```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,
classification_report,ConfusionMatrixDisplay
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier

df = pd.read_csv('pd_speech_features.csv')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 756 entries, 0 to 755
Columns: 755 entries, id to class
dtypes: float64(749), int64(6)
memory usage: 4.4 MB
```

```
df.head()
```

```
   id  gender      PPE      DFA     RPDE  numPulses  numPeriodsPulses
\
0   0       1  0.85247  0.71826  0.57227        240               239

1   0       1  0.76686  0.69481  0.53966        234               233

2   0       1  0.85083  0.67604  0.58982        232               231

3   1       0  0.41121  0.79672  0.59257        178               177

4   1       0  0.32790  0.79782  0.53028        236               235


   meanPeriodPulses  stdDevPeriodPulses  locPctJitter  ...  \
0          0.008064            0.000087       0.00218  ...
1          0.008258            0.000073       0.00195  ...
2          0.008340            0.000060       0.00176  ...
3          0.010858            0.000183       0.00419  ...
4          0.008162            0.002669       0.00535  ...

   tqwt_kurtosisValue_dec_28  tqwt_kurtosisValue_dec_29  \
0                     1.5620                     2.6445
1                     1.5589                     3.6107
2                     1.5643                     2.3308
3                     3.7805                     3.5664
4                     6.1727                     5.8416
```

```
   tqwt_kurtosisValue_dec_30  tqwt_kurtosisValue_dec_31  \
0                     3.8686                     4.2105
1                    23.5155                    14.1962
2                     9.4959                    10.7458
3                     5.2558                    14.0403
4                     6.0805                     5.7621

   tqwt_kurtosisValue_dec_32  tqwt_kurtosisValue_dec_33  \
0                     5.1221                     4.4625
1                    11.0261                     9.5082
2                    11.0177                     4.8066
3                     4.2235                     4.6857
4                     7.7817                    11.6891

   tqwt_kurtosisValue_dec_34  tqwt_kurtosisValue_dec_35  \
0                     2.6202                     3.0004
1                     6.5245                     6.3431
2                     2.9199                     3.1495
3                     4.8460                     6.2650
4                     8.2103                     5.0559

   tqwt_kurtosisValue_dec_36  class
0                    18.9405      1
1                    45.1780      1
2                     4.7666      1
3                     4.0603      1
4                     6.1164      1

[5 rows x 755 columns]
```

```
null_values=df.isnull().sum()
```

```
sum(null_values!=0)
```

```
0
```

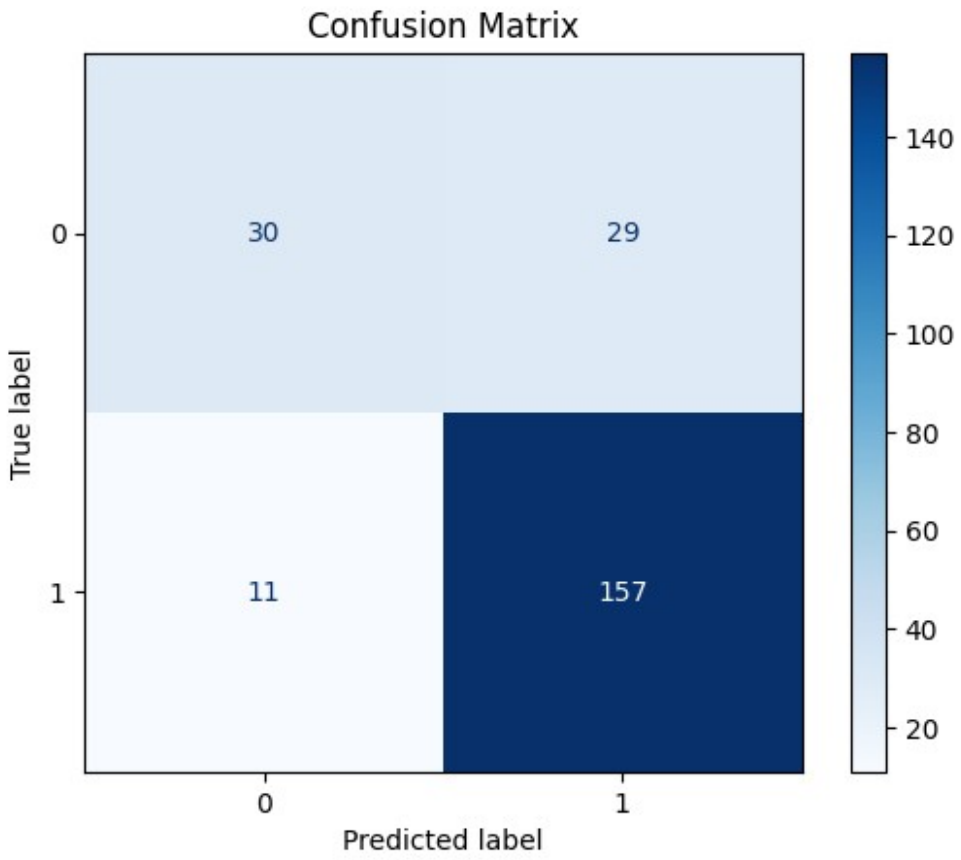
```
y = df.loc[:,'class']
X = df.drop(['class', 'id'], axis=1)

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=80)

sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

clf = DecisionTreeClassifier(max_depth=5, random_state=42)
clf.fit(X_train, y_train)
```

```
DecisionTreeClassifier(max_depth=5, random_state=42)
```

```
y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Classification Report:\n", classification_report(y_test,
y_pred))
```

```
Accuracy: 0.8237885462555066
Classification Report:
              precision    recall  f1-score   support

           0       0.73      0.51      0.60        59
           1       0.84      0.93      0.89       168

    accuracy                           0.82       227
   macro avg       0.79      0.72      0.74       227
weighted avg       0.81      0.82      0.81       227
```

```
cm = confusion_matrix(y_test, y_pred, labels=clf.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=clf.classes_)
disp.plot(cmap='Blues')
plt.title("Confusion Matrix")
plt.show()
```



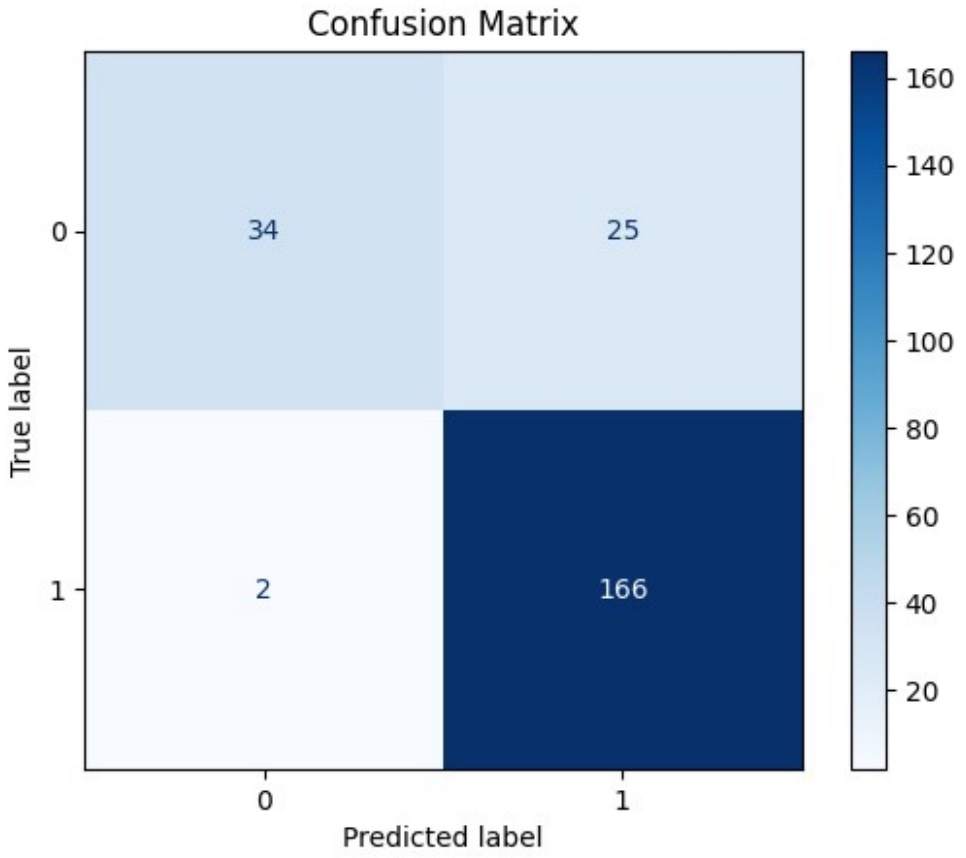

```
rf_model = RandomForestClassifier (n_estimators =100, max_depth =5,
random_state =42)

rf_model.fit(X_train, y_train)
```

```
RandomForestClassifier(max_depth=5, random_state=42)
```

```
y_pred = rf_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Classification Report:\n", classification_report(y_test,
y_pred))
```

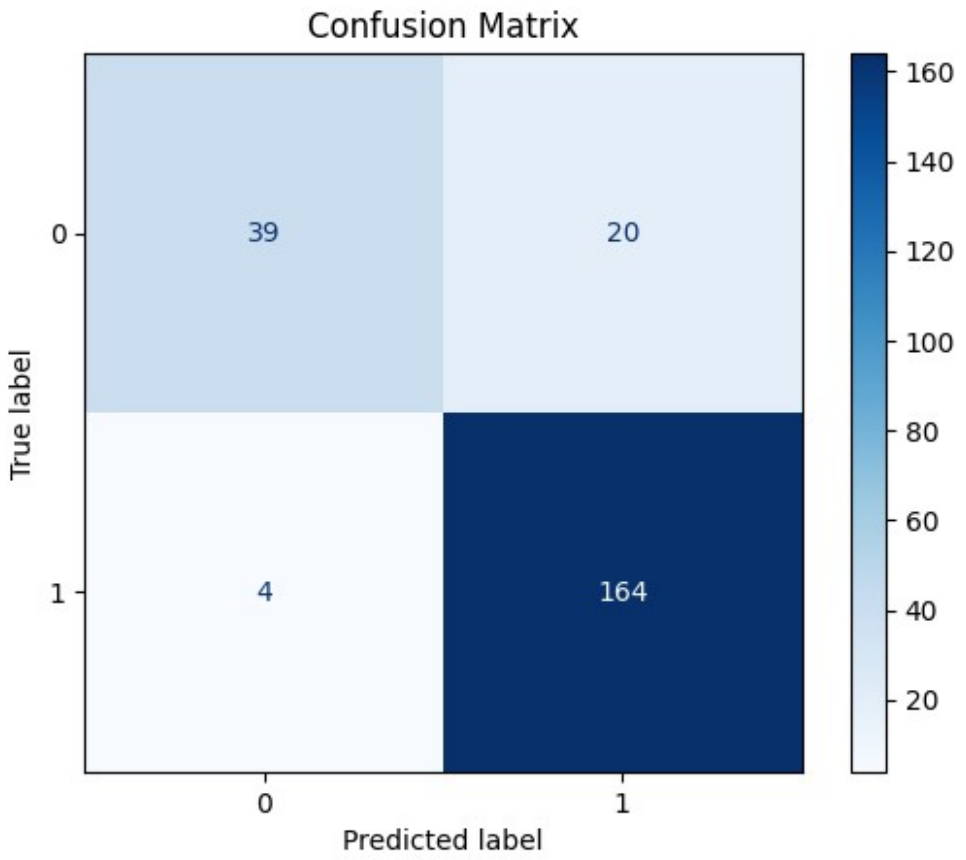
```
Accuracy: 0.8810572687224669
Classification Report:
               precision    recall  f1-score   support

           0       0.94      0.58      0.72        59
           1       0.87      0.99      0.92       168

    accuracy                           0.88       227
   macro avg       0.91      0.78      0.82       227
weighted avg       0.89      0.88      0.87       227
```

```
cm = confusion_matrix(y_test, y_pred, labels=rf_model.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=clf.classes_)
disp.plot(cmap='Blues')
plt.title("Confusion Matrix")
plt.show()
```




```
xgb = XGBClassifier(n_estimators =100 , max_depth =5 , learning_rate
=0.1)


xgb.fit(X_train, y_train )

XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None,
early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None,
feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.1,
max_bin=None,
```

```
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=5, max_leaves=None,
              min_child_weight=None, missing=nan,
monotone_constraints=None,
              multi_strategy=None, n_estimators=100, n_jobs=None,
              num_parallel_tree=None, random_state=None, ...)

y_pred = xgb.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Classification Report:\n", classification_report(y_test,
y_pred))

Accuracy: 0.8942731277533039
Classification Report:
               precision    recall  f1-score   support

           0       0.91      0.66      0.76        59
           1       0.89      0.98      0.93       168

    accuracy                           0.89       227
   macro avg       0.90      0.82      0.85       227
weighted avg       0.90      0.89      0.89       227


cm = confusion_matrix(y_test, y_pred, labels=xgb.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,
display_labels=xgb.classes_)
disp.plot(cmap='Blues')
plt.title("Confusion Matrix")
plt.show()
```


Confusion Matrix
39
20
4
164
0
1
True label
0
1
Predicted label
160
140
120
100
80
60
40
20