In [84]:
```
import pandas as pd
df = pd.read_csv('SmokingDataSet.csv')
df.head()
```

...

In [85]:
```
obj_list = list(df.select_dtypes(include='object'))
obj_list
```

Out[85]:
```
['gender', 'ever_married', 'work_type', 'Residence_type', 'smoking
_status']
```

In [86]:
```
from sklearn import preprocessing
for i in obj_list:
    Encoder = preprocessing.LabelEncoder()
    df[i]= Encoder.fit_transform(df[i])
```

In [87]:
```
df.head()
```

Out[87]:

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	av
0	1	67.0	0	1	1	1	1	
1	1	80.0	0	1	1	1	0	
2	0	49.0	0	0	1	1	1	
3	0	79.0	1	0	1	2	0	
4	1	81.0	0	0	1	1	1	

In [88]:
```
x = df.drop(columns=['stroke'],axis=1)
y = df['stroke']
```

In [89]:
```
x,y
```

Out[89]:
```
(      gender   age  hypertension  heart_disease  ever_married  wo
rk_type  \
 0          1  67.0             0              1             1
1
 1          1  80.0             0              1             1
1
 2          0  49.0             0              0             1
1
 3          0  79.0             1              0             1
2
 4          1  81.0             0              0             1
1
 ...      ...   ...           ...            ...           ...
...
 4976       1  41.0             0              0             0
1
 4977       1  40.0             0              0             1
 4977        1  40.0             0              0             1
1
 4978        0  45.0             1              0             1
0
 4979        1  40.0             0              0             1
1
 4980        0  80.0             1              0             1
1

      Residence_type  avg_glucose_level   bmi  smoking_status
 0                 1             228.69  36.6               1
 1                 0             105.92  32.5               2
 2                 1             171.23  34.4               3
 3                 0             174.12  24.0               2
 4                 1             186.21  29.0               1
 ...             ...                ...   ...             ...
 4976              0              70.15  29.8               1
 4977              1             191.15  31.1               3
 4978              0              95.02  31.8               3
 4979              0              83.94  30.0               3
 4980              1              83.75  29.1               2

 [4981 rows x 10 columns],
 0       1
 1       1
 2       1
 3       1
 4       1
        ..
 4976    0
 4977    0
 4978    0
 4979    0
 4980    0
 Name: stroke, Length: 4981, dtype: int64)
```

In [90]:
```
from imblearn.over_sampling import RandomOverSampler
over_sampler = RandomOverSampler(sampling_strategy='minority')
x,y = over_sampler.fit_resample(x,y)
```

In [91]:
```
x,y
```

Out[91]:
```
(      gender   age  hypertension  heart_disease  ever_married  wo
rk_type  \
 0          1  67.0             0              1             1
1
 1          1  80.0             0              1             1
1
 2          0  49.0             0              0             1
1
 3          0  79.0             1              0             1
2
 4          1  81.0             0              0             1
1
 ...      ...   ...           ...            ...           ...
...
 9461       0  70.0             0              1             1
1
 9462       1  54.0             0              0             1
1
 9463       0  75.0             0              0             1
1
 9464       0  61.0             0              0             1
2
 9465       0  66.0             1              0             1
0

       Residence_type  avg_glucose_level   bmi  smoking_status
 0                  1             228.69  36.6               1
 1                  0             105.92  32.5               2
 2                  1             171.23  34.4               3
 3                  0             174.12  24.0               2
 4                  1             186.21  29.0               1
 ...              ...                ...   ...             ...
 9461               0             239.07  26.1               2
 9462               1              71.22  28.5               2
 9463               0             199.20  26.6               0
 9464               0             202.21  31.6               2
 9465               0             116.55  31.1               1

 [9466 rows x 10 columns],
 0       1
 1       1
 2       1
 3       1
 4       1
        ..
 9461    1
 9462    1
 9463    1
 9464    1
 9465    1
 Name: stroke, Length: 9466, dtype: int64)
```

In [92]:

```python
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 Forest Classifier

In [93]:

```python
from sklearn.ensemble import RandomForestClassifier
RF_model = RandomForestClassifier(n_estimators=1000)
RF_model.fit(x_train,y_train)
y_pred = RF_model.predict(x_test)
```

```
In [94]: from sklearn.metrics import precision_score,recall_score,roc_auc_sc
         print("Precision Score for RandomForest model",precision_score(y_te
```

```
Precision Score for RandomForest model 0.9809286898839138
```

DecisionTreeClassifier

```
In [95]: from sklearn.tree import DecisionTreeClassifier
         DT_model = DecisionTreeClassifier()
         DT_model.fit(x_train,y_train)
         tree_pred = DT_model.predict(x_test)
```

```
In [96]: print("Precision Score for Decision model",precision_score(y_test,t
```

```
Precision Score for Decision model 0.9441340782122905
```

```
In [97]: print("Recall Score for Decision model",recall_score(y_test,tree_pr
         print("Recall Score for RandomForest model",recall_score(y_test,y_p
```

```
Recall Score for Decision model 1.0
Recall Score for RandomForest model 1.0
```

```
In [98]: print("ROC-AUC Score for Decision model",roc_auc_score(y_test,tree_
         print("ROC-AUC Score for RandomForest model",roc_auc_score(y_test,y
```

```
ROC-AUC Score for Decision model 0.9704391891891893
ROC-AUC Score for RandomForest model 0.9902871621621622
```

Confusion Matrix

```
In [100]: from sklearn.metrics import confusion_matrix
          DT_model.fit(x_train,y_train)
          DT_model = DT_model.predict(x_test)
          cm_log = confusion_matrix(y_test,tree_pred)
          cm_log
```

```
Out[100]: array([[1114,   70],
                 [   0, 1183]])
```

In [101]:

```
from prettytable import PrettyTable
Comparision_table    = PrettyTable(["Model", "Precision Score", "Re
Comparision_table.add_row(["Decision Model","0.94", "1.0", "0.96"])
Comparision_table.add_row(["RandomForest Model","0.98", "1.0", "0.9
print(Comparision_table)
```

```
+--------------------+-----------------+--------------+-----------
----+
|       Model        | Precision Score | Recall Score | ROC-AUC Sc
ore |
+--------------------+-----------------+--------------+-----------
----+
|   Decision Model   |       0.94      |     1.0      |      0.96
|
| RandomForest Model |       0.98      |     1.0      |      0.99
|
+--------------------+-----------------+--------------+-----------
----+
```