Gender Voice Prediction Project

In [1]:
```
# Importing required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.ensemble import RandomForestClassifier
import mglearn
```

In [2]:
```
# Import the dataset
gen_data=pd.read_csv('voice.csv')
```

In [3]:
```
gen_data.head()
```

Out[3]:

	meanfreq	sd	median	Q25	Q75	IQR	skew	kurt	sp.ent	sfm	...	centroid	meanfun	minfun	ma
0	0.059781	0.064241	0.032027	0.015071	0.090193	0.075122	12.863462	274.402906	0.893369	0.491918	...	0.059781	0.084279	0.015702	0.27
1	0.066009	0.067310	0.040229	0.019414	0.092666	0.073252	22.423285	634.613855	0.892193	0.513724	...	0.066009	0.107937	0.015826	0.25
2	0.077316	0.083829	0.036718	0.008701	0.131908	0.123207	30.757155	1024.927705	0.846389	0.478905	...	0.077316	0.098706	0.015656	0.27
3	0.151228	0.072111	0.158011	0.096582	0.207955	0.111374	1.232831	4.177296	0.963322	0.727232	...	0.151228	0.088965	0.017798	0.25
4	0.135120	0.079146	0.124656	0.078720	0.206045	0.127325	1.101174	4.333713	0.971955	0.783568	...	0.135120	0.106398	0.016931	0.26

5 rows × 21 columns

In [4]:
```
#Check the null values presene in the dataset.
gen_data.isnull().sum()
```

Out[4]:
```
meanfreq    0
sd          0
median      0
Q25         0
Q75         0
IQR         0
skew        0
kurt        0
sp.ent      0
sfm         0
mode        0
centroid    0
meanfun     0
minfun      0
maxfun      0
meandom     0
mindom      0
maxdom      0
dfrange     0
modindx     0
label       0
dtype: int64
```

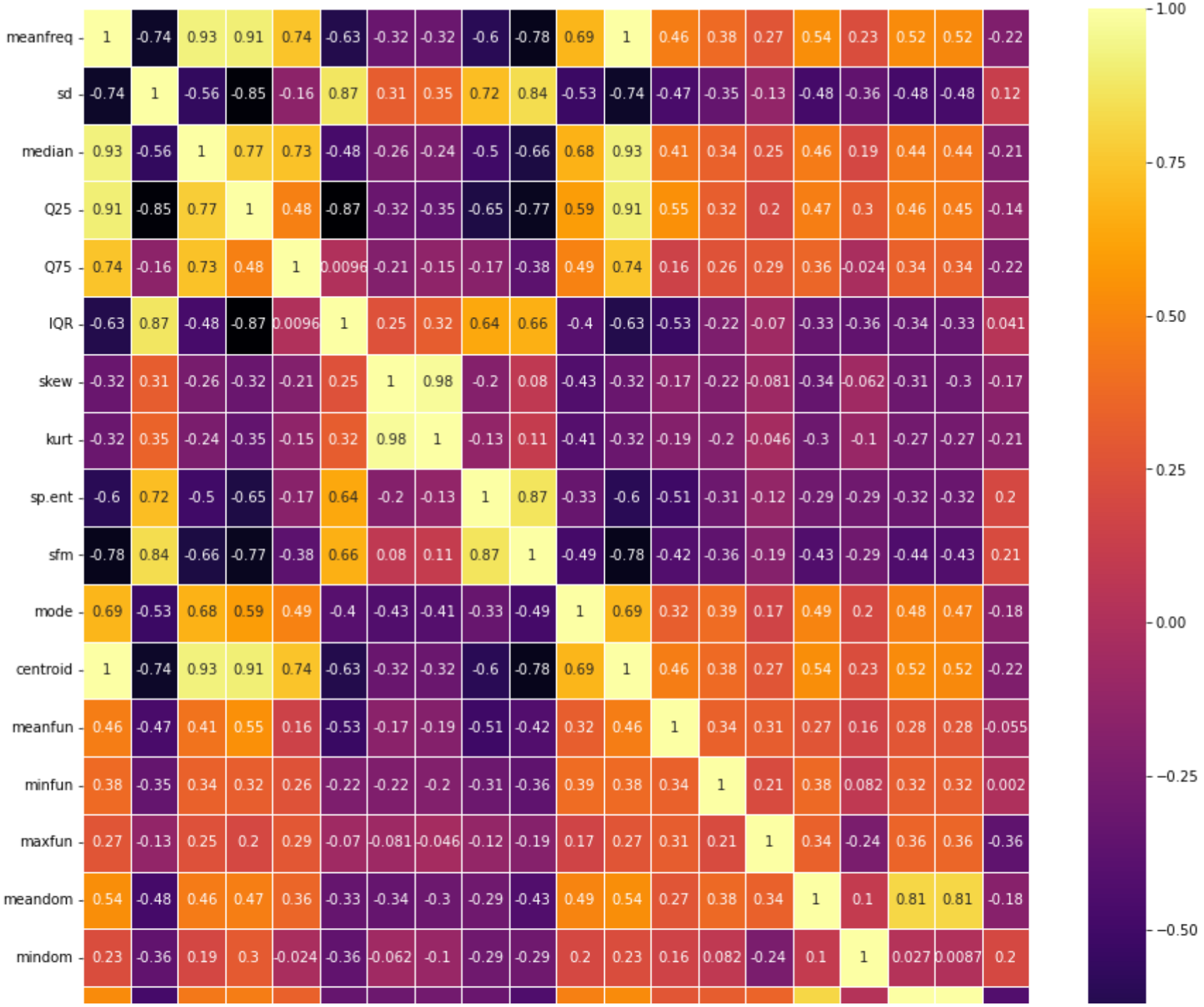
In [5]:
```
# To extract total count, min, max & so on...
gen_data.describe()
```

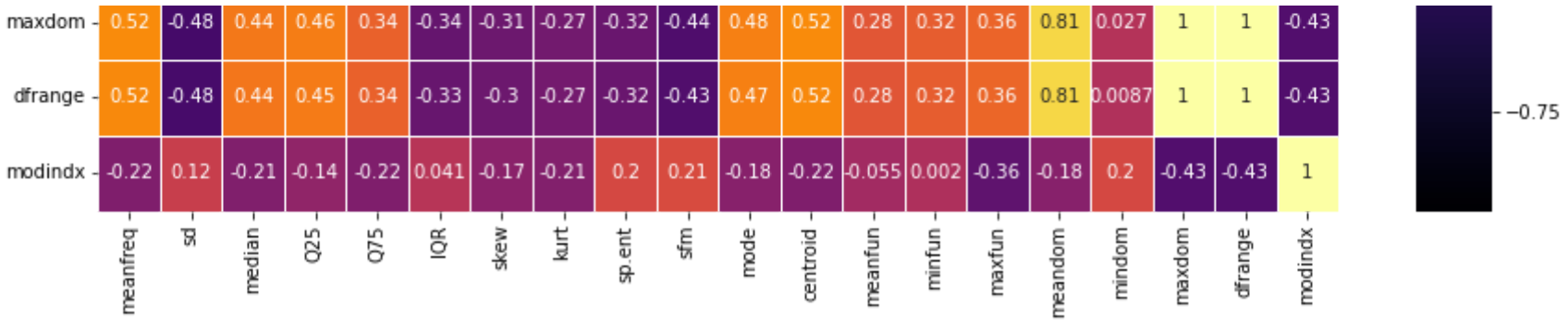
Out[5]:

	meanfreq	sd	median	Q25	Q75	IQR	skew	kurt	sp.ent	sfm	
count	3168.000000	3168.000000	3168.000000	3168.000000	3168.000000	3168.000000	3168.000000	3168.000000	3168.000000	3168.000000	3168.0
mean	0.180907	0.057126	0.185621	0.140456	0.224765	0.084309	3.140168	36.568461	0.895127	0.408216	0.1
std	0.029918	0.016652	0.036360	0.048680	0.023639	0.042783	4.240529	134.928661	0.044980	0.177521	0.0
min	0.039363	0.018363	0.010975	0.000229	0.042946	0.014558	0.141735	2.068455	0.738651	0.036876	0.0
25%	0.163662	0.041954	0.169593	0.111087	0.208747	0.042560	1.649569	5.669547	0.861811	0.258041	0.1
50%	0.184838	0.059155	0.190032	0.140286	0.225684	0.094280	2.197101	8.318463	0.901767	0.396335	0.1
75%	0.199146	0.067020	0.210618	0.175939	0.243660	0.114175	2.931694	13.648905	0.928713	0.533676	0.2
max	0.251124	0.115273	0.261224	0.247347	0.273469	0.252225	34.725453	1309.612887	0.981997	0.842936	0.2

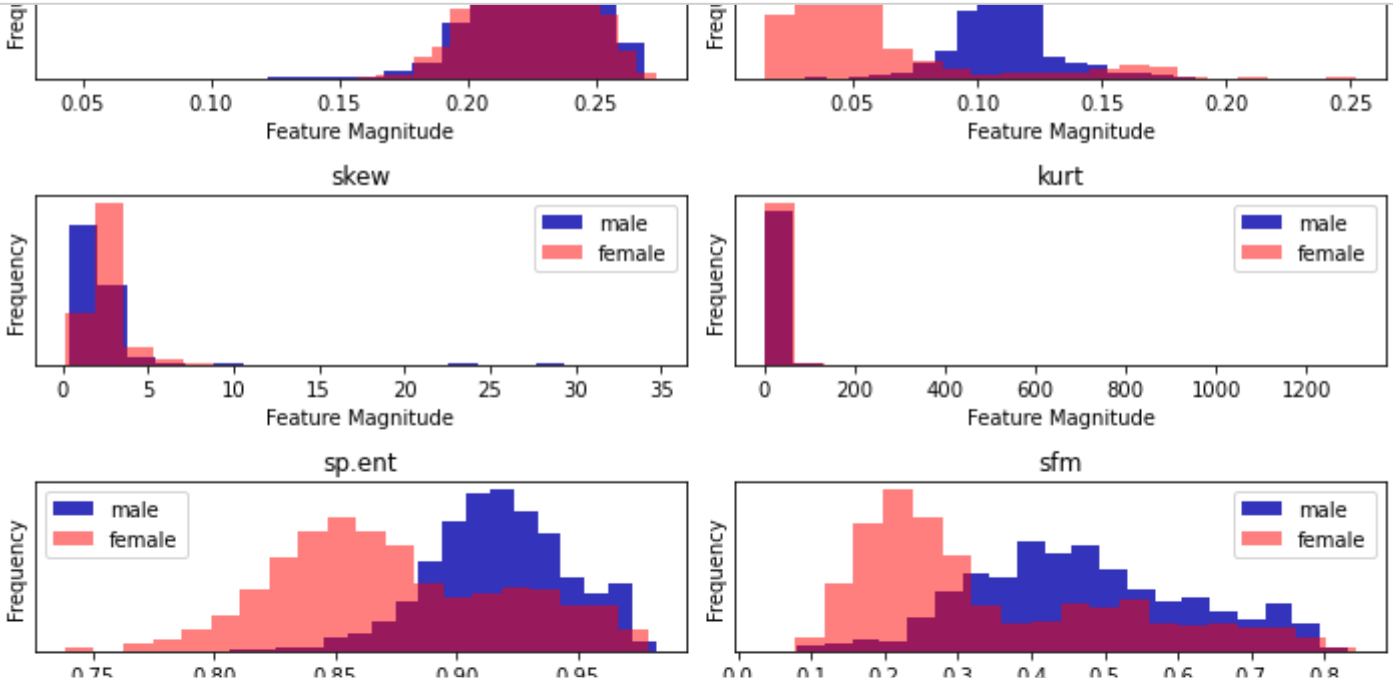
In [6]:
```
# Generate Heat Map
plt.figure(figsize=(15,15))
sns.heatmap(gen_data.corr(),annot=True,cmap='inferno',linewidth=0.5)
```

Out[6]:
```
<AxesSubplot:>
```


meanfreq 1 -0.74 0.93 0.91 0.74 -0.63 -0.32 -0.32 -0.6 -0.78 0.69 1 0.46 0.38 0.27 0.54 0.23 0.52 0.52 -0.22
sd -0.74 1 -0.56 -0.85 -0.16 0.87 0.31 0.35 0.72 0.84 -0.53 -0.74 -0.47 -0.35 -0.13 -0.48 -0.36 -0.48 -0.48 0.12
median 0.93 -0.56 1 0.77 0.73 -0.48 -0.26 -0.24 -0.5 -0.66 0.68 0.93 0.41 0.34 0.25 0.46 0.19 0.44 0.44 -0.21
Q25 0.91 -0.85 0.77 1 0.48 -0.87 -0.32 -0.35 -0.65 -0.77 0.59 0.91 0.55 0.32 0.2 0.47 0.3 0.46 0.45 -0.14
Q75 0.74 -0.16 0.73 0.48 1 0.0096 -0.21 -0.15 -0.17 -0.38 0.49 0.74 0.16 0.26 0.29 0.36 -0.024 0.34 0.34 -0.22
IQR -0.63 0.87 -0.48 -0.87 0.0096 1 0.25 0.32 0.64 0.66 -0.4 -0.63 -0.53 -0.22 -0.07 -0.33 -0.36 -0.34 -0.33 0.041
skew -0.32 0.31 -0.26 -0.32 -0.21 0.25 1 0.98 -0.2 0.08 -0.43 -0.32 -0.17 -0.22 -0.081 -0.34 -0.062 -0.31 -0.3 -0.17
kurt -0.32 0.35 -0.24 -0.35 -0.15 0.32 0.98 1 -0.13 0.11 -0.41 -0.32 -0.19 -0.2 -0.046 -0.3 -0.1 -0.27 -0.27 -0.21
sp.ent -0.6 0.72 -0.5 -0.65 -0.17 0.64 -0.2 -0.13 1 0.87 -0.33 -0.6 -0.51 -0.31 -0.12 -0.29 -0.29 -0.32 -0.32 0.2
sfm -0.78 0.84 -0.66 -0.77 -0.38 0.66 0.08 0.11 0.87 1 -0.49 -0.78 -0.42 -0.36 -0.19 -0.43 -0.29 -0.44 -0.43 0.21
mode 0.69 -0.53 0.68 0.59 0.49 -0.4 -0.43 -0.41 -0.33 -0.49 1 0.69 0.32 0.39 0.17 0.49 0.2 0.48 0.47 -0.18
centroid 1 -0.74 0.93 0.91 0.74 -0.63 -0.32 -0.32 -0.6 -0.78 0.69 1 0.46 0.38 0.27 0.54 0.23 0.52 0.52 -0.22
meanfun 0.46 -0.47 0.41 0.55 0.16 -0.53 -0.17 -0.19 -0.51 -0.42 0.32 0.46 1 0.34 0.31 0.27 0.16 0.28 0.28 -0.055
minfun 0.38 -0.35 0.34 0.32 0.26 -0.22 -0.22 -0.2 -0.31 -0.36 0.39 0.38 0.34 1 0.21 0.38 0.082 0.32 0.32 0.002
maxfun 0.27 -0.13 0.25 0.2 0.29 -0.07 -0.081 -0.046 -0.12 -0.19 0.17 0.27 0.31 0.21 1 0.34 -0.24 0.36 0.36 -0.36
meandom 0.54 -0.48 0.46 0.47 0.36 -0.33 -0.34 -0.3 -0.29 -0.43 0.49 0.54 0.27 0.38 0.34 1 0.1 0.81 0.81 -0.18
mindom 0.23 -0.36 0.19 0.3 -0.024 -0.36 -0.062 -0.1 -0.29 -0.29 0.2 0.23 0.16 0.082 -0.24 0.1 1 0.027 0.0087 0.2
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50


maxdom
dfrange
modindx
0.52 -0.48 0.44 0.46 0.34 -0.34 -0.31 -0.27 -0.32 -0.44 0.48 0.52 0.28 0.32 0.36 0.81 0.027 1 1 -0.43
0.52 -0.48 0.44 0.45 0.34 -0.33 -0.3 -0.27 -0.32 -0.43 0.47 0.52 0.28 0.32 0.36 0.81 0.0087 1 1 -0.43
-0.22 0.12 -0.21 -0.14 -0.22 0.041 -0.17 -0.21 0.2 0.21 -0.18 -0.22 -0.055 0.002 -0.36 -0.18 0.2 -0.43 -0.43 1
meanfreq
sd
median
Q25
Q75
IQR
skew
kurt
sp.ent
sfm
mode
centroid
meanfun
minfun
maxfun
meandom
mindom
maxdom
dfrange
modindx
0.75

In [7]:
```
# PLot the Histograms
male=gen_data.loc[gen_data['label']=='male']
female=gen_data.loc[gen_data['label']=='female']
fig, axes = plt.subplots(10,2,figsize=(10,20))
ax = axes.ravel()
for i in range(20):
    ax[i].hist(male.iloc[:,i],bins=20,color=mglearn.cm3(0),alpha=0.8)
    ax[i].hist(female.iloc[:,i],bins=20,color='red',alpha=0.5)
    ax[i].set_title(list(male)[i])
    ax[i].set_yticks(())
    ax[i].set_xlabel("Feature Magnitude")
    ax[i].set_ylabel("Frequency")
    ax[i].legend(["male","female"],loc="best")
fig.tight_layout()
```



In [8]:
```
# drop the unwanted columns
gen_new=gen_data.drop(['skew','kurt','minfun','maxfun','mindom'],axis=1)
```

In [9]:
```
gen_new.columns
```

Out[9]:
```
Index(['meanfreq', 'sd', 'median', 'Q25', 'Q75', 'IQR', 'sp.ent', 'sfm',
       'mode', 'centroid', 'meanfun', 'meandom', 'maxdom', 'dfrange',
       'modindx', 'label'],
      dtype='object')
```

In [10]:
```
# Define X & y variables.
X = gen_new.drop(['label'],axis=1)
y = gen_new['label']
```

In [11]:
```
# Split the data into train & test
Xtrain, Xtest, ytrain, ytest = train_test_split(X,y,test_size=0.2)
```

In [12]:
```
# Bulid the model using Random Forest Classifier & fit it on Xtrain & ytrain
forest = RandomForestClassifier(n_estimators=500, random_state=101).fit(Xtrain,ytrain)
print("Random Forest")
print("Accuracy on Training Set: {:.2f}".format(forest.score(Xtrain,ytrain)))
print("Accuracy on Test Set: {:.2f}".format(forest.score(Xtest,ytest)))
```

```
Random Forest
Accuracy on Training Set: 1.00
Accuracy on Test Set: 0.98
```

In [15]:
```
from sklearn.metrics import classification_report, confusion_matrix
y_pred = forest.predict(Xtest)
print(classification_report(ytest,y_pred))
```

```
              precision    recall  f1-score   support

      female       0.98      0.99      0.98       323
        male       0.99      0.98      0.98       311

    accuracy                           0.98       634
   macro avg       0.98      0.98      0.98       634
weighted avg       0.98      0.98      0.98       634
```

Conclusion:

The gender voice prediction project successfully employed machine learning technique to accurately predict gender based on voice features. By analyzing and processing vocal characteristics, the model achieved commendable performance, showcasing its potential for real-world applications such as speech recognition systems and gender-specific services.