In [1]:
```
%matplotlib inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv("C:\\Users\\Abdul\\voice.csv")

print(df.columns)
```

```
Index(['meanfreq', 'sd', 'median', 'Q25', 'Q75', 'IQR', 'skew', 'kurt',
       'sp.ent', 'sfm', 'mode', 'centroid', 'meanfun', 'minfun', 'maxfun',
       'meandom', 'mindom', 'maxdom', 'dfrange', 'modindx', 'label'],
      dtype='object')
```

In [3]:
```
df.shape
```

Out[3]:
```
(3168, 21)
```

In [4]:
```
df.dtypes
```

Out[4]:
```
meanfreq    float64
sd          float64
median      float64
Q25         float64
Q75         float64
IQR         float64
skew        float64
kurt        float64
sp.ent      float64
sfm         float64
mode        float64
centroid    float64
meanfun     float64
minfun      float64
maxfun      float64
meandom     float64
mindom      float64
maxdom      float64
dfrange     float64
modindx     float64
label        object
dtype: object
```

In [5]:
```
df.head(3)
```

Out[5]:

	meanfreq	sd	median	Q25	Q75	IQR	skew	kurt	sp.ent	sfm
0	0.059781	0.064241	0.032027	0.015071	0.090193	0.075122	12.863462	274.402906	0.893369	0.491918
1	0.066009	0.067310	0.040229	0.019414	0.092666	0.073252	22.423285	634.613855	0.892193	0.513724
2	0.077316	0.083829	0.036718	0.008701	0.131908	0.123207	30.757155	1024.927705	0.846389	0.478905

