IMPORTING LIBRARIES

```
import numpy as np
import pandas as pd
df=pd.read_csv('/content/stroke.csv')
df
```

	id	gender	age	hypertension	heart_disease	ever_married	work_
0	9046	Male	67.0	0	1	Yes	P
1	51676	Female	61.0	0	0	Yes	emp
2	31112	Male	80.0	0	1	Yes	P
3	60182	Female	49.0	0	0	Yes	P
4	1665	Female	79.0	1	0	Yes	emp
...	...	...	...	...	...	...	
5105	18234	Female	80.0	1	0	Yes	P
5106	44873	Female	81.0	0	0	Yes	emp

```
df.head()
```

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_
0	9046	Male	67.0	0	1	Yes	Private	U
1	51676	Female	61.0	0	0	Yes	Self-employed	R
2	31112	Male	80.0	0	1	Yes	Private	R
3	60182	Female	49.0	0	0	Yes	Private	U

```
df.tail()
```

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residen
5105	18234	Female	80.0	1	0	Yes	Private	
5106	44873	Female	81.0	0	0	Yes	Self-employed	
5107	19723	Female	35.0	0	0	Yes	Self-employed	
5108	37544	Male	51.0	0	0	Yes	Private	

```
df.info
```

```
<bound method DataFrame.info of          id  gender   age  hypertension  heart_disease ever_married  \
0      9046    Male  67.0             0              1          Yes
1     51676  Female  61.0             0              0          Yes
2     31112    Male  80.0             0              1          Yes
3     60182  Female  49.0             0              0          Yes
4      1665  Female  79.0             1              0          Yes
...     ...     ...   ...           ...            ...          ...
5105  18234  Female  80.0             1              0          Yes
5106  44873  Female  81.0             0              0          Yes
5107  19723  Female  35.0             0              0          Yes
5108  37544    Male  51.0             0              0          Yes
5109  44679  Female  44.0             0              0          Yes

          work_type Residence_type  avg_glucose_level   bmi   smoking_status  \
0           Private          Urban             228.69  36.6  formerly smoked
1     Self-employed          Rural             202.21   NaN     never smoked
```

```
2          Private          Rural            105.92  32.5     never smoked
3          Private          Urban            171.23  34.4           smokes
4     Self-employed         Rural            174.12  24.0     never smoked
...             ...           ...               ...   ...             ...
5105       Private          Urban             83.75   NaN     never smoked
5106  Self-employed         Urban            125.20  40.0     never smoked
5107  Self-employed         Rural             82.99  30.6     never smoked
5108       Private          Rural            166.29  25.6  formerly smoked
5109      Govt_job          Urban             85.28  26.2          Unknown

      stroke
0          1
1          1
2          1
3          1
4          1
...      ...
5105       0
5106       0
5107       0
5108       0
5109       0

[5110 rows x 12 columns]>
```

```
df.dtypes
```

```
id                   int64
gender              object
age                float64
hypertension         int64
heart_disease        int64
ever_married        object
work_type           object
Residence_type      object
avg_glucose_level  float64
bmi                float64
smoking_status      object
stroke               int64
dtype: object
```

TOTAL NUMBER OF MISSING VALUES

```
df.isna().sum()
```

```
id                   0
gender               0
age                  0
hypertension         0
heart_disease        0
ever_married         0
work_type            0
Residence_type       0
avg_glucose_level    0
bmi                201
smoking_status       0
stroke               0
dtype: int64
```

```
#Here there is missing values in 'bmi' ; for filling it we are using 'fillna'
x=df['bmi'].mean()
x
df['bmi'].fillna(x,inplace=True)
print(df)
```

```
           id  gender   age  hypertension  heart_disease ever_married  \
0        9046    Male  67.0             0              1          Yes
1       51676  Female  61.0             0              0          Yes
2       31112    Male  80.0             0              1          Yes
3       60182  Female  49.0             0              0          Yes
4        1665  Female  79.0             1              0          Yes
...       ...     ...   ...           ...            ...          ...
5105    18234  Female  80.0             1              0          Yes
5106    44873  Female  81.0             0              0          Yes
5107    19723  Female  35.0             0              0          Yes
5108    37544    Male  51.0             0              0          Yes
5109    44679  Female  44.0             0              0          Yes

          work_type Residence_type  avg_glucose_level        bmi  \
```

```
0          Private          Urban            228.69  36.600000
1    Self-employed          Rural            202.21  28.893237
2          Private          Rural            105.92  32.500000
3          Private          Urban            171.23  34.400000
4    Self-employed          Rural            174.12  24.000000
...            ...            ...               ...        ...
5105       Private          Urban             83.75  28.893237
5106 Self-employed          Urban            125.20  40.000000
5107 Self-employed          Rural             82.99  30.600000
5108       Private          Rural            166.29  25.600000
5109      Govt_job          Urban             85.28  26.200000

      smoking_status  stroke
0     formerly smoked       1
1        never smoked       1
2        never smoked       1
3              smokes       1
4        never smoked       1
...               ...     ...
5105     never smoked       0
5106     never smoked       0
5107     never smoked       0
5108  formerly smoked       0
5109          Unknown       0

[5110 rows x 12 columns]
```

```
#After filling when we check the missing values it will be 0
df.isna().sum()
```

```
id                   0
gender               0
age                  0
hypertension         0
heart_disease        0
ever_married         0
work_type            0
Residence_type       0
avg_glucose_level    0
bmi                  0
smoking_status       0
stroke               0
dtype: int64
```

IMPORTING LABEL ENCODER

```
from sklearn.preprocessing import LabelEncoder
encoder=LabelEncoder()
columns=['gender','ever_married','work_type','Residence_type','smoking_status']
for i in df[columns]:
  df[i]=encoder.fit_transform(df[i])
df
```

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_st
0	9046	1	67.0	0	1	1	2	1	228.69	36.600000	
1	51676	0	61.0	0	0	1	3	0	202.21	28.893237	
2	31112	1	80.0	0	1	1	2	0	105.92	32.500000	
3	60182	0	49.0	0	0	1	2	1	171.23	34.400000	
4	1665	0	79.0	1	0	1	3	0	174.12	24.000000	
...	...	...	...	...	...	...	...	...	...	...	
5105	18234	0	80.0	1	0	1	2	1	83.75	28.893237	
5106	44873	0	81.0	0	0	1	3	1	125.20	40.000000	
5107	19723	0	35.0	0	0	1	3	0	82.99	30.600000	
5108	37544	1	51.0	0	0	1	2	0	166.29	25.600000	
5109	44679	0	44.0	0	0	1	0	1	85.28	26.200000	

5110 rows × 12 columns

SEPARATING X AND Y

```
x=df.iloc[:,:-1].values
x
```

```
array([[9.04600000e+03, 1.00000000e+00, 6.70000000e+01, ...,
        2.28690000e+02, 3.66000000e+01, 1.00000000e+00],
       [5.16760000e+04, 0.00000000e+00, 6.10000000e+01, ...,
        2.02210000e+02, 2.88932369e+01, 2.00000000e+00],
       [3.11120000e+04, 1.00000000e+00, 8.00000000e+01, ...,
        1.05920000e+02, 3.25000000e+01, 2.00000000e+00],
       ...,
       [1.97230000e+04, 0.00000000e+00, 3.50000000e+01, ...,
        8.29900000e+01, 3.06000000e+01, 2.00000000e+00],
       [3.75440000e+04, 1.00000000e+00, 5.10000000e+01, ...,
        1.66290000e+02, 2.56000000e+01, 1.00000000e+00],
       [4.46790000e+04, 0.00000000e+00, 4.40000000e+01, ...,
        8.52800000e+01, 2.62000000e+01, 0.00000000e+00]])
```

```
y=df.iloc[:,-1]
y
```

```
0       1
1       1
2       1
3       1
4       1
       ..
5105    0
5106    0
5107    0
5108    0
5109    0
Name: stroke, Length: 5110, dtype: int64
```

```
x.ndim
```

```
2
```

```
y.ndim
```

```
1
```

SPLITTING DATA INTO TRAINING AND TESTING DATA

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
x_train
```

```
array([[2.4257e+04, 1.0000e+00, 4.0000e+00, ..., 9.0420e+01, 1.6200e+01,
        0.0000e+00],
       [5.6179e+04, 1.0000e+00, 2.9000e+01, ..., 2.0758e+02, 2.2800e+01,
        3.0000e+00],
       [3.6388e+04, 1.0000e+00, 4.4000e+01, ..., 9.1280e+01, 2.6500e+01,
        2.0000e+00],
       ...,
       [3.1481e+04, 0.0000e+00, 1.1600e+00, ..., 9.7280e+01, 1.7800e+01,
        0.0000e+00],
       [6.1827e+04, 1.0000e+00, 8.0000e+01, ..., 1.9608e+02, 3.1000e+01,
        1.0000e+00],
       [2.8933e+04, 0.0000e+00, 4.6000e+01, ..., 1.0015e+02, 5.0300e+01,
        3.0000e+00]])
```

```
x_test
```

```
array([[4.0041e+04, 1.0000e+00, 3.1000e+01, ..., 6.4850e+01, 2.3000e+01,
        0.0000e+00],
       [5.5244e+04, 1.0000e+00, 4.0000e+01, ..., 6.5290e+01, 2.8300e+01,
        2.0000e+00],
       [7.0992e+04, 0.0000e+00, 8.0000e+00, ..., 7.4420e+01, 2.2500e+01,
        0.0000e+00],
       ...,
       [3.0753e+04, 1.0000e+00, 4.2000e+01, ..., 9.3790e+01, 2.7200e+01,
        2.0000e+00],
       [6.6270e+04, 0.0000e+00, 5.7000e+01, ..., 6.9400e+01, 2.4000e+01,
        0.0000e+00],
```

```
       [1.0243e+04, 0.0000e+00, 6.0000e+01, ..., 7.3040e+01, 2.5300e+01,
        2.0000e+00]])
```

NORMALIZATION OF DATA USING MINMAX SCALER

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
scaler.fit(x_train)
x_train=scaler.fit_transform(x_train)
x_test=scaler.fit_transform(x_test)
x_train
```

```
array([[0.33185567, 1.        , 0.04692082, ..., 0.16295818, 0.06758305,
        0.        ],
       [0.76996555, 1.        , 0.35239492, ..., 0.70381313, 0.14318442,
        1.        ],
       [0.49834621, 1.        , 0.53567937, ..., 0.16692826, 0.18556701,
        0.66666667],
       ...,
       [0.43100065, 0.        , 0.01221896, ..., 0.19462653, 0.08591065,
        0.        ],
       [0.84748089, 1.        , 0.97556207, ..., 0.65072477, 0.2371134 ,
        0.33333333],
       [0.39603091, 0.        , 0.5601173 , ..., 0.20787554, 0.45819015,
        1.        ]])
```

```
x_test
```

```
array([[0.54873161, 0.5       , 0.37744141, ..., 0.04551476, 0.19039735,
        0.        ],
       [0.75742642, 0.5       , 0.48730469, ..., 0.04759652, 0.2781457 ,
        0.66666667],
       [0.97360257, 0.        , 0.09667969, ..., 0.09079296, 0.18211921,
        0.        ],
       ...,
       [0.42123325, 0.5       , 0.51171875, ..., 0.18243755, 0.25993377,
        0.66666667],
       [0.90878267, 0.        , 0.69482422, ..., 0.06704201, 0.20695364,
        0.        ],
       [0.13968812, 0.        , 0.73144531, ..., 0.08426382, 0.22847682,
        0.66666667]])
```

CREATION OF SVM MODEL

```
from sklearn.svm import SVC
model=SVC()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
y_pred
```

```
array([0, 0, 0, ..., 0, 0, 0])
```

PERFORMANCE EVALUATION

```
from sklearn.metrics import confusion_matrix,accuracy_score
result=confusion_matrix(y_test,y_pred)
result
```

```
array([[1444,    0],
       [  89,    0]])
```

```
score=accuracy_score(y_test,y_pred)
score
```

```
0.9419439008480104
```

✓ 0s completed at 8:52 AM