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, "extratrees", "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","extrat
rees"))
for i in calm:
    c = np.append(c,aT.fileClassification("test/calm/"+i, "svmSMtemp","extratr
ees"))
for i in disgust:
    c = np.append(c,aT.fileClassification("test/disgust/"+i,"svmSMtemp","extra
trees"))
for i in fearful:
    c = np.append(c,aT.fileClassification("test/fearful/"+i,"svmSMtemp","extra
trees"))
for i in happy:
    c = np.append(c,aT.fileClassification("test/happy/"+i,"svmSMtemp","extratr
ees"))
for i in sad:
    c = np.append(c,aT.fileClassification("test/sad/"+i,"svmSMtemp","extratree
s"))
for i in surprised:
    c = np.append(c,aT.fileClassification("test/surprised/"+i,"svmSMtemp","ext
ratrees"))
for i in neutral:
    c = np.append(c,aT.fileClassification("test/neutral/"+i,"svmSMtemp","extra
trees"))
c = np.reshape(c,(-1,8))
```

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

Out[4]:
```
array([[0.156, 0.11 , 0.204, ..., 0.136, 0.098, 0.078],
       [0.278, 0.014, 0.058, ..., 0.118, 0.154, 0.01 ],
       [0.318, 0.028, 0.07 , ..., 0.054, 0.066, 0.006],
       ...,
       [0.072, 0.234, 0.07 , ..., 0.13 , 0.096, 0.208],
       [0.088, 0.276, 0.082, ..., 0.106, 0.096, 0.202],
       [0.102, 0.256, 0.132, ..., 0.118, 0.088, 0.098]])
```

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

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

```
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
```

```
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
2,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
```

```
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
3,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
```

```
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
4,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
5,
```

```
5,
5,
5,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
6,
7,
7,
7,
7,
7,
7,
```

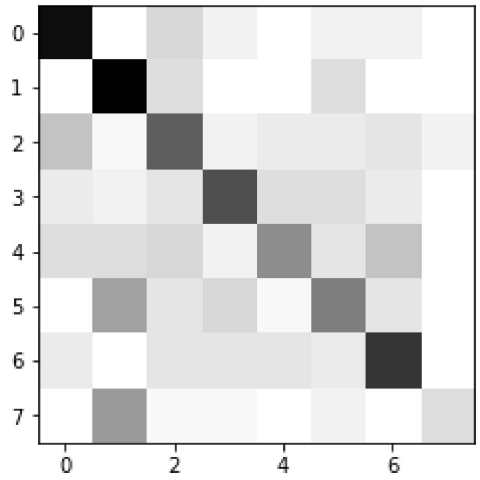
```
7,
7,
7,
7,
7,
7,
7,
7,
7,
7,
7,
7,
7,
7,
7,
7,
7,
7]
```

In [7]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
cm
```

Out[7]:
```
array([[36,  0,  6,  2,  0,  2,  2,  0],
       [ 0, 38,  5,  0,  0,  5,  0,  0],
       [ 9,  1, 24,  2,  3,  3,  4,  2],
       [ 3,  2,  4, 26,  5,  5,  3,  0],
       [ 5,  5,  6,  2, 17,  4,  9,  0],
       [ 0, 14,  4,  6,  1, 19,  4,  0],
       [ 3,  0,  4,  4,  4,  3, 30,  0],
       [ 0, 15,  1,  1,  0,  2,  0,  5]])
```

In [8]:
```
import matplotlib.pyplot as plt
plt.imshow(cm, cmap='binary')
```

Out[8]:
```
<matplotlib.image.AxesImage at 0x7f324ec19c10>
```



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]: 0.5416666666666666

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

Out[11]: 0.562294417468836

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

Out[12]: 0.5208333333333333

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

Out[13]: 0.5160338970666642