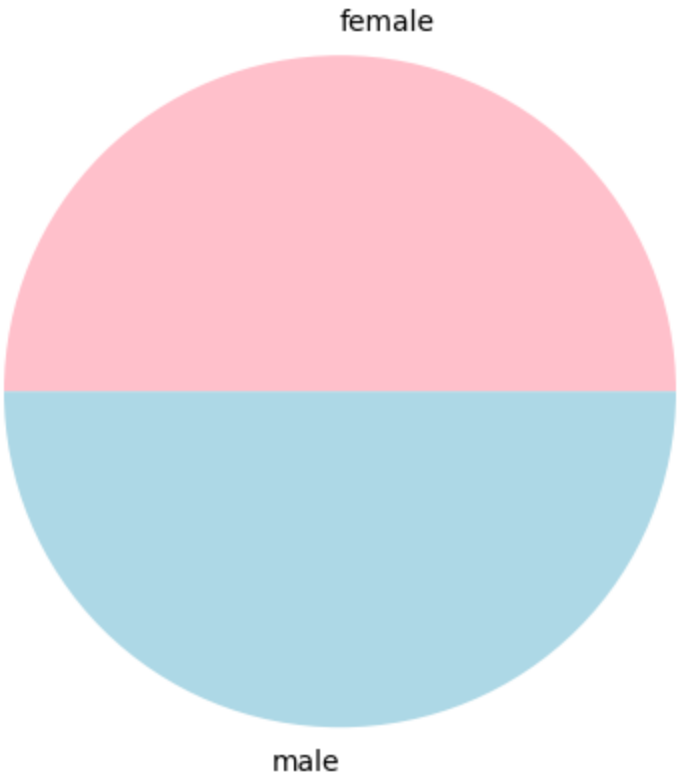
3 rows × 21 columns

In [6]:
```
df.isnull().values.any()
```

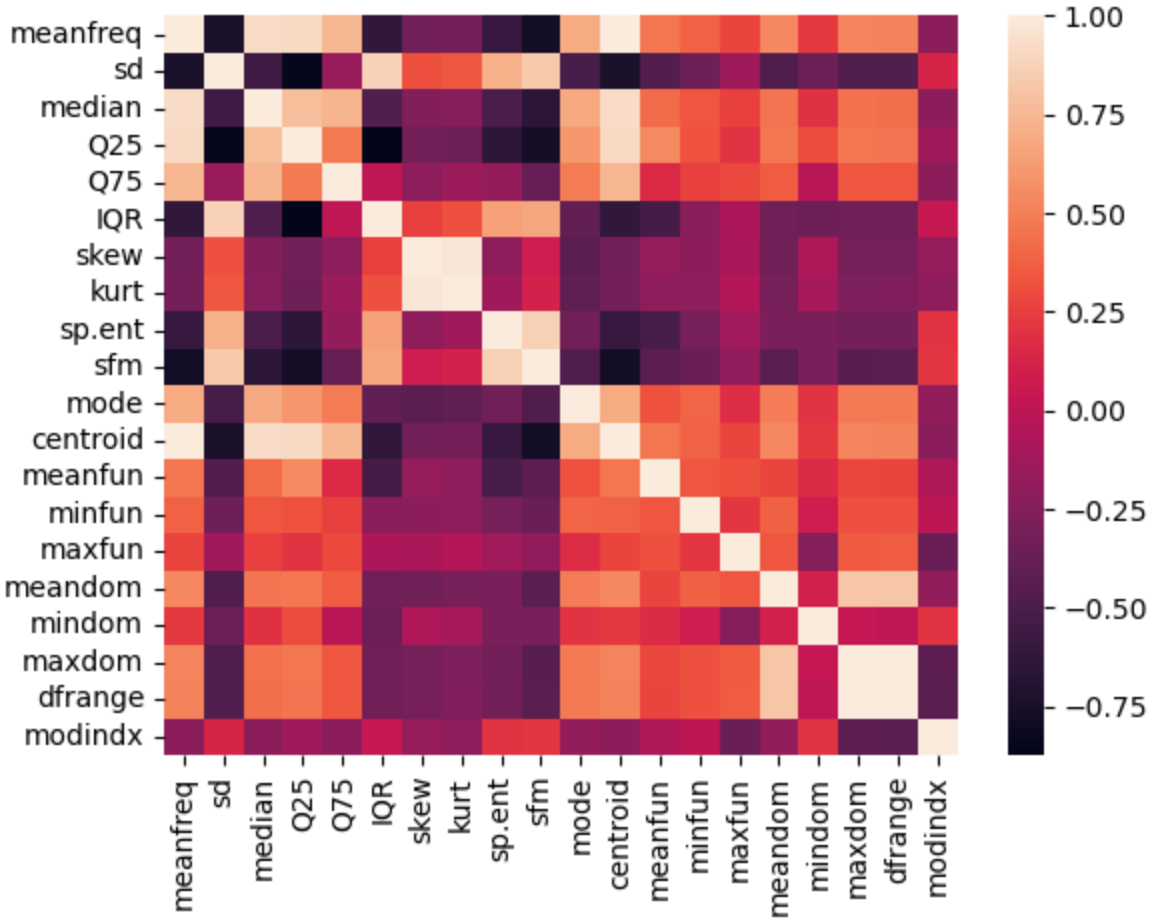
Out[6]:
```
False
```

In [7]:
```
colors =["pink","lightblue"]
data_y =df[df.columns[-1]]
plt.pie(data_y.value_counts(),colors=colors,labels=["female","male"])
plt.axis("equal")
print (df["label"].value_counts())
```

```
male      1584
female    1584
Name: label, dtype: int64
```



In [8]:
```
correlation =df.corr()
sns.heatmap(correlation)
plt.show()
```



In [9]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn import metrics , neighbors
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
```

In [10]:
```
x= df[df.columns[:-1]].values
y= df[df.columns[-1]].values
xtrain,xtest,ytrain,ytest =train_test_split(x,y ,test_size=0.30)
```

In [11]:
```
rand_forest = RandomForestClassifier()
rand_forest.fit(xtrain,ytrain)
y_pred = rand_forest.predict(xtest)
```

In [12]:
```
print(metrics.accuracy_score(ytest,y_pred))
```

```
0.9768664563617245
```

In [13]:
```
print(confusion_matrix(ytest,y_pred))
```

```
[[482   8]
 [ 14 447]]
```

In [14]:
```
from sklearn.model_selection import cross_val_score
CVFirst=GaussianNB()
CVFirst=CVFirst.fit(xtrain,ytrain)
test_result=cross_val_score (CVFirst,x,y,cv=10 ,scoring="accuracy")
print("Accuracy obtained from 10-fold cross validation is :",test_result.mean())
```

```
Accuracy obtained from 10-fold cross validation is : 0.8563410933194906
```

In [15]:
```
#data cleaning
male_funfreq_outlier_index =df[((df["meanfun"] < 0.085)| (df["meanfun"]>0.180)) & (df["]
female_funfreq_outlier_index =df[((df["meanfun"] < 0.165)| (df["meanfun"]>0.255)) & (df[
```

In [16]:
```
index_to_remove =list(male_funfreq_outlier_index)+ list(female_funfreq_outlier_index)
len(index_to_remove)
```

Out[16]:
```
710
```

In [17]:
```
data_x=df[df.columns[0:20]].copy()
data2= data_x.drop(['kurt','centroid', 'dfrange'],axis=1).copy()
data2.head (3)
data2= data2.drop(index_to_remove,axis=0)

data_y = pd.Series(y).drop(index_to_remove,axis=0)
xtrain, xtest, ytrain, ytest = train_test_split(data2, data_y, test_size=0.30 )
CLF1 = RandomForestClassifier()
CLF1.fit(xtrain, ytrain)
y_pred = CLF1.predict (xtest)
print(metrics. accuracy_score(ytest, y_pred))
```

```
0.994579945799458
```

In [18]:
```
clf2 = DecisionTreeClassifier()
clf2.fit(xtrain, ytrain)
y_predict =clf2.predict (xtest)
print (metrics.accuracy_score(ytest,y_predict))
```

```
0.9932249322493225
```

In [19]:
```
clf3 = GaussianNB()
clf3 = clf3.fit(xtrain, ytrain)
y_predd = clf3.predict(xtest)
print(metrics.accuracy_score(ytest,y_predd))
```

```
0.981029810298103
```

In [20]:
```
clf4 = LogisticRegression()
clf4.fit(xtrain, ytrain)
y_predict4 = clf4. predict(xtest)
print(metrics.accuracy_score(ytest,y_predict4))
```

```
0.9092140921409214

C:\Users\Abdul\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: Conv
ergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.o
rg/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (htt
ps://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

In [21]:
```
test_result = cross_val_score(clf3, data2, data_y, cv=10, scoring='accuracy')
print('Accuracy obtained from 10-fold cross validation is:', test_result.mean())
```

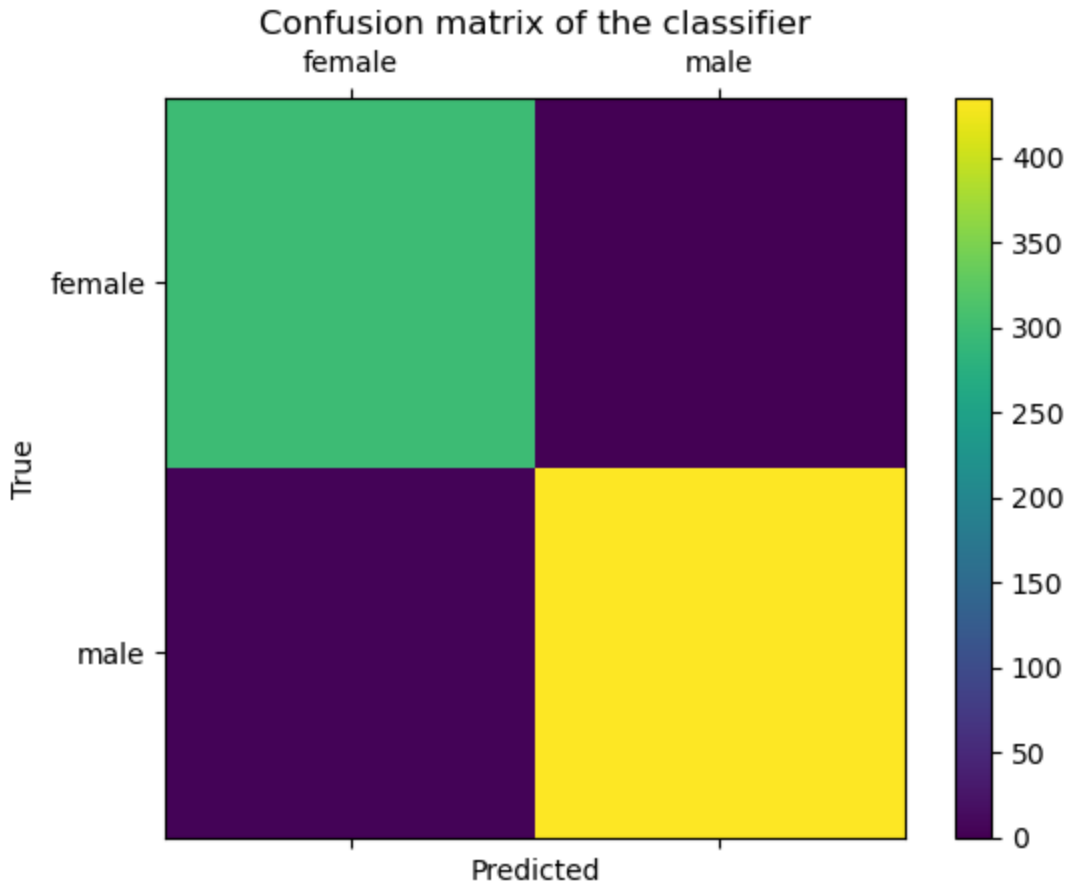
```
Accuracy obtained from 10-fold cross validation is: 0.9637979094076655
```

In [22]:
```
test_result - cross_val_score(clf2, data2, data_y, cv=10, scoring='accuracy')
print('Accuracy obtained from 10-fold cross validation is:',test_result.mean())
```

```
Accuracy obtained from 10-fold cross validation is: 0.9637979094076655
```

In [28]:
```
import pylab as pl
labels = ['female','male']

cm = confusion_matrix(ytest, y_pred)

print(cm)
fig = plt.figure()
ax = fig.add_subplot(111)

cax = ax.matshow(cm)

pl.title('Confusion matrix of the classifier')
fig.colorbar(cax)
ax.set_xticklabels(['']+labels)
ax.set_yticklabels(['']+labels)
pl.xlabel('Predicted')

pl.ylabel('True')

pl.show()
```

```
[[299   0]
 [  4 435]]

C:\Users\Abdul\AppData\Local\Temp\ipykernel_12212\355348629.py:14: UserWarning: FixedF
ormatter should only be used together with FixedLocator
  ax.set_xticklabels(['']+labels)
C:\Users\Abdul\AppData\Local\Temp\ipykernel_12212\355348629.py:15: UserWarning: FixedF
ormatter should only be used together with FixedLocator
  ax.set_yticklabels(['']+labels)
```

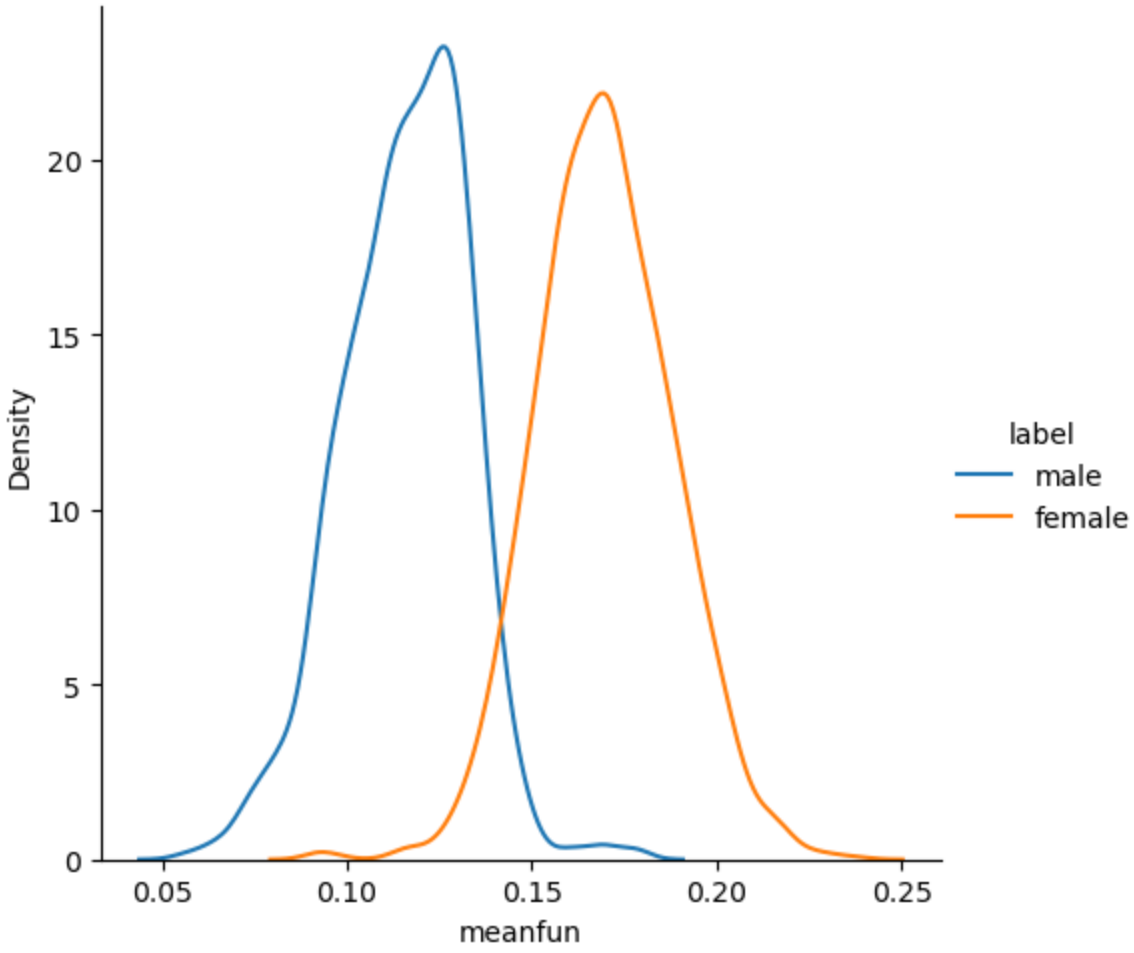


In [29]:
```
print (classification_report(ytest, y_pred))
```

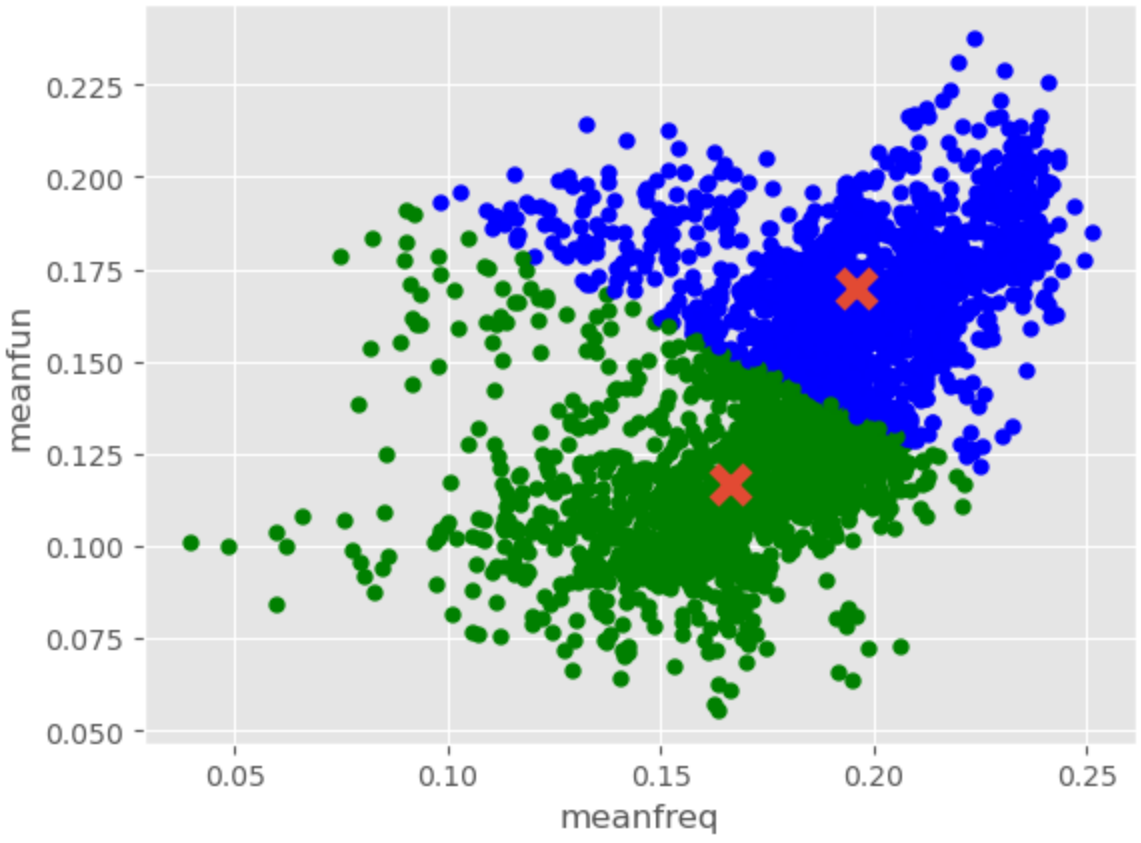
```
              precision    recall  f1-score   support

      female       0.99      1.00      0.99       299
        male       1.00      0.99      1.00       439

    accuracy                           0.99       738
   macro avg       0.99      1.00      0.99       738
weighted avg       0.99      0.99      0.99       738
```

In [30]:
```
sns.FacetGrid(df, hue="label", size=5) .map(sns.kdeplot, "meanfun").add_legend()
plt. show()
```

```
C:\Users\Abdul\anaconda3\lib\site-packages\seaborn\axisgrid.py:337: UserWarning: The `
size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```



In [33]:
```
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import numpy as np
from matplotlib import style

style.use("ggplot")

df = pd.read_csv("D:\\Last Term\\Speach Processing\\Project\\Dataset\\voice.csv")

data_x = np.array(df[['meanfreq', 'meanfun']])

kmeans = KMeans(n_clusters=2)
kmeans.fit(data_x)

centroids = kmeans.cluster_centers_
labels = kmeans.labels_

colors = ["g.", "b."]

for i in range(len(data_x)):
    plt.plot(data_x[i][0], data_x[i][1], colors[labels[i]], markersize=10)

plt.scatter(centroids[:, 0], centroids[:, 1], marker="x", s=150, linewidths=5, zorder=10

plt.ylabel('meanfun')
plt.xlabel('meanfreq')

plt.show()
```



In [ ]: