

```
In [1]: from pyAudioAnalysis import audioTrainTest as aT
aT.featureAndTrain(["train/angry","train/calm","train/disgust","train/fearful"
,"train/happy","train/sad","train/surprised","train/neutral"], 1.0, 1.0, aT.sh
ortTermWindow, aT.shortTermStep, "randomforest", "svmSMtemp", False)
```

```
In [2]: import os
import numpy as np
angry = []
for root, dirs, files in os.walk(r'test/angry/'):
    for file in files:
        if file.endswith('.wav'):
            angry.append(file)
calm = []
for root, dirs, files in os.walk(r'test/calm/'):
    for file in files:
        if file.endswith('.wav'):
            calm.append(file)
disgust = []
for root, dirs, files in os.walk(r'test/disgust/'):
    for file in files:
        if file.endswith('.wav'):
            disgust.append(file)
fearful = []
for root, dirs, files in os.walk(r'test/fearful/'):
    for file in files:
        if file.endswith('.wav'):
            fearful.append(file)
happy = []
for root, dirs, files in os.walk(r'test/happy/'):
    for file in files:
        if file.endswith('.wav'):
            happy.append(file)
sad = []
for root, dirs, files in os.walk(r'test/sad/'):
    for file in files:
        if file.endswith('.wav'):
            sad.append(file)
surprised = []
for root, dirs, files in os.walk(r'test/surprised/'):
    for file in files:
        if file.endswith('.wav'):
            surprised.append(file)
neutral = []
for root, dirs, files in os.walk(r'test/neutral/'):
    for file in files:
        if file.endswith('.wav'):
            neutral.append(file)
```

```
In [3]: c = []
        for i in angry:
            c = np.append(c,aT.fileClassification("test/angry/"+i, "svmSMtemp","randomforest"))
        for i in calm:
            c = np.append(c,aT.fileClassification("test/calm/"+i, "svmSMtemp","randomforest"))
        for i in disgust:
            c = np.append(c,aT.fileClassification("test/disgust/"+i,"svmSMtemp","randomforest"))
        for i in fearful:
            c = np.append(c,aT.fileClassification("test/fearful/"+i,"svmSMtemp","randomforest"))
        for i in happy:
            c = np.append(c,aT.fileClassification("test/happy/"+i,"svmSMtemp","randomforest"))
        for i in sad:
            c = np.append(c,aT.fileClassification("test/sad/"+i,"svmSMtemp","randomforest"))
        for i in surprised:
            c = np.append(c,aT.fileClassification("test/surprised/"+i,"svmSMtemp","randomforest"))
        for i in neutral:
            c = np.append(c,aT.fileClassification("test/neutral/"+i,"svmSMtemp","randomforest"))
        c = np.reshape(c,(-1,8))
```

```
In [4]: c
```

```
Out[4]: array([[0.198, 0.09 , 0.23 , ..., 0.144, 0.088, 0.056],
               [0.28 , 0.014, 0.05 , ..., 0.128, 0.13 , 0.016],
               [0.258, 0.036, 0.062, ..., 0.064, 0.064, 0.014],
               ...,
               [0.07 , 0.21 , 0.06 , ..., 0.132, 0.11 , 0.232],
               [0.078, 0.28 , 0.066, ..., 0.094, 0.11 , 0.228],
               [0.074, 0.292, 0.08 , ..., 0.104, 0.098, 0.156]])
```

```
In [5]: y_pred = np.argmax(c,axis = 1)
        y_pred
```

```
Out[5]: array([2, 0, 3, 0, 5, 0, 2, 0, 2, 0, 0, 0, 0, 0, 6, 0, 0, 3, 0, 0, 3, 0,
               2, 0, 0, 7, 0, 0, 0, 0, 0, 0, 2, 3, 0, 0, 2, 0, 0, 0, 3, 3, 0, 1,
               0, 0, 0, 6, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 5, 1, 1, 1,
               1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 5, 1, 5, 2, 2, 1, 5, 1, 2,
               1, 1, 1, 1, 1, 1, 5, 2, 2, 2, 0, 6, 4, 2, 2, 2, 4, 2, 7, 4, 6, 2,
               2, 0, 1, 0, 6, 6, 0, 2, 2, 0, 2, 2, 0, 7, 2, 3, 0, 2, 2, 0, 2, 2,
               2, 5, 2, 1, 5, 2, 2, 0, 0, 2, 3, 2, 5, 6, 5, 3, 1, 3, 3, 1, 4, 3,
               5, 2, 4, 6, 3, 3, 3, 3, 3, 5, 3, 3, 3, 2, 3, 3, 3, 6, 0, 3, 3, 3,
               5, 4, 0, 3, 3, 3, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 0, 6, 4, 4, 1, 0,
               0, 5, 2, 4, 5, 5, 5, 6, 1, 5, 4, 0, 6, 6, 6, 6, 1, 4, 5, 4, 4, 0,
               4, 6, 4, 6, 4, 2, 5, 4, 6, 1, 6, 6, 4, 2, 4, 2, 6, 4, 3, 3, 5, 1,
               1, 2, 3, 1, 1, 5, 5, 3, 3, 5, 3, 2, 2, 2, 1, 5, 1, 3, 1, 1, 5, 1,
               1, 5, 6, 1, 2, 2, 5, 1, 5, 3, 1, 5, 5, 6, 1, 5, 1, 5, 1, 5, 5, 5,
               5, 5, 6, 6, 6, 2, 6, 6, 4, 6, 3, 6, 6, 6, 6, 7, 0, 3, 6, 6, 6, 6,
               6, 4, 6, 5, 6, 6, 6, 6, 6, 0, 6, 0, 6, 2, 6, 6, 2, 5, 6, 0, 6, 5,
               3, 4, 2, 4, 6, 6, 7, 5, 1, 1, 1, 1, 5, 7, 1, 5, 1, 1, 2, 1, 7, 5,
               7, 1, 2, 1, 7, 7, 1, 1])
```

```
In [6]: y_test = []
        for i in range(len(y_pred)):
            if i < (len(angry)):
                y_test.append(0)
            elif i < (len(angry)+len(calm)):
                y_test.append(1)
            elif i < (len(angry)+len(calm)+len(disgust)):
                y_test.append(2)
            elif i < (len(angry)+len(calm)+len(disgust)+len(fearful)):
                y_test.append(3)
            elif i < (len(angry)+len(calm)+len(disgust)+len(fearful)+len(happy)):
                y_test.append(4)
            elif i < (len(angry)+len(calm)+len(disgust)+len(fearful)+len(happy)+len(sad)
)):
                y_test.append(5)
            elif i < (len(angry)+len(calm)+len(disgust)+len(fearful)+len(happy)+len(sad)
+len(surprised)):
                y_test.append(6)
            else:
                y_test.append(7)
        y_test
```

```
Out[6]: [0,
```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]


```
In [9]: from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score
Accuracy_Score = accuracy_score(y_test,y_pred)
Precision_Score = precision_score(y_test, y_pred, average="macro")
Recall_Score = recall_score(y_test, y_pred, average="macro")
F1_Score = f1_score(y_test, y_pred, average="macro")
```

```
In [10]: Accuracy_Score
```

```
Out[10]: 0.525
```

```
In [11]: Precision_Score
```

```
Out[11]: 0.5360582890917425
```

```
In [12]: Recall_Score
```

```
Out[12]: 0.5078125
```

```
In [13]: F1_Score
```

```
Out[13]: 0.5021964136048329
```