In [1]:
```
import sklearn as sk
import pandas as pd
import os
from sklearn.linear_model import LogisticRegression
```

In [2]:
```
os.chdir('/Python/XPProject/HeartClassification/')
data = pd.read_csv('dataset.csv', sep=',',header=0)
data.head()
```

```
       HeartDisease    BMI Smoking AlcoholDrinking Stroke  PhysicalHealth  \
0                No  16.60     Yes              No     No             3.0
1                No  20.34      No              No    Yes             0.0
2                No  26.58     Yes              No     No            20.0
3                No  24.21      No              No     No             0.0
4                No  23.71      No              No     No            28.0
...             ...    ...     ...             ...    ...             ...
319790          Yes  27.41     Yes              No     No             7.0
319791           No  29.84     Yes              No     No             0.0
319792           No  24.24      No              No     No             0.0
319793           No  32.81      No              No     No             0.0
319794           No  46.56      No              No     No             0.0

        MentalHealth DiffWalking     Sex  AgeCategory      Race Diabetic  \
0               30.0          No  Female        55-59     White      Yes
1                0.0          No  Female  80 or older     White       No
2               30.0          No    Male        65-69     White      Yes
3                0.0          No  Female        75-79     White       No
4                0.0         Yes  Female        40-44     White       No
...              ...         ...     ...          ...       ...      ...
319790           0.0         Yes    Male        60-64  Hispanic      Yes
319791           0.0          No    Male        35-39  Hispanic       No
319792           0.0          No  Female        45-49  Hispanic       No
319793           0.0          No  Female        25-29  Hispanic       No
319794           0.0          No  Female  80 or older  Hispanic       No

       PhysicalActivity  GenHealth  SleepTime Asthma KidneyDisease SkinCancer
0                   Yes  Very good        5.0    Yes            No        Yes
1                   Yes  Very good        7.0     No            No         No
2                   Yes       Fair        8.0    Yes            No         No
3                    No       Good        6.0     No            No        Yes
4                   Yes  Very good        8.0     No            No         No
...                 ...        ...        ...    ...           ...        ...
319790               No       Fair        6.0    Yes            No         No
319791              Yes  Very good        5.0    Yes            No         No
319792              Yes       Good        6.0     No            No         No
319793               No       Good       12.0     No            No         No
319794              Yes       Good        8.0     No            No         No

[319795 rows x 18 columns]
```

In [5]:
```
for feature in data:
    print(feature)
    print(data[feature].unique(),"\n")
```

```
HeartDisease
['No' 'Yes']

BMI
[16.6  20.34 26.58 ... 62.42 51.46 46.56]

Smoking
['Yes' 'No']

AlcoholDrinking
['No' 'Yes']

Stroke
['No' 'Yes']

PhysicalHealth
[ 3.  0. 20. 28.  6. 15.  5. 30.  7.  1.  2. 21.  4. 10. 14. 18.  8. 25.
 16. 29. 27. 17. 24. 12. 23. 26. 22. 19.  9. 13. 11.]

MentalHealth
[30.  0.  2.  5. 15.  8.  4.  3. 10. 14. 20.  1.  7. 24.  9. 28. 16. 12.
  6. 25. 17. 18. 21. 29. 22. 13. 23. 27. 26. 11. 19.]

DiffWalking
['No' 'Yes']

Sex
['Female' 'Male']

AgeCategory
['55-59' '80 or older' '65-69' '75-79' '40-44' '70-74' '60-64' '50-54'
 '45-49' '18-24' '35-39' '30-34' '25-29']

Race
['White' 'Black' 'Asian' 'American Indian/Alaskan Native' 'Other'
 'Hispanic']

Diabetic
['Yes' 'No' 'No, borderline diabetes' 'Yes (during pregnancy)']

PhysicalActivity
['Yes' 'No']

GenHealth
['Very good' 'Fair' 'Good' 'Poor' 'Excellent']

SleepTime
[ 5.  7.  8.  6. 12.  4.  9. 10. 15.  3.  2.  1. 16. 18. 14. 20. 11. 13.
 17. 24. 19. 21. 22. 23.]

Asthma
['Yes' 'No']

KidneyDisease
['No' 'Yes']

SkinCancer
['Yes' 'No']
```

