Data Set: Voice Data Set This database was created to identify a voice as male or female, based upon acoustic properties of the voice and speech. The dataset consists of 3168 recorded voice samples. The voice samples are pre-processed by acoustic analysis in R using the seewave and tuneR packages. The following acoustic properties of each voice are measured and included within the CSV: meanfreq: mean frequency (in kHz) sd: standard deviation of frequency median: median frequency (in kHz) Q25: first quantile (in kHz) Q75: third quantile (in kHz) IQR: interquantile range (in kHz) skew: skewness (see note in specprop description) kurt: kurtosis (see note in specprop description) sp.ent: spectral entropy sfm: spectral flatness mode: mode frequency centroid: frequency centroid (see specprop) peakf: peak frequency (frequency with highest energy) meanfun: average of fundamental frequency measured across acoustic signal minfun: minimum fundamental frequency measured across acoustic signal maxfun: maximum fundamental frequency measured across acoustic signal meandom: average of dominant frequency measured across acoustic signal mindom: minimum of dominant frequency measured across acoustic signal maxdom: maximum of dominant frequency measured across acoustic signal dfrange: range of dominant frequency measured across acoustic signal modindx: modulation index. Calculated as the accumulated absolute difference between adjacent measurements of fundamental frequencies divided by the frequency range label: male or female

our problem is to identify the voices either male or female voice.

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
df = pd.read_csv("voice-classification.csv")
```

In [3]:
```
df.info() #All columns are float except label column which is dependdent variable is ob
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3168 entries, 0 to 3167
Data columns (total 21 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   meanfreq  3168 non-null   float64
 1   sd        3168 non-null   float64
 2   median    3168 non-null   float64
 3   Q25       3168 non-null   float64
 4   Q75       3168 non-null   float64
 5   IQR       3168 non-null   float64
 6   skew      3168 non-null   float64
 7   kurt      3168 non-null   float64
 8   sp.ent    3168 non-null   float64
 9   sfm       3168 non-null   float64
 10  mode      3168 non-null   float64
 11  centroid  3168 non-null   float64
 12  meanfun   3168 non-null   float64
 13  minfun    3168 non-null   float64
 14  maxfun    3168 non-null   float64
 15  meandom   3168 non-null   float64
 16  mindom    3168 non-null   float64
 17  maxdom    3168 non-null   float64
 18  dfrange   3168 non-null   float64
 19  modindx   3168 non-null   float64
```

```
 20  label     3168 non-null   object
dtypes: float64(20), object(1)
memory usage: 519.9+ KB
```

Pick up the IV and DV

In [4]:

```
x = df.iloc[:,0:-1] #iloc location command
x.info() #All columns except the label column
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3168 entries, 0 to 3167
Data columns (total 20 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   meanfreq  3168 non-null   float64
 1   sd        3168 non-null   float64
 2   median    3168 non-null   float64
 3   Q25       3168 non-null   float64
 4   Q75       3168 non-null   float64
 5   IQR       3168 non-null   float64
 6   skew      3168 non-null   float64
 7   kurt      3168 non-null   float64
 8   sp.ent    3168 non-null   float64
 9   sfm       3168 non-null   float64
 10  mode      3168 non-null   float64
 11  centroid  3168 non-null   float64
 12  meanfun   3168 non-null   float64
 13  minfun    3168 non-null   float64
 14  maxfun    3168 non-null   float64
 15  meandom   3168 non-null   float64
 16  mindom    3168 non-null   float64
 17  maxdom    3168 non-null   float64
 18  dfrange   3168 non-null   float64
 19  modindx   3168 non-null   float64
dtypes: float64(20)
memory usage: 495.1 KB
```

In [5]:

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

Out[5]:

```
0         male
1         male
2         male
3         male
4         male
         ...
3163    female
3164    female
3165    female
3166    female
3167    female
Name: label, Length: 3168, dtype: object
```

In [6]:

```
#Now will convert the object value in label column to the numeric

from sklearn.preprocessing import LabelEncoder
gender_encoder = LabelEncoder()
```

In [7]:
```
y = gender_encoder.fit_transform(y)
y #The values of label column convert to 1 and 0
```

Out[7]:
```
array([1, 1, 1, ..., 0, 0, 0])
```

In [8]:
```
# SVM is distance based ML algo , we awill apply standard scaler

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
```

In [9]:
```
x = scaler.fit_transform(x)
x
```

Out[9]:
```
array([[-4.04924806,  0.4273553 , -4.22490077, ..., -1.43142165,
        -1.41913712, -1.45477229],
       [-3.84105325,  0.6116695 , -3.99929342, ..., -1.41810716,
        -1.4058184 , -1.01410294],
       [-3.46306647,  1.60384791, -4.09585052, ..., -1.42920257,
        -1.41691733, -1.06534356],
       ...,
       [-1.29877326,  2.32272355, -0.05197279, ..., -0.5992661 ,
        -0.58671739,  0.17588664],
       [-1.2452018 ,  2.012196  , -0.01772849, ..., -0.41286326,
        -0.40025537,  1.14916112],
       [-0.51474626,  2.14765111, -0.07087873, ..., -1.27608595,
        -1.2637521 ,  1.47567886]])
```

In [10]:
```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test  = train_test_split(x,y, random_state=123)
```

Applying Support Vector Classifier

In [11]:
```
from sklearn.svm import SVC

from sklearn import metrics #  #To show us how much progress we have made
```

In [12]:
```
svc_model = SVC()
```

In [13]:
```
svc_model.fit(x_train, y_train)
```

Out[13]:
```
SVC()
```

In [14]:
```
y_pred = svc_model.predict(x_test)
y_pred
```

Out[14]:
```
array([1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0,
       1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1,
       1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0,
       0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0,
       0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0,
       1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1,
       1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0,
       1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0,
       1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0])
```

In [15]:
```
y_test
```

Out[15]:
```
array([0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0,
       1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1,
       1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1,
```

```
0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0,
0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0,
0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0,
0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0,
1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1,
1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0,
0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0,
1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0,
1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0,
1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0])
```

Model Evalution

In [16]:
```
acuuracy_svm = metrics.accuracy_score(y_test, y_pred)
```

In [17]:
```
acuuracy_svm
```

Out[17]:
```
0.9810606060606061
```

In [18]:
```
# print the confusion matrix

print(metrics.confusion_matrix(y_test, y_pred))
```

```
[[393   7]
 [  8 384]]
```

In [19]:
```
#Checking if my model is overfitted

y_pred_train = svc_model.predict(x_train)
print(metrics.accuracy_score(y_train, y_pred_train))
```

```
0.9852693602693603
```

Let us do hyperparameter tuning

In [20]:
```
from sklearn.model_selection import GridSearchCV
```

In [28]:
```
param_grid = {'C':[.1,1,10, 100], 'gamma': [100, 10, 1, .1, .01, .001]}
param_grid
```

Out[28]:
```
{'C': [0.1, 1, 10, 100], 'gamma': [100, 10, 1, 0.1, 0.01, 0.001]}
```

In [29]:
```
grid = GridSearchCV(SVC(), param_grid, refit= True, verbose= 2)
```

In [30]:
```
grid
```

Out[30]:
```
GridSearchCV(estimator=SVC(),
             param_grid={'C': [0.1, 1, 10, 100],
```

```
                    'gamma': [100, 10, 1, 0.1, 0.01, 0.001]},
              verbose=2)
```

In [31]:

```
grid.fit(x_train, y_train)
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
[CV] END .....................................C=0.1, gamma=100; total time=   0.9s
[CV] END .....................................C=0.1, gamma=100; total time=   0.8s
[CV] END .....................................C=0.1, gamma=100; total time=   0.7s
[CV] END .....................................C=0.1, gamma=100; total time=   0.8s
[CV] END .....................................C=0.1, gamma=100; total time=   0.8s
[CV] END ......................................C=0.1, gamma=10; total time=   0.6s
[CV] END ......................................C=0.1, gamma=10; total time=   0.6s
[CV] END ......................................C=0.1, gamma=10; total time=   0.7s
[CV] END ......................................C=0.1, gamma=10; total time=   0.6s
[CV] END ......................................C=0.1, gamma=10; total time=   0.6s
[CV] END .......................................C=0.1, gamma=1; total time=   0.5s
[CV] END .......................................C=0.1, gamma=1; total time=   0.5s
[CV] END .......................................C=0.1, gamma=1; total time=   0.5s
[CV] END .......................................C=0.1, gamma=1; total time=   0.5s
[CV] END .......................................C=0.1, gamma=1; total time=   0.5s
[CV] END .....................................C=0.1, gamma=0.1; total time=   0.2s
[CV] END .....................................C=0.1, gamma=0.1; total time=   0.1s
[CV] END .....................................C=0.1, gamma=0.1; total time=   0.2s
[CV] END .....................................C=0.1, gamma=0.1; total time=   0.1s
[CV] END .....................................C=0.1, gamma=0.1; total time=   0.2s
[CV] END ....................................C=0.1, gamma=0.01; total time=   0.2s
[CV] END ....................................C=0.1, gamma=0.01; total time=   0.2s
[CV] END ....................................C=0.1, gamma=0.01; total time=   0.2s
[CV] END ....................................C=0.1, gamma=0.01; total time=   0.2s
[CV] END ....................................C=0.1, gamma=0.01; total time=   0.2s
[CV] END ...................................C=0.1, gamma=0.001; total time=   0.4s
[CV] END ...................................C=0.1, gamma=0.001; total time=   0.4s
[CV] END ...................................C=0.1, gamma=0.001; total time=   0.4s
[CV] END ...................................C=0.1, gamma=0.001; total time=   0.4s
[CV] END ...................................C=0.1, gamma=0.001; total time=   0.5s
[CV] END ......................................C=1, gamma=100; total time=   0.9s
[CV] END ......................................C=1, gamma=100; total time=   0.7s
[CV] END ......................................C=1, gamma=100; total time=   0.8s
[CV] END ......................................C=1, gamma=100; total time=   0.9s
[CV] END ......................................C=1, gamma=100; total time=   0.9s
[CV] END .......................................C=1, gamma=10; total time=   0.6s
[CV] END .......................................C=1, gamma=10; total time=   0.6s
[CV] END .......................................C=1, gamma=10; total time=   0.6s
[CV] END .......................................C=1, gamma=10; total time=   0.8s
[CV] END .......................................C=1, gamma=10; total time=   0.6s
[CV] END ......................................C=1, gamma=1; total time=   0.5s
[CV] END ......................................C=1, gamma=1; total time=   0.6s
[CV] END ......................................C=1, gamma=1; total time=   0.5s
[CV] END ......................................C=1, gamma=1; total time=   0.5s
[CV] END ......................................C=1, gamma=1; total time=   0.5s
[CV] END ....................................C=1, gamma=0.1; total time=   0.0s
[CV] END ....................................C=1, gamma=0.1; total time=   0.0s
[CV] END ....................................C=1, gamma=0.1; total time=   0.0s
[CV] END ....................................C=1, gamma=0.1; total time=   0.0s
[CV] END ....................................C=1, gamma=0.1; total time=   0.0s
[CV] END ...................................C=1, gamma=0.01; total time=   0.0s
[CV] END ...................................C=1, gamma=0.01; total time=   0.0s
[CV] END ...................................C=1, gamma=0.01; total time=   0.0s
[CV] END ...................................C=1, gamma=0.01; total time=   0.0s
[CV] END ...................................C=1, gamma=0.01; total time=   0.1s
[CV] END ..................................C=1, gamma=0.001; total time=   0.2s
[CV] END ..................................C=1, gamma=0.001; total time=   0.2s
[CV] END ..................................C=1, gamma=0.001; total time=   0.2s
```

