train
```

logistic regression

In [ ]:

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression(solver='liblinear')
model.fit(x_train,y_train)
```

In [ ]:

```
y_pred = model.predict(x_test)
```

In [ ]:

```
from sklearn.metrics import *
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

In [ ]:

```
model.score(x_test,y_test)
```

In [ ]:

```

```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js