In [4]:
```
for feature in data:
    print(feature)
    print(data[feature].unique(),"\n")
```

```
HeartDisease
['No' 'Yes']

BMI
[16.6  20.34 26.58 ... 62.42 51.46 46.56]

Smoking
['Yes' 'No']

AlcoholDrinking
['No' 'Yes']

Stroke
['No' 'Yes']

PhysicalHealth
[ 3.  0. 20. 28.  6. 15.  5. 30.  7.  1.  2. 21.  4. 10. 14. 18.  8. 25.
 16. 29. 27. 17. 24. 12. 23. 26. 22. 19.  9. 13. 11.]

MentalHealth
[30.  0.  2.  5. 15.  8.  4.  3. 10. 14. 20.  1.  7. 24.  9. 28. 16. 12.
  6. 25. 17. 18. 21. 29. 22. 13. 23. 27. 26. 11. 19.]

DiffWalking
['No' 'Yes']

Sex
['Female' 'Male']

AgeCategory
['55-59' '80 or older' '65-69' '75-79' '40-44' '70-74' '60-64' '50-54'
 '45-49' '18-24' '35-39' '30-34' '25-29']

Race
['White' 'Black' 'Asian' 'American Indian/Alaskan Native' 'Other'
 'Hispanic']

Diabetic
['Yes' 'No' 'No, borderline diabetes' 'Yes (during pregnancy)']

PhysicalActivity
['Yes' 'No']

GenHealth
['Very good' 'Fair' 'Good' 'Poor' 'Excellent']

SleepTime
[ 5.  7.  8.  6. 12.  4.  9. 10. 15.  3.  2.  1. 16. 18. 14. 20. 11. 13.
 17. 24. 19. 21. 22. 23.]

Asthma
['Yes' 'No']

KidneyDisease
['No' 'Yes']

SkinCancer
['Yes' 'No']
```

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 319795 entries, 0 to 319794
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   HeartDisease      319795 non-null  object 
 1   BMI               319795 non-null  float64
 2   Smoking           319795 non-null  object 
 3   AlcoholDrinking   319795 non-null  object 
 4   Stroke            319795 non-null  object 
 5   PhysicalHealth    319795 non-null  float64
 6   MentalHealth      319795 non-null  float64
 7   DiffWalking       319795 non-null  object 
 8   Sex               319795 non-null  object 
 9   AgeCategory       319795 non-null  object 
 10  Race              319795 non-null  object 
 11  Diabetic          319795 non-null  object 
 12  PhysicalActivity  319795 non-null  object 
 13  GenHealth         319795 non-null  object 
 14  SleepTime         319795 non-null  float64
 15  Asthma            319795 non-null  object 
 16  KidneyDisease     319795 non-null  object 
 17  SkinCancer        319795 non-null  object 
dtypes: float64(4), object(14)
memory usage: 43.9+ MB
```

In [7]:
```
cat_features = []
num_features = []
for column, i in zip(data.columns, data.dtypes):
    if i == object:
        cat_features.append(column)
    else:
        num_features.append(column)
```

In [10]:
```
from sklearn.preprocessing import OrdinalEncoder

df_cat = data[cat_features].copy()
ordinal_encoder = OrdinalEncoder()
df_cat_encoded = ordinal_encoder.fit_transform(df_cat)
df_cat_encoded = pd.DataFrame(df_cat_encoded, columns = cat_features)
df_cat_encoded.head()
```

Out[10]:

	HeartDisease	Smoking	AlcoholDrinking	Stroke	DiffWalking	Sex	AgeCategory	Race	Diabetic
0	0.0	1.0	0.0	0.0	0.0	0.0	7.0	5.0	2.0
1	0.0	0.0	0.0	1.0	0.0	0.0	12.0	5.0	0.0
2	0.0	1.0	0.0	0.0	0.0	1.0	9.0	5.0	2.0
3	0.0	0.0	0.0	0.0	0.0	0.0	11.0	5.0	0.0
4	0.0	0.0	0.0	0.0	1.0	0.0	4.0	5.0	0.0

In [11]:
```
for feature in df_cat_encoded.columns:
    print(feature)
```

```
    print(df_cat_encoded[feature].unique(),"\n")
```

```
HeartDisease
[0. 1.]

Smoking
[1. 0.]

AlcoholDrinking
[0. 1.]

Stroke
[0. 1.]

DiffWalking
[0. 1.]

Sex
[0. 1.]

AgeCategory
[ 7. 12.  9. 11.  4. 10.  8.  6.  5.  0.  3.  2.  1.]

Race
[5. 2. 1. 0. 4. 3.]

Diabetic
[2. 0. 1. 3.]

PhysicalActivity
[1. 0.]

GenHealth
[4. 1. 2. 3. 0.]

Asthma
[1. 0.]

KidneyDisease
[0. 1.]

SkinCancer
[1. 0.]
```

In [15]:
```
data_merged = pd.merge(df_cat_encoded, data[num_features],left_index=True, right_index
```

In [16]:
```
data_merged.head(10)
```

Out[16]:

	HeartDisease	Smoking	AlcoholDrinking	Stroke	DiffWalking	Sex	AgeCategory	Race	Diabetic
0	0.0	1.0	0.0	0.0	0.0	0.0	7.0	5.0	2.0
1	0.0	0.0	0.0	1.0	0.0	0.0	12.0	5.0	0.0
2	0.0	1.0	0.0	0.0	0.0	1.0	9.0	5.0	2.0
3	0.0	0.0	0.0	0.0	0.0	0.0	11.0	5.0	0.0
4	0.0	0.0	0.0	0.0	1.0	0.0	4.0	5.0	0.0
5	1.0	1.0	0.0	0.0	1.0	0.0	11.0	2.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	10.0	5.0	0.0
7	0.0	1.0	0.0	0.0	1.0	0.0	12.0	5.0	2.0
8	0.0	0.0	0.0	0.0	0.0	0.0	12.0	5.0	1.0
9	0.0	0.0	0.0	0.0	1.0	1.0	9.0	5.0	0.0

In [18]:
```
from sklearn.preprocessing import StandardScaler
stand_scale = StandardScaler()
df_num = data[num_features].copy()
```

In [19]:
```
df_num_scaler = stand_scale.fit_transform(df_num)
df_num_scaler = pd.DataFrame(df_num_scaler, columns = num_features)
df_num_scaler
```

Out[19]:

	BMI	PhysicalHealth	MentalHealth	SleepTime
0	-1.844750	-0.046751	3.281069	-1.460354
1	-1.256338	-0.424070	-0.490039	-0.067601
2	-0.274603	2.091388	3.281069	0.628776
3	-0.647473	-0.424070	-0.490039	-0.763977
4	-0.726138	3.097572	-0.490039	0.628776
...	...	...	...	...
319790	-0.144019	0.456341	-0.490039	-0.763977
319791	0.238291	-0.424070	-0.490039	-1.460354
319792	-0.642753	-0.424070	-0.490039	-0.763977
319793	0.705560	-0.424070	-0.490039	3.414282
319794	2.868839	-0.424070	-0.490039	0.628776

319795 rows × 4 columns

In [22]:
```
data_ready = pd.merge(df_cat_encoded, df_num_scaler,left_index=True, right_index=True)
```

In [24]:
```
data_ready.head(10)
```

Out[24]:

	HeartDisease	Smoking	AlcoholDrinking	Stroke	DiffWalking	Sex	AgeCategory	Race	Diabetic
0	0.0	1.0	0.0	0.0	0.0	0.0	7.0	5.0	2.0
1	0.0	0.0	0.0	1.0	0.0	0.0	12.0	5.0	0.0
2	0.0	1.0	0.0	0.0	0.0	1.0	9.0	5.0	2.0
3	0.0	0.0	0.0	0.0	0.0	0.0	11.0	5.0	0.0
4	0.0	0.0	0.0	0.0	1.0	0.0	4.0	5.0	0.0
5	1.0	1.0	0.0	0.0	1.0	0.0	11.0	2.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	10.0	5.0	0.0
7	0.0	1.0	0.0	0.0	1.0	0.0	12.0	5.0	2.0
8	0.0	0.0	0.0	0.0	0.0	0.0	12.0	5.0	1.0
9	0.0	0.0	0.0	0.0	1.0	1.0	9.0	5.0	0.0

In [27]:
```
X = data_ready.iloc[:,1:]
y = data_ready.iloc[:,0]
```

In [29]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y , test_size=0.25, random_stat
```

In [30]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_train, y_train)
```

Out[30]:
```
LogisticRegression()
```

In [31]:
```
predictions = model.predict(X_test)
```

In [32]:
```
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, predictions)

TN, FP, FN, TP = confusion_matrix(y_test, predictions).ravel()

print('True Positive(TP)  = ', TP)
print('False Positive(FP) = ', FP)
print('True Negative(TN)  = ', TN)
print('False Negative(FN) = ', FN)

accuracy =  (TP+TN) /(TP+FP+TN+FN)

print('Accuracy of the binary classification = {:0.3f}'.format(accuracy))
```

```
True Positive(TP)  =  597
False Positive(FP) =  585
True Negative(TN)  =  72552
False Negative(FN) =  6215
Accuracy of the binary classification = 0.915
```

In [ ]: