In [1]:
```python
import pandas as pd
import numpy as np
```

In [2]:
```python
df1 = pd.read_csv('./heart.csv')
df2 = pd.read_csv('./AirQuality.csv',sep=";")
```

In [3]:
```python
df1.head()
```

Out[3]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0

In [4]:
```python
df2.head()
```

Out[4]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	T	RH	AH	Unnamed: 15	Unnamed: 16
0	10/03/2004	18.00.00	2,6	1360.0	150.0	11,9	1046.0	166.0	1056.0	113.0	1692.0	1268.0	13,6	48,9	0,7578	NaN	NaN
1	10/03/2004	19.00.00	2	1292.0	112.0	9,4	955.0	103.0	1174.0	92.0	1559.0	972.0	13,3	47,7	0,7255	NaN	NaN
2	10/03/2004	20.00.00	2,2	1402.0	88.0	9,0	939.0	131.0	1140.0	114.0	1555.0	1074.0	11,9	54,0	0,7502	NaN	NaN
3	10/03/2004	21.00.00	2,2	1376.0	80.0	9,2	948.0	172.0	1092.0	122.0	1584.0	1203.0	11,0	60,0	0,7867	NaN	NaN
4	10/03/2004	22.00.00	1,6	1272.0	51.0	6,5	836.0	131.0	1205.0	116.0	1490.0	1110.0	11,2	59,6	0,7888	NaN	NaN

DATA CLEANING

In [5]:
```python
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Age             918 non-null    int64
 1   Sex             918 non-null    object
 2   ChestPainType   918 non-null    object
 3   RestingBP       918 non-null    int64
 4   Cholesterol     918 non-null    int64
 5   FastingBS       918 non-null    int64
 6   RestingECG      918 non-null    object
 7   MaxHR           918 non-null    int64
 8   ExerciseAngina  918 non-null    object
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object
 11  HeartDisease    918 non-null    int64
dtypes: float64(1), int64(6), object(5)
memory usage: 86.2+ KB
```

In [6]:
```python
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9471 entries, 0 to 9470
Data columns (total 17 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Date           9357 non-null   object
 1   Time           9357 non-null   object
 2   CO(GT)         9357 non-null   object
 3   PT08.S1(CO)    9357 non-null   float64
 4   NMHC(GT)       9357 non-null   float64
 5   C6H6(GT)       9357 non-null   object
 6   PT08.S2(NMHC)  9357 non-null   float64
 7   NOx(GT)        9357 non-null   float64
 8   PT08.S3(NOx)   9357 non-null   float64
 9   NO2(GT)        9357 non-null   float64
 10  PT08.S4(NO2)   9357 non-null   float64
 11  PT08.S5(O3)    9357 non-null   float64
 12  T              9357 non-null   object
 13  RH             9357 non-null   object
 14  AH             9357 non-null   object
 15  Unnamed: 15    0 non-null      float64
 16  Unnamed: 16    0 non-null      float64
dtypes: float64(10), object(7)
memory usage: 1.2+ MB
```

In [7]:
```python
df1.isnull().sum()
```

Out[7]:
```
Age               0
Sex               0
ChestPainType     0
RestingBP         0
Cholesterol       0
FastingBS         0
RestingECG        0
MaxHR             0
ExerciseAngina    0
Oldpeak           0
ST_Slope          0
HeartDisease      0
dtype: int64
```

In [8]:
```python
df2.isnull().sum()
```

Out[8]:
```
Date              114
Time              114
CO(GT)            114
PT08.S1(CO)       114
NMHC(GT)          114
C6H6(GT)          114
PT08.S2(NMHC)     114
NOx(GT)           114
PT08.S3(NOx)      114
NO2(GT)           114
PT08.S4(NO2)      114
PT08.S5(O3)       114
T                 114
RH                114
AH                114
Unnamed: 15      9471
Unnamed: 16      9471
dtype: int64
```

In [9]:
```python
df2.drop(["Unnamed: 15","Unnamed: 16"],axis=1,inplace=True)
```

