```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data=pd.read_csv("pd_speech_features.csv")
```

```
data.head()
```

	id	gender	PPE	DFA	RPDE	numPulses	numPeriodsPulses	meanPeriodPulses	stdDevPeriodPulses	locPctJitter	...	tqwt_kurto
0	0	1	0.85247	0.71826	0.57227	240	239	0.008064	0.000087	0.00218	...	
1	0	1	0.76686	0.69481	0.53966	234	233	0.008258	0.000073	0.00195	...	
2	0	1	0.85083	0.67604	0.58982	232	231	0.008340	0.000060	0.00176	...	
3	1	0	0.41121	0.79672	0.59257	178	177	0.010858	0.000183	0.00419	...	
4	1	0	0.32790	0.79782	0.53028	236	235	0.008162	0.002669	0.00535	...	

5 rows × 755 columns

```
data.corr() #Checking Correlation
```

	id	gender	PPE	DFA	RPDE	numPulses	numPeriodsPulses	meanPeriodPulses	stdDevPeriodPu
id	1.000000	-0.133605	0.026667	0.041938	-0.084606	0.085828	0.085226	-0.100831	-0.06
gender	-0.133605	1.000000	0.010175	0.099356	0.168321	-0.478367	-0.477710	0.460422	-0.11
PPE	0.026667	0.010175	1.000000	-0.094775	-0.405558	0.191535	0.194098	-0.201907	-0.42
DFA	0.041938	0.099356	-0.094775	1.000000	0.155075	-0.286791	-0.286611	0.253708	0.08
RPDE	-0.084606	0.168321	-0.405558	0.155075	1.000000	-0.521193	-0.524839	0.506707	0.33
...	...	...	...	...	...	...	...	...	
tqwt_kurtosisValue_dec_33	-0.053781	0.107618	0.045755	-0.033914	-0.097161	-0.047946	-0.047837	0.039487	-0.09
tqwt_kurtosisValue_dec_34	-0.058034	0.128936	0.046175	0.043454	-0.036406	-0.068664	-0.068561	0.046140	-0.06
tqwt_kurtosisValue_dec_35	-0.057807	0.107734	0.037385	0.067843	-0.018052	-0.069645	-0.069468	0.040828	-0.06
tqwt_kurtosisValue_dec_36	-0.055775	0.104828	0.039588	0.116699	0.021945	-0.062925	-0.062790	0.031685	-0.06
class	-0.111661	0.182713	-0.072939	0.306070	0.247444	-0.284056	-0.284002	0.211368	0.07

755 rows × 755 columns

```
data.describe() #Checking for Outliers
```

	id	gender	PPE	DFA	RPDE	numPulses	numPeriodsPulses	meanPeriodPulses	stdDevPeriodPulses	locP
count	756.000000	756.000000	756.000000	756.000000	756.000000	756.000000	756.000000	756.000000	756.000000	75
mean	125.500000	0.515873	0.746284	0.700414	0.489058	323.972222	322.678571	0.006360	0.000383	
std	72.793721	0.500079	0.169294	0.069718	0.137442	99.219059	99.402499	0.001826	0.000728	
min	0.000000	0.000000	0.041551	0.543500	0.154300	2.000000	1.000000	0.002107	0.000011	
25%	62.750000	0.000000	0.762833	0.647053	0.386537	251.000000	250.000000	0.005003	0.000049	
50%	125.500000	1.000000	0.809655	0.700525	0.484355	317.000000	316.000000	0.006048	0.000077	
75%	188.250000	1.000000	0.834315	0.754985	0.586515	384.250000	383.250000	0.007528	0.000171	
max	251.000000	1.000000	0.907660	0.852640	0.871230	907.000000	905.000000	0.012966	0.003483	

8 rows × 755 columns

```
X=data.drop("class",axis=1) #Splited the data in Dependent
y=data["class"] #Splited the data in Independent
```

```
!pip install xgboost
!pip install catboost
```

```
Requirement already satisfied: xgboost in /usr/local/lib/python3.10/dist-packages (2.1.3)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from xgboost) (1.26.4)
Requirement already satisfied: nvidia-nccl-cu12 in /usr/local/lib/python3.10/dist-packages (from xgboost) (2.23.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from xgboost) (1.13.1)
Requirement already satisfied: catboost in /usr/local/lib/python3.10/dist-packages (1.2.7)
Requirement already satisfied: graphviz in /usr/local/lib/python3.10/dist-packages (from catboost) (0.20.3)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from catboost) (3.8.0)
Requirement already satisfied: numpy<2.0,>=1.16.0 in /usr/local/lib/python3.10/dist-packages (from catboost) (1.26.4)
Requirement already satisfied: pandas>=0.24 in /usr/local/lib/python3.10/dist-packages (from catboost) (2.2.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from catboost) (1.13.1)
Requirement already satisfied: plotly in /usr/local/lib/python3.10/dist-packages (from catboost) (5.24.1)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from catboost) (1.17.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.24->catboost) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.24->catboost) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas>=0.24->catboost) (2024.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (4.55.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (24.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->catboost) (3.2.0)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly->catboost) (9.0.0)
```

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=43) #Splited the data in test and train as 80% train and 20% t
```

```
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score

models={
    "RandomForestClassifier":RandomForestClassifier(),
    "XGBClassifier":XGBClassifier()
} #Checking with both models to know which can get more accuracy and precision

for i in models:
  model=models[i]
  model.fit(X_train,y_train)
  y_pred_test=model.predict(X_test)
  print(i)
  print("Accuracy_score:- ",accuracy_score(y_pred_test,y_test))
```

```
RandomForestClassifier
Accuracy_score:-  0.8448844884488449
XGBClassifier
Accuracy_score:-  0.8613861386138614
```

```
params={
    "n_estimators" : [110,100,150],
    "criterion" : ["gini","entropy"],
    "max_depth" : [1,2,3,4],
    "min_samples_leaf" :[1,2,3]
} # Custom Parameters Testing so that we can find best parameters for our models
rfc_model=XGBClassifier()
gd=GridSearchCV(rfc_model,params,cv=5,n_jobs=-1,verbose=2)
gd.fit(X_train,y_train)
```

```
Fitting 5 folds for each of 72 candidates, totalling 360 fits
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:158: UserWarning: [07:52:27] WARNING: /workspace/src/learner.cc:740:
Parameters: { "criterion", "min_samples_leaf" } are not used.

  warnings.warn(smsg, UserWarning)
```

```
▸ GridSearchCV
▸ best_estimator_: XGBClassifier
    ▸ XGBClassifier
```

```
gd.best_params_ # Best Parameters
```

```
{'criterion': 'gini',
 'max_depth': 2,
 'min_samples_leaf': 1,
 'n_estimators': 100}
```

```
params={
    "n_estimators" : 100,
    "max_depth" : 3,
    "min_samples_leaf" :2
}
xgb_model=XGBClassifier(**params)
xgb_model.fit(X_train,y_train)
y_pred_test=xgb_model.predict(X_test)
print("Accuracy_score:- ",accuracy_score(y_pred_test,y_test))
```

```
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:158: UserWarning: [07:58:30] WARNING: /workspace/src/learner.cc:740:
Parameters: { "min_samples_leaf" } are not used.

  warnings.warn(smsg, UserWarning)
Accuracy_score:-  0.881578947368421
```

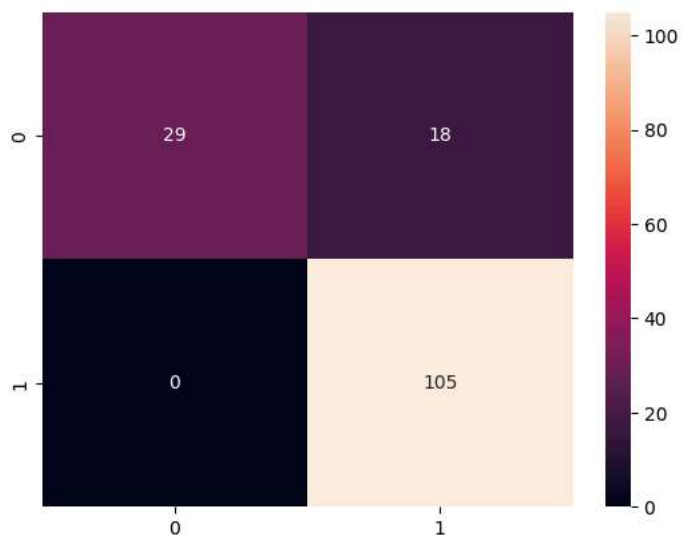
```
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,y_pred_test))
sns.heatmap(confusion_matrix(y_test,y_pred_test),annot=True,fmt="d")
```

```
              precision    recall  f1-score   support

           0       1.00      0.62      0.76        47
           1       0.85      1.00      0.92       105

    accuracy                           0.88       152
   macro avg       0.93      0.81      0.84       152
weighted avg       0.90      0.88      0.87       152

<Axes: >
```




```
from sklearn.metrics import accuracy_score
print(accuracy_score(y_test,y_pred_test))
```

```
0.881578947368421
```

```
from joblib import load,dump
dump(xgb_model,"xgb_model.joblib")
```

```
['xgb_model.joblib']
```