```
[CV] END ..................................C=1, gamma=0.001; total time=   0.2s
[CV] END ..................................C=1, gamma=0.001; total time=   0.2s
[CV] END ...................................C=10, gamma=100; total time=   0.9s
[CV] END ...................................C=10, gamma=100; total time=   0.9s
[CV] END ...................................C=10, gamma=100; total time=   0.8s
[CV] END ...................................C=10, gamma=100; total time=   0.7s
[CV] END ...................................C=10, gamma=100; total time=   0.9s
[CV] END ....................................C=10, gamma=10; total time=   0.6s
[CV] END ....................................C=10, gamma=10; total time=   0.6s
[CV] END ....................................C=10, gamma=10; total time=   0.6s
[CV] END ....................................C=10, gamma=10; total time=   0.7s
[CV] END ....................................C=10, gamma=10; total time=   0.7s
[CV] END .....................................C=10, gamma=1; total time=   0.5s
[CV] END .....................................C=10, gamma=1; total time=   0.5s
[CV] END .....................................C=10, gamma=1; total time=   0.5s
[CV] END .....................................C=10, gamma=1; total time=   0.5s
[CV] END .....................................C=10, gamma=1; total time=   0.5s
[CV] END ...................................C=10, gamma=0.1; total time=   0.0s
[CV] END ...................................C=10, gamma=0.1; total time=   0.0s
[CV] END ...................................C=10, gamma=0.1; total time=   0.0s
[CV] END ...................................C=10, gamma=0.1; total time=   0.0s
[CV] END ...................................C=10, gamma=0.1; total time=   0.0s
[CV] END ..................................C=10, gamma=0.01; total time=   0.0s
[CV] END ..................................C=10, gamma=0.01; total time=   0.0s
[CV] END ..................................C=10, gamma=0.01; total time=   0.0s
[CV] END ..................................C=10, gamma=0.01; total time=   0.0s
[CV] END ..................................C=10, gamma=0.01; total time=   0.0s
[CV] END .................................C=10, gamma=0.001; total time=   0.0s
[CV] END .................................C=10, gamma=0.001; total time=   0.0s
[CV] END .................................C=10, gamma=0.001; total time=   0.0s
[CV] END .................................C=10, gamma=0.001; total time=   0.0s
[CV] END .................................C=10, gamma=0.001; total time=   0.0s
[CV] END ..................................C=100, gamma=100; total time=   0.9s
[CV] END ..................................C=100, gamma=100; total time=   0.8s
[CV] END ..................................C=100, gamma=100; total time=   0.8s
[CV] END ..................................C=100, gamma=100; total time=   0.9s
[CV] END ..................................C=100, gamma=100; total time=   0.8s
[CV] END ...................................C=100, gamma=10; total time=   0.6s
[CV] END ...................................C=100, gamma=10; total time=   0.6s
[CV] END ...................................C=100, gamma=10; total time=   0.6s
[CV] END ...................................C=100, gamma=10; total time=   0.6s
[CV] END ...................................C=100, gamma=10; total time=   0.6s
[CV] END ....................................C=100, gamma=1; total time=   0.5s
[CV] END ....................................C=100, gamma=1; total time=   0.5s
[CV] END ....................................C=100, gamma=1; total time=   0.6s
[CV] END ....................................C=100, gamma=1; total time=   0.5s
[CV] END ....................................C=100, gamma=1; total time=   0.5s
[CV] END ..................................C=100, gamma=0.1; total time=   0.0s
[CV] END ..................................C=100, gamma=0.1; total time=   0.0s
[CV] END ..................................C=100, gamma=0.1; total time=   0.0s
[CV] END ..................................C=100, gamma=0.1; total time=   0.1s
[CV] END ..................................C=100, gamma=0.1; total time=   0.0s
[CV] END .................................C=100, gamma=0.01; total time=   0.0s
[CV] END .................................C=100, gamma=0.01; total time=   0.0s
[CV] END .................................C=100, gamma=0.01; total time=   0.0s
[CV] END .................................C=100, gamma=0.01; total time=   0.0s
[CV] END .................................C=100, gamma=0.01; total time=   0.1s
[CV] END ................................C=100, gamma=0.001; total time=   0.0s
[CV] END ................................C=100, gamma=0.001; total time=   0.0s
[CV] END ................................C=100, gamma=0.001; total time=   0.0s
[CV] END ................................C=100, gamma=0.001; total time=   0.0s
[CV] END ................................C=100, gamma=0.001; total time=   0.0s
```

Out[31]:
```
GridSearchCV(estimator=SVC(),
             param_grid={'C': [0.1, 1, 10, 100],
```

```
                    'gamma': [100, 10, 1, 0.1, 0.01, 0.001]},
             verbose=2)
```

In [33]:
```
grid_prediction = grid.predict(x_test)
grid_prediction
```

Out[33]:
```
array([1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1,
       0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0,
       1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1,
       1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1,
       0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0,
       0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0,
       1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1,
       1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0,
       1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0,
       1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0])
```

In [34]:
```
print(metrics.accuracy_score(y_test,grid_prediction))
```

```
0.9797979797979798
```

In [35]:
```
grid.best_params_ #Best parameters in Gridsearchcv for the model
```

Out[35]:
```
{'C': 1, 'gamma': 0.1}
```

Create the final model with the help of the final and best parameters found ot in the Gridsearch CV

In [36]:
```
svc_grid_model = SVC(C = 1.0, gamma= 0.1)
```

In [37]:
```
svc_grid_model
```

Out[37]:
```
SVC(gamma=0.1)
```

In [38]:
```
svc_grid_model.fit(x_train, y_train)
```

Out[38]:
```
SVC(gamma=0.1)
```

In [39]:
```
y_grid_pred = svc_grid_model.predict(x_test)
y_grid_pred
```

Out[39]:
```
array([1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1,
       0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1,
       0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0,
       1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1,
       0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0,
       1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1,
       1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1,
       1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1,
       1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1,
       0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1,
       0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0,
       0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1,
       1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0,
       1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1,
       1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0,
       1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0,
       1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0])
```

In [40]:
```
print(metrics.accuracy_score(y_test,y_grid_pred))
```

```
0.9797979797979798
```

In [ ]:
```

```