In [10]:
```python
df2.sample(10)
```

Out[10]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	T	RH	AH
6361	30/11/2004	19.00.00	6,3	1408.0	-200.0	28,0	1502.0	692.0	640.0	178.0	1843.0	1598.0	11,6	64,3	0,8789
4839	28/09/2004	09.00.00	4,1	1236.0	-200.0	21,3	1332.0	641.0	601.0	152.0	1683.0	1550.0	17,8	45,7	0,9224
3768	14/08/2004	18.00.00	0,9	964.0	-200.0	5,1	769.0	40.0	873.0	56.0	1546.0	572.0	35,2	32,1	1,7898
3262	24/07/2004	16.00.00	1,1	1012.0	-200.0	5,2	775.0	49.0	899.0	53.0	1547.0	562.0	36,6	28,2	1,6970
698	08/04/2004	20.00.00	4,3	1319.0	544.0	15,8	1172.0	232.0	746.0	136.0	1699.0	1425.0	15,8	39,8	0,7096
5421	22/10/2004	15.00.00	3,7	1476.0	-200.0	22,0	1352.0	365.0	474.0	112.0	1996.0	1568.0	26,4	49,8	1,6928
575	03/04/2004	17.00.00	-200	1398.0	-200.0	15,2	1156.0	-200.0	812.0	-200.0	1722.0	1076.0	25,1	27,1	0,8496
8411	24/02/2005	05.00.00	0,6	883.0	-200.0	1,2	518.0	87.0	1135.0	81.0	962.0	606.0	3,3	84,5	0,6612
8646	06/03/2005	00.00.00	2,4	1201.0	-200.0	9,2	949.0	358.0	665.0	170.0	1220.0	1260.0	6,8	72,1	0,7143
3077	16/07/2004	23.00.00	2,2	1090.0	-200.0	13,6	1103.0	142.0	695.0	119.0	1686.0	1235.0	25,7	33,5	1,0918

In [11]:
```python
df2.dropna(inplace=True)
```

In [12]:
```python
df2.tail()
```

Out[12]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	T	RH	AH
9352	04/04/2005	10.00.00	3,1	1314.0	-200.0	13,5	1101.0	472.0	539.0	190.0	1374.0	1729.0	21,9	29,3	0,7568
9353	04/04/2005	11.00.00	2,4	1163.0	-200.0	11,4	1027.0	353.0	604.0	179.0	1264.0	1269.0	24,3	23,7	0,7119
9354	04/04/2005	12.00.00	2,4	1142.0	-200.0	12,4	1063.0	293.0	603.0	175.0	1241.0	1092.0	26,9	18,3	0,6406
9355	04/04/2005	13.00.00	2,1	1003.0	-200.0	9,5	961.0	235.0	702.0	156.0	1041.0	770.0	28,3	13,5	0,5139
9356	04/04/2005	14.00.00	2,2	1071.0	-200.0	11,9	1047.0	265.0	654.0	168.0	1129.0	816.0	28,5	13,1	0,5028

In [13]:
```python
df2.drop_duplicates()
```

Out[13]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	T	RH	AH
0	10/03/2004	18.00.00	2,6	1360.0	150.0	11,9	1046.0	166.0	1056.0	113.0	1692.0	1268.0	13,6	48,9	0,7578
1	10/03/2004	19.00.00	2	1292.0	112.0	9,4	955.0	103.0	1174.0	92.0	1559.0	972.0	13,3	47,7	0,7255
2	10/03/2004	20.00.00	2,2	1402.0	88.0	9,0	939.0	131.0	1140.0	114.0	1555.0	1074.0	11,9	54,0	0,7502
3	10/03/2004	21.00.00	2,2	1376.0	80.0	9,2	948.0	172.0	1092.0	122.0	1584.0	1203.0	11,0	60,0	0,7867
4	10/03/2004	22.00.00	1,6	1272.0	51.0	6,5	836.0	131.0	1205.0	116.0	1490.0	1110.0	11,2	59,6	0,7888
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9352	04/04/2005	10.00.00	3,1	1314.0	-200.0	13,5	1101.0	472.0	539.0	190.0	1374.0	1729.0	21,9	29,3	0,7568
9353	04/04/2005	11.00.00	2,4	1163.0	-200.0	11,4	1027.0	353.0	604.0	179.0	1264.0	1269.0	24,3	23,7	0,7119
9354	04/04/2005	12.00.00	2,4	1142.0	-200.0	12,4	1063.0	293.0	603.0	175.0	1241.0	1092.0	26,9	18,3	0,6406
9355	04/04/2005	13.00.00	2,1	1003.0	-200.0	9,5	961.0	235.0	702.0	156.0	1041.0	770.0	28,3	13,5	0,5139
9356	04/04/2005	14.00.00	2,2	1071.0	-200.0	11,9	1047.0	265.0	654.0	168.0	1129.0	816.0	28,5	13,1	0,5028

