

SVM Classifier

In [1]: *#Exp no.:12*

In [2]: *#Aim: To perform SVM Classifier*

In [3]: *#Name: Riya Anand Kedar
#Roll No.:07
#Sec: 3B
#Subject:ET - 1
#Date: 10/10/2024*

Importing Libraries

In [4]: `import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')`

In [5]: `import os`

In [6]: `os.getcwd()`

Out[6]: 'C:\\Users\\riyak'

In [7]: `os.chdir("C:\\Users\\riyak\\OneDrive\\Desktop")`

In [8]: `df=pd.read_csv("framingham.csv")`

In [9]: *#The "Framingham" heart disease dataset includes over 4,240 records, 15 attributes
#The goal of the dataset is to predict whether the patient has 10-year risk of*

In [10]: `df.head()`

Out[10]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	di
0	1	39	4.0	0	0.0	0.0	0	0	
1	0	46	2.0	0	0.0	0.0	0	0	
2	1	48	1.0	1	20.0	0.0	0	0	
3	0	61	3.0	1	30.0	0.0	0	1	
4	0	46	3.0	1	23.0	0.0	0	0	

In [11]: `df.describe()`

Out[11]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	di
count	4238.000000	4238.000000	4133.000000	4238.000000	4209.000000	4185.000000			423
mean	0.429212	49.584946	1.978950	0.494101	9.003089	0.029630			
std	0.495022	8.572160	1.019791	0.500024	11.920094	0.169584			
min	0.000000	32.000000	1.000000	0.000000	0.000000	0.000000			
25%	0.000000	42.000000	1.000000	0.000000	0.000000	0.000000			
50%	0.000000	49.000000	2.000000	0.000000	0.000000	0.000000			
75%	1.000000	56.000000	3.000000	1.000000	20.000000	0.000000			
max	1.000000	70.000000	4.000000	1.000000	70.000000	1.000000			

In [12]: `df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4238 entries, 0 to 4237
Data columns (total 16 columns):
#   Column                Non-Null Count  Dtype
---  -
0   male                   4238 non-null   int64
1   age                    4238 non-null   int64
2   education               4133 non-null   float64
3   currentSmoker           4238 non-null   int64
4   cigsPerDay              4209 non-null   float64
5   BPMeds                  4185 non-null   float64
6   prevalentStroke         4238 non-null   int64
7   prevalentHyp            4238 non-null   int64
8   diabetes                4238 non-null   int64
9   totChol                 4188 non-null   float64
10  sysBP                   4238 non-null   float64
11  diaBP                   4238 non-null   float64
12  BMI                     4219 non-null   float64
13  heartRate               4237 non-null   float64
14  glucose                 3850 non-null   float64
15  TenYearCHD              4238 non-null   int64
dtypes: float64(9), int64(7)
memory usage: 529.9 KB
```

In [13]: `df.isna().sum()`

```
Out[13]: male                0
age                0
education          105
currentSmoker      0
cigsPerDay         29
BPMeds             53
prevalentStroke    0
prevalentHyp       0
diabetes           0
totChol            50
sysBP              0
diaBP              0
BMI                19
heartRate          1
glucose            388
TenYearCHD         0
dtype: int64
```

In [14]: *#Since, only a few rows have null values in them, we are only removing those r*
#df = df.dropna(subset=['heartRate', 'BMI', 'cigsPerDay', 'totChol', 'BPMeds'])

In [15]: df

Out[15]:

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp
0	1	39	4.0	0	0.0	0.0	0	0
1	0	46	2.0	0	0.0	0.0	0	0
2	1	48	1.0	1	20.0	0.0	0	0
3	0	61	3.0	1	30.0	0.0	0	1
4	0	46	3.0	1	23.0	0.0	0	0
...
4233	1	50	1.0	1	1.0	0.0	0	1
4234	1	51	3.0	1	43.0	0.0	0	0
4235	0	48	2.0	1	20.0	NaN	0	0
4236	0	44	1.0	1	15.0	0.0	0	0
4237	0	52	2.0	0	0.0	0.0	0	0

4238 rows × 16 columns



Missing Value Treatment

Since, 'glucose' and 'education' columns had a significant amount of null values, so we replaced them with the mean of values for their respective columns

In [16]: df['glucose'].fillna(value = df['glucose'].mean(),inplace=True)

In [17]: df['education'].fillna(value = df['education'].mean(),inplace=True)

In [18]: df['heartRate'].fillna(value = df['heartRate'].mean(),inplace=True)

In [19]: df['BMI'].fillna(value = df['BMI'].mean(),inplace=True)

In [20]: df['cigsPerDay'].fillna(value = df['cigsPerDay'].mean(),inplace=True)

In [21]: df['totChol'].fillna(value = df['totChol'].mean(),inplace=True)

In [22]: df['BPMeds'].fillna(value = df['BPMeds'].mean(),inplace=True)

```
In [23]: df.isna().sum()
```

```
Out[23]: male          0
age          0
education    0
currentSmoker 0
cigsPerDay   0
BPMeds       0
prevalentStroke 0
prevalentHyp 0
diabetes     0
totChol      0
sysBP        0
diaBP        0
BMI          0
heartRate    0
glucose      0
TenYearCHD   0
dtype: int64
```

```
In [24]: #Splitting the dependent and independent variables.
x = df.drop("TenYearCHD",axis=1)
y = df['TenYearCHD']
```

```
In [25]: x #checking the features
```

```
Out[25]:
```

	male	age	education	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp
0	1	39	4.0	0	0.0	0.00000	0	0
1	0	46	2.0	0	0.0	0.00000	0	0
2	1	48	1.0	1	20.0	0.00000	0	0
3	0	61	3.0	1	30.0	0.00000	0	1
4	0	46	3.0	1	23.0	0.00000	0	0
...
4233	1	50	1.0	1	1.0	0.00000	0	1
4234	1	51	3.0	1	43.0	0.00000	0	0
4235	0	48	2.0	1	20.0	0.02963	0	0
4236	0	44	1.0	1	15.0	0.00000	0	0
4237	0	52	2.0	0	0.0	0.00000	0	0

4238 rows × 15 columns



Train Test Split

```
In [26]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_stat
```

```
In [27]: y_train
```

```
Out[27]: 3252    0
          3946    0
          1261    0
          2536    0
          4089    0
          ..
          3444    0
          466     0
          3092    0
          3772    0
          860     0
          Name: TenYearCHD, Length: 3390, dtype: int64
```

SVM Classifier

```
In [28]: from sklearn.svm import SVC
          from sklearn.metrics import accuracy_score
          svc=SVC()
          svc.fit(x_test,y_test)
          acc = svc.score(x_test,y_test)*100
          print(acc)
```

```
85.37735849056604
```

```
In [ ]:
```