9357 rows × 15 columns

In [14]:
```python
df1.drop_duplicates()
```

Out[14]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0
...	...	...	...	...	...	...	...	...	...	...	...	...
913	45	M	TA	110	264	0	Normal	132	N	1.2	Flat	1
914	68	M	ASY	144	193	1	Normal	141	N	3.4	Flat	1
915	57	M	ASY	130	131	0	Normal	115	Y	1.2	Flat	1
916	57	F	ATA	130	236	0	LVH	174	N	0.0	Flat	1
917	38	M	NAP	138	175	0	Normal	173	N	0.0	Up	0

918 rows × 12 columns

In [15]:
```python
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Age             918 non-null    int64
 1   Sex             918 non-null    object
 2   ChestPainType   918 non-null    object
 3   RestingBP       918 non-null    int64
 4   Cholesterol     918 non-null    int64
 5   FastingBS       918 non-null    int64
 6   RestingECG      918 non-null    object
 7   MaxHR           918 non-null    int64
 8   ExerciseAngina  918 non-null    object
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object
 11  HeartDisease    918 non-null    int64
dtypes: float64(1), int64(6), object(5)
memory usage: 86.2+ KB
```

In [16]:
```python
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 9357 entries, 0 to 9356
Data columns (total 15 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Date           9357 non-null   object
 1   Time           9357 non-null   object
 2   CO(GT)         9357 non-null   object
 3   PT08.S1(CO)    9357 non-null   float64
 4   NMHC(GT)       9357 non-null   float64
 5   C6H6(GT)       9357 non-null   object
 6   PT08.S2(NMHC)  9357 non-null   float64
 7   NOx(GT)        9357 non-null   float64
 8   PT08.S3(NOx)   9357 non-null   float64
 9   NO2(GT)        9357 non-null   float64
 10  PT08.S4(NO2)   9357 non-null   float64
 11  PT08.S5(O3)    9357 non-null   float64
 12  T              9357 non-null   object
 13  RH             9357 non-null   object
 14  AH             9357 non-null   object
dtypes: float64(8), object(7)
memory usage: 1.1+ MB
```

In [17]:
```python
df2.isnull().sum()
```

Out[17]:
```
Date             0
Time             0
CO(GT)           0
PT08.S1(CO)      0
NMHC(GT)         0
C6H6(GT)         0
PT08.S2(NMHC)    0
NOx(GT)          0
PT08.S3(NOx)     0
NO2(GT)          0
PT08.S4(NO2)     0
PT08.S5(O3)      0
T                0
RH               0
AH               0
dtype: int64
```

Data transformation

In [18]:
```python
df2.replace(",",".",regex=True, inplace=True)
```

In [19]:
```python
df2.drop(["Date","Time"],axis=1,inplace=True)
```

In [20]:
```python
df2.head()
```

Out[20]:

	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	T	RH	AH
0	2.6	1360.0	150.0	11.9	1046.0	166.0	1056.0	113.0	1692.0	1268.0	13.6	48.9	0.7578
1	2	1292.0	112.0	9.4	955.0	103.0	1174.0	92.0	1559.0	972.0	13.3	47.7	0.7255
2	2.2	1402.0	88.0	9.0	939.0	131.0	1140.0	114.0	1555.0	1074.0	11.9	54.0	0.7502
3	2.2	1376.0	80.0	9.2	948.0	172.0	1092.0	122.0	1584.0	1203.0	11.0	60.0	0.7867
4	1.6	1272.0	51.0	6.5	836.0	131.0	1205.0	116.0	1490.0	1110.0	11.2	59.6	0.7888

In [21]:
```python
df2.dtypes
```

Out[21]:
```
CO(GT)            object
PT08.S1(CO)      float64
NMHC(GT)         float64
C6H6(GT)          object
PT08.S2(NMHC)    float64
NOx(GT)          float64
PT08.S3(NOx)     float64
NO2(GT)          float64
PT08.S4(NO2)     float64
PT08.S5(O3)      float64
T                 object
RH                object
AH                object
dtype: object
```

In [23]:
```python
df2=df2.astype("float64")
```

In [24]:
```python
df2.dtypes
```

Out[24]:
```
CO(GT)           float64
PT08.S1(CO)      float64
NMHC(GT)         float64
C6H6(GT)         float64
PT08.S2(NMHC)    float64
NOx(GT)          float64
PT08.S3(NOx)     float64
NO2(GT)          float64
PT08.S4(NO2)     float64
PT08.S5(O3)      float64
T                float64
RH               float64
AH               float64
dtype: object
```

In [25]:
```python
df1.head()
```

Out[25]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0

In [26]:
```python
df1.Sex.replace(["M","F"],[1,0] , inplace =True)
```

In [27]:
```python
df1.ChestPainType.value_counts()
```

Out[27]:
```
ASY    496
NAP    203
ATA    173
TA      46
Name: ChestPainType, dtype: int64
```

In [28]:
```python
df1.ChestPainType.replace(["ASY","NAP","ATA","TA"],[0,1,2,3], inplace=True)
```

In [29]:
```python
df1.RestingECG.value_counts()
```

Out[29]:
```
Normal    552
LVH       188
ST        178
Name: RestingECG, dtype: int64
```

In [30]:
```python
df1.RestingECG.replace(["Normal","LVH","ST"],[0,1,2],inplace=True)
```

In [31]:
```python
df1.ExerciseAngina.value_counts()
```

Out[31]:
```
N    547
Y    371
Name: ExerciseAngina, dtype: int64
```

In [32]:
```python
df1.ExerciseAngina.replace(["N","Y"],[0,1] , inplace =True)
```

In [33]:
```python
df1.ST_Slope.value_counts()
```

Out[33]:
```
Flat    460
Up      395
Down     63
Name: ST_Slope, dtype: int64
```

In [34]:
```python
df1.ST_Slope.replace(["Flat","Up","Down"],[0,1,2], inplace=True)
```

In [35]:
```python
df1.head()
```

Out[35]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	1	2	140	289	0	0	172	0	0.0	1	0
1	49	0	1	160	180	0	0	156	0	1.0	0	1
2	37	1	2	130	283	0	2	98	0	0.0	1	0
3	48	0	0	138	214	0	0	108	1	1.5	0	1
4	54	1	1	150	195	0	0	122	0	0.0	1	0

Data integration

In [36]:
```python
df1.shape
```

Out[36]:
```
(918, 12)
```

In [37]:
```python
df2.shape
```

Out[37]:
```
(9357, 13)
```

In [38]:
```python
final_df = pd.concat([df2.iloc[:800,:], df1.iloc[:800,:]], axis=1)
final_df
```

Out[38]:

	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	...	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldp
0	2.6	1360.0	150.0	11.9	1046.0	166.0	1056.0	113.0	1692.0	1268.0	...	2	140	289	0	0	172	0	
1	2.0	1292.0	112.0	9.4	955.0	103.0	1174.0	92.0	1559.0	972.0	...	1	160	180	0	0	156	0	
2	2.2	1402.0	88.0	9.0	939.0	131.0	1140.0	114.0	1555.0	1074.0	...	2	130	283	0	2	98	0	
3	2.2	1376.0	80.0	9.2	948.0	172.0	1092.0	122.0	1584.0	1203.0	...	0	138	214	0	0	108	1	
4	1.6	1272.0	51.0	6.5	836.0	131.0	1205.0	116.0	1490.0	1110.0	...	1	150	195	0	0	122	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
795	0.8	904.0	56.0	2.7	635.0	66.0	1269.0	74.0	1233.0	489.0	...	1	120	240	1	0	194	0	
796	0.8	913.0	49.0	2.6	626.0	61.0	1271.0	68.0	1234.0	474.0	...	0	134	409	0	1	150	1	
797	0.8	914.0	56.0	2.4	610.0	54.0	1322.0	65.0	1230.0	455.0	...	0	110	172	0	1	158	0	
798	0.7	893.0	48.0	2.0	587.0	46.0	1368.0	58.0	1219.0	433.0	...	0	102	265	0	1	122	0	
799	0.5	844.0	27.0	1.3	532.0	25.0	1499.0	36.0	1176.0	376.0	...	1	130	246	1	1	173	0	

800 rows × 25 columns

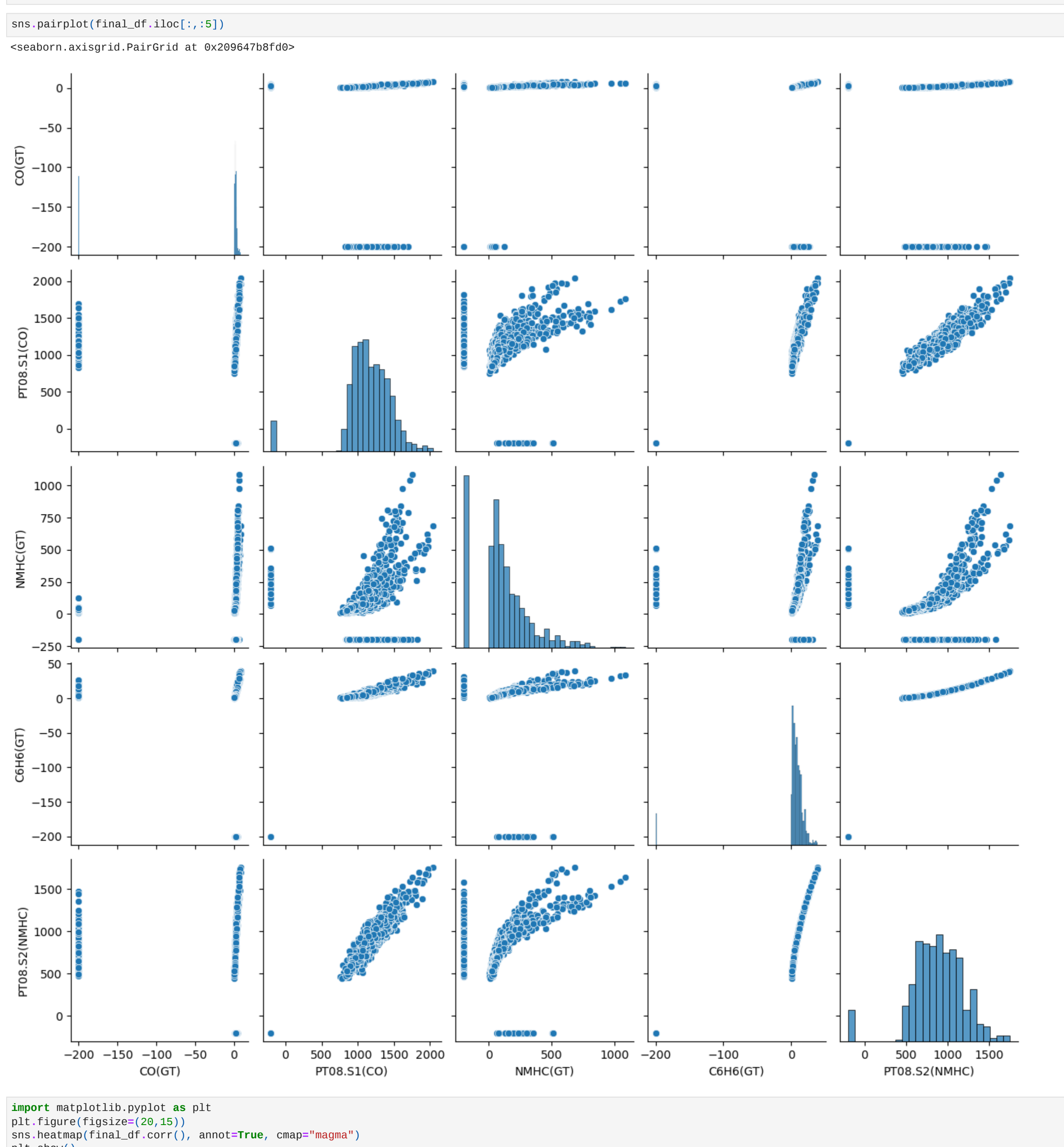
In [39]:
```python
final_df.columns
```

Out[39]:
```
Index(['CO(GT)', 'PT08.S1(CO)', 'NMHC(GT)', 'C6H6(GT)', 'PT08.S2(NMHC)',
       'NOx(GT)', 'PT08.S3(NOx)', 'NO2(GT)', 'PT08.S4(NO2)', 'PT08.S5(O3)',
       'T', 'RH', 'AH', 'Age', 'Sex', 'ChestPainType', 'RestingBP',
       'Cholesterol', 'FastingBS', 'RestingECG', 'MaxHR', 'ExerciseAngina',
       'Oldpeak', 'ST_Slope', 'HeartDisease'],
      dtype='object')
```

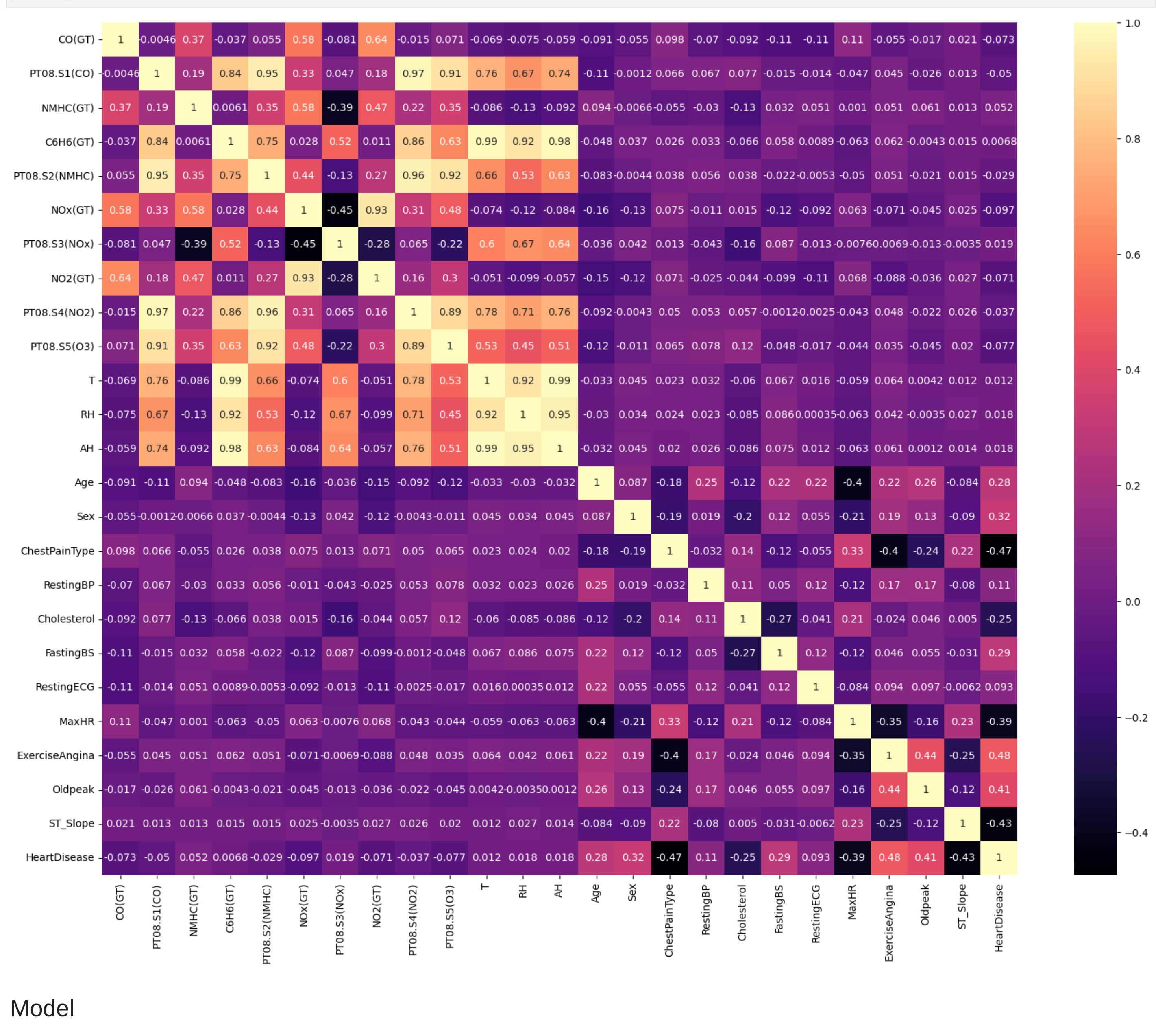
In [42]:
```python
import seaborn as sns
```

In [43]:
```python
sns.pairplot(final_df.iloc[:,:5])
```

Out[43]:
```
<seaborn.axisgrid.PairGrid at 0x209647b8fd0>
```



In [44]:
```python
import matplotlib.pyplot as plt
plt.figure(figsize=(20,15))
sns.heatmap(final_df.corr(), annot=True, cmap="magma")
plt.show()
```



Model

In [45]:
```python
X = final_df.drop(["HeartDisease"] , axis =1 )
```

In [46]:
```python
y = final_df.HeartDisease
```

In [48]:
```python
X
```

Out[48]:

	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	...	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina
0	2.6	1360.0	150.0	11.9	1046.0	166.0	1056.0	113.0	1692.0	1268.0	...	1	2	140	289	0	0	172	0
1	2.0	1292.0	112.0	9.4	955.0	103.0	1174.0	92.0	1559.0	972.0	...	0	1	160	180	0	0	156	0
2	2.2	1402.0	88.0	9.0	939.0	131.0	1140.0	114.0	1555.0	1074.0	...	1	2	130	283	0	2	98	0
3	2.2	1376.0	80.0	9.2	948.0	172.0	1092.0	122.0	1584.0	1203.0	...	0	0	138	214	0	0	108	1
4	1.6	1272.0	51.0	6.5	836.0	131.0	1205.0	116.0	1490.0	1110.0	...	1	1	150	195	0	0	122	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
795	0.8	904.0	56.0	2.7	635.0	66.0	1269.0	74.0	1233.0	489.0	...	1	1	120	240	1	0	194	0
796	0.8	913.0	49.0	2.6	626.0	61.0	1271.0	68.0	1234.0	474.0	...	0	0	134	409	0	1	150	1
797	0.8	914.0	56.0	2.4	610.0	54.0	1322.0	65.0	1230.0	455.0	...	1	0	110	172	0	1	158	0
798	0.7	893.0	48.0	2.0	587.0	46.0	1368.0	58.0	1219.0	433.0	...	0	0	102	265	0	1	122	0
799	0.5	844.0	27.0	1.3	532.0	25.0	1499.0	36.0	1176.0	376.0	...	1	1	130	246	1	1	173	0

800 rows × 24 columns

In [49]:
```python
y
```

Out[49]:
```
0      0
1      1
2      0
3      1
4      0
      ..
795    0
796    1
797    1
798    0
799    0
Name: HeartDisease, Length: 800, dtype: int64
```

In [52]:
```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

In [53]:
```python
Xtrain,Xtest,ytrain,ytest = train_test_split(X,y,test_size=0.2, random_state=7)

model = LogisticRegression()
model.fit(Xtrain, ytrain)
```

```
C:\Users\Vinay.Mahajan\AppData\Local\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[53]:
```
▾ LogisticRegression
LogisticRegression()
```

In [54]:
```python
final_pred=model.predict(Xtest)
final_pred
```

Out[54]:
```
array([0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0,
       1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0,
       0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0,
       1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1,
       0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0,
       1, 1, 1, 1, 1, 1], dtype=int64)
```

In [55]:
```python
pd.DataFrame({"Actual":ytest, "Predict":final_pred})
```

Out[55]:

	Actual	Predict
61	0	0
799	0	0
747	0	0
668	0	0
726	0	0
...	...	...
261	0	1
556	0	1
251	1	1
442	1	1
326	0	1

160 rows × 2 columns

In [56]:
```python
print("Accuracy of the Logistics Regression Model is ::",accuracy_score(ytest, final_pred)*100, "%")
```

```
Accuracy of the Logistics Regression Model is :: 70.0 %
```

In [ ]: