EMOTION RECOGNITION USING EEG SIGNALS

Date: 12 /06/2022

Dr. Shyama Prasad Mukherjee International Institute of Information Technology, Naya Raipur

Team

(1) Abhay Tiwari – 201010216 ECE
(2) Chahat Mittal – 201010215 ECE
(3) Chinmay Singh – 201010216 ECE

INTRODUCTION

- An electroencephalogram (EEG) is a test that detects electrical activity in our brain using small, metal discs (electrodes) attached to our scalp.
- EEG records the electrical activity of the brain via electrodes affixed to the scalp
- The electrodes detect tiny electrical charges that result from the activity of our brain cells.

DATASET

- DEAP (**A Dataset for Emotion Analysis using Physiological and Audiovisual Signals**) is used.
- EEG and peripheral physiological signals of 32 participants were recorded as each watched 40 one-minute long excerpts of music videos.
- Participants rated each video in terms of the levels of arousal, valence, like/dislike, dominance and familiarity.
- These readings are analyzed to recognize the emotion of the participant.

DATASET

Current Folder

Name

- s01.mat
- s02.mat
- s03.mat
- s04.mat
- s05.mat
- s06.mat
- s07.mat
- s08.mat
- s09.mat
- s10.mat
- s11.mat
- s12.mat
- s13.mat
- s14.mat
- s15.mat
- s16.mat
- s17.mat
- s18.mat
- s19.mat
- s20.mat
- s21.mat
- s22.mat
- s23.mat
- s24.mat
- s25.mat
- s26.mat
- s27.mat
- s28.mat
- s29.mat
- s30.mat
- s31.mat
- s32.mat

DATA

	1	2
1	8.1300	4.8300
2	4.9900	2.9900
3	8.0500	7.0900
4	6.9600	5.1400
5	7.1500	5.9400
6	5.7800	3.9900
7	4.9400	4.0900
8	7.9600	6.0600
9	7.8600	4.1700
10	4.0800	5.9500
11	8.2400	6.2200
12	7.3100	3.8800
13	7.0900	3.8700
14	7.1000	6.0300
15	5.0100	1.7700
16	3.9700	6
17	6.0900	5.0300
18	8.0300	7.0600
19	8.2400	7.2400
20	6.0300	4.1200
21	4.1200	5.9900
22	4.1500	6.0600
23	3.0100	6.1500
24	1	7.3100
25	4.0100	7.1700
26	5.1400	3.0900
27	6	7.2400
28	6.0300	5
29	4.0900	6.0800
30	1	7.2700
31	4.1700	5.9600
32	3.8700	7.1500
33	4.0600	1
34	4.0500	6.2700
35	3.8800	7.2600
36	3.9100	6.9600
37	2.8100	6.1300
38	3.0500	7.0100
39	3.9900	7.1700
40	7.1500	4.0300

LABELS (Arousal, Valence)

PROBLEM STATEMENT

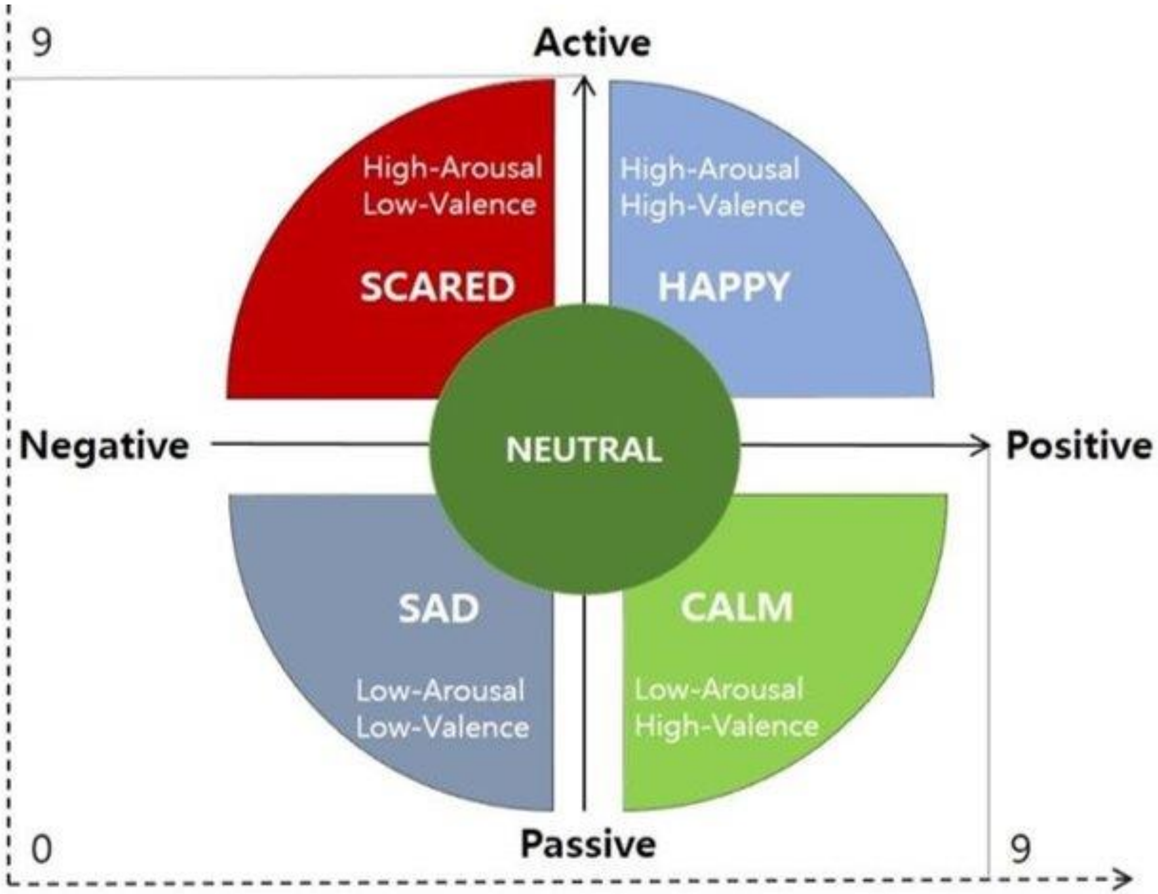
- Compute mean, kurtosis, skewness and standard deviation of the EEG signals.
- Implement grid search Random forest that gives the optimal parameters
- Plot OOB error estimates with the changes in the no. of trees. for classification task.

DATA DESCRIPTION

- We have data of 32 persons each watching 40 videos (32 x 40 = 1280 samples).
- Dataset of each person for each video is of dictionary type with two key values - (i) data (ii) labels
- Data contains 40 x 40 x 8064 matrix (40 videos * (32channels + 8 peripherals) * 8064(63 * 128))

#128 is frequency sample rate and 60 (sec video + 3 sec baseline signal)

VALENCE AROUSAL MODEL

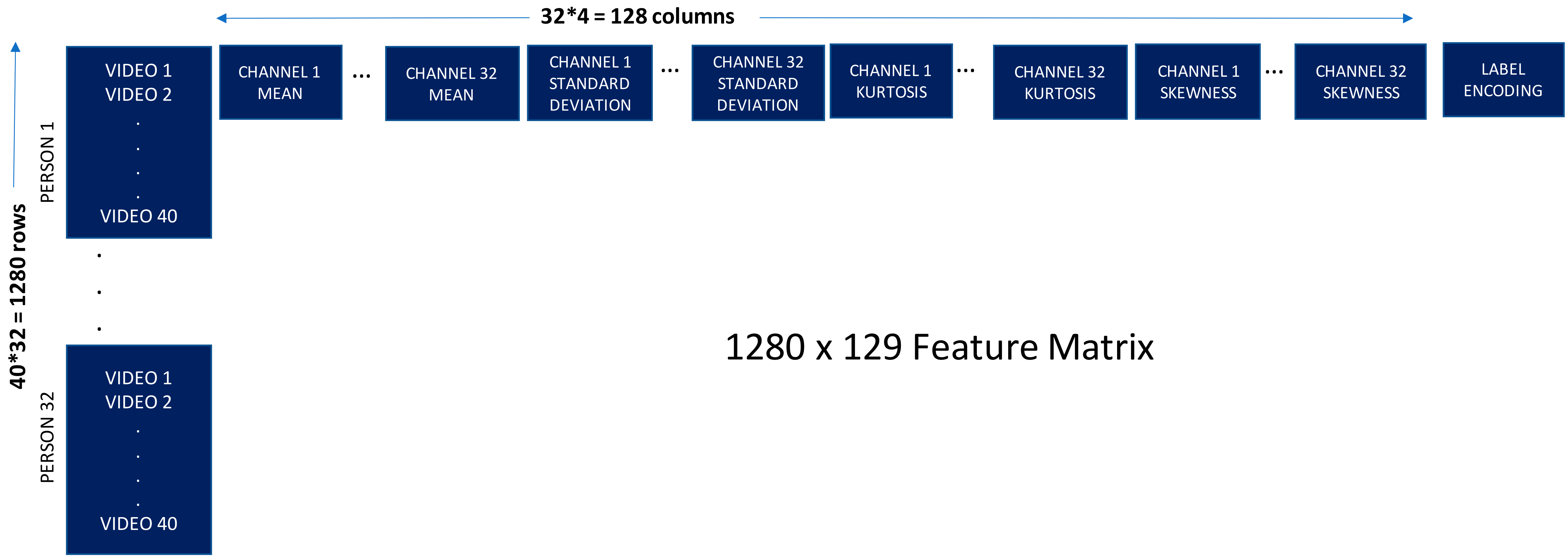


FEATURE MATRIX

- We will create feature matrix of 1280 * 129
- 32 channels x 4 features (mean, kurtosis, standard deviation, skewness) + 1 (label encoding)
- Label Encoding (Valence and arousal have values ranging from 1 to 9).

  HAPPY (1) – Valence, Arousal both are higher than threshold
  SCARED (2) – Arousal higher than threshold, Valence lower than threshold
  SAD (3) – Valence, Arousal both are lower than threshold
  CALM (4) – Arousal lesser than threshold, Valence greater than threshold

FEATURE MATRIX



FEATURE MATRIX

features

1280x129 double

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	-0.0283	-0.0153	-0.0033	-0.0565	-0.0285	0.0143	-2.5199e-04	-0.0683	-0.0107	0.0402	0.0233	-0.0185	0.0200	0.0203
2	-0.0827	-0.0314	-0.0205	-0.1132	-0.0840	-0.0263	-0.0348	-0.0517	-0.0273	0.0213	0.0047	0.0081	0.1194	0.1383
3	0.0607	0.0811	0.0598	0.0231	-0.0080	0.0451	0.0068	0.0193	-0.0322	-0.0078	-0.0636	-0.0909	-0.0672	-0.0854
4	-0.0278	-0.0119	0.0174	-0.0273	0.0495	0.0375	0.0150	0.0036	0.0448	0.0250	0.0917	0.0891	0.0439	0.0371
5	-0.0625	-0.0729	-0.0713	-0.0512	-0.0290	-0.0534	-0.0409	0.0135	0.0153	-0.0387	0.0287	0.0533	0.0134	0.0362
6	-0.0143	0.0218	0.0412	0.0336	-0.0055	-0.0184	0.0808	0.0673	0.0618	0.0272	-0.0184	0.0043	0.0209	1.0384e-04
7	-0.0481	-0.1409	-0.1792	-0.2123	0.1026	0.1598	-0.2356	-0.3296	-0.1745	-0.0435	0.2190	0.0354	-0.0840	-0.0612
8	-0.0481	-0.0656	-0.0639	-0.0368	-0.0497	-0.0693	-0.0440	-0.0261	-0.0394	-0.0092	0.0082	-0.0176	0.0401	0.0266
9	-0.0506	-0.0485	-0.0304	-0.0314	0.0017	-0.0053	-0.0110	-0.0113	0.0351	0.0347	0.0378	0.0060	0.0242	0.0319
10	-0.0588	-0.0455	-0.0273	-0.0658	-0.0108	9.4996e-04	0.0291	-0.0679	-0.0085	0.0488	0.0057	-0.0483	-0.0701	-0.0683
11	0.0267	0.0327	0.0333	0.0577	-0.0013	-0.0208	-5.1202e-04	0.0278	5.2597e-04	0.0019	-0.0311	-0.0023	-0.0073	0.0066
12	-0.0919	-0.1130	-0.1163	-0.1332	0.0309	0.0367	-0.0552	-0.1323	-0.0025	0.0467	0.1535	0.0723	0.0647	0.0450
13	-0.0517	-0.0875	-0.0848	-0.0510	0.0753	0.0206	-0.0413	-0.0591	0.0037	0.0096	0.1317	0.0756	0.0427	0.0386
14	3.8065e-04	0.0179	0.0263	0.0668	0.0287	-0.0277	0.0057	0.0682	0.0199	-0.0421	-0.0242	0.0238	-0.0069	0.0408

FEATURE MATRIX

Variables - features

features

1280x129 double

	16	117	118	119	120	121	122	123	124	125	126	127	128	129
1	0.0178	-0.1241	-0.1454	0.0958	0.0716	-0.1715	-0.1746	-0.0638	0.0017	0.0139	-0.0455	-0.0155	-0.0612	1
2	-0.0232	-0.1300	-0.0492	0.0188	0.0202	-0.0240	-0.0451	-0.0028	-0.0116	-0.1273	-0.0119	-0.0086	0.0438	1
3	0.0943	-0.1553	0.6988	0.0343	-0.0877	0.3392	-1.2229	0.9319	-0.0194	-0.0577	0.0762	-0.0021	0.0413	1
4	0.0192	-0.0447	0.0672	-0.0722	-1.6805	0.1499	0.0491	0.0325	0.0081	0.0869	-0.0241	0.0310	0.0215	1
5	-0.0419	-0.1144	-0.0940	-0.0852	0.0384	-0.1413	-0.0685	-0.0722	-0.1210	-0.1311	0.0239	-0.0941	-0.0629	2
6	-0.0197	-0.0524	-0.0142	0.0772	0.0366	-0.0565	-0.0755	-0.0839	-0.0806	-0.0816	-0.0540	-0.0355	0.0559	2
7	0.1744	-0.0269	-0.0945	0.0792	0.0455	-0.2007	-0.1726	-0.0517	0.0375	-0.2784	-0.0729	-0.0898	0.0135	2
8	-0.0408	-0.0806	0.0110	-0.1028	-0.0178	-0.1150	-0.0223	-0.0838	-0.0081	-0.1301	-0.0144	-0.0609	-0.0193	2
9	-0.1005	-0.0546	-0.0362	-0.1292	-0.0321	-0.0669	-0.0195	-0.0943	-0.0986	0.0208	0.0165	0.0134	-0.0117	4
10	0.0348	0.0328	-0.0509	-0.0861	-0.0263	-0.2384	-0.1171	-0.1344	-0.0687	-0.3189	-0.1084	-0.0963	0.0025	3
11	0.0566	-0.1322	-0.0407	0.0665	-0.0166	-0.1123	-0.0937	-0.0951	-0.0573	-0.1605	-0.0610	-0.0831	-0.0449	4

WORKING

- Defining the size of the feature matrix and creating loop to go through all the data of 32 people to collect data

```
clear;
features = zeros(1280,129);

row = 1;

% looping through all the file to collect data
for i = 1:32

    if (i < 10)
        load(['s0' num2str(i) '.mat']);
    else
        load(['s' num2str(i) '.mat']);
    end
```

WORKING

- Creating loop to go through all the trials (40 videos) and for each video – 32 channels
- Extracting features – mean, standard deviation, kurtosis and skewness.

```
for j = 1:40

    column = 1;

    for k=1:32

        channel = squeeze(data(j, k, :));

        features(row,column)=mean(channel);
        features(row,32+column)=std(channel);
        features(row,64+column)=kurtosis(channel);
        features(row,96+column)=skewness(channel);


        column=column+1;
    end
```

WORKING

Labelling the model according to valence – arousal model with threshold value of 0.45.

Plotting the labels in column 129

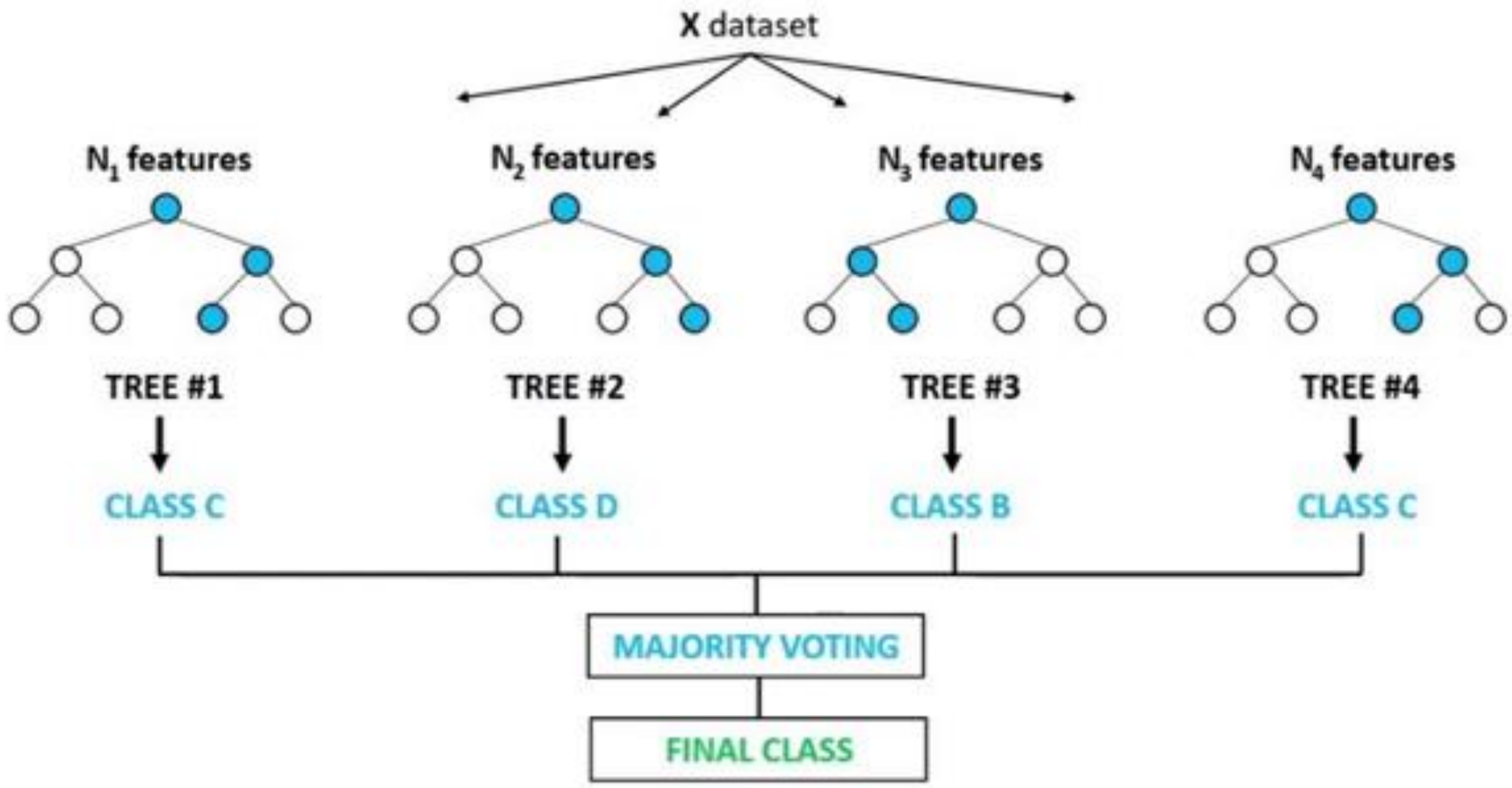
```
 if labels(j,1)>4.5   %labels(row,column) in labels.mat
     if labels(j,2)>4.5
       features(row,129)=1;
    else
        features(row,129)=2;
     end
else
   if labels(j,2)>4.5
        features(row,129)=3;
   else
      features(row,129)=4;
    end
 end
```

RANDOM FOREST CLASSIFIER

- Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.

- Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

Random Forest Classifier



RANDOM FOREST CLASSIFIER – DATASET USED

- The dataset used for random forest classifier application is the feature matrix prepared by feature extraction from the DEAP EEG signals.

WORKING

- Splitting the data into training, testing and validation.

```
X_training, X_valid, y_training, y_valid =
train_test_split(X_train, y_train, test_size=0.10, random_state=0)

print(X_training.shape)
print(X_valid.shape)
print(y_training.shape)
print(y_valid.shape)
```

```
(1152, 128)
(128, 128)
(1152,)
(128,)
```

WORKING

- Applying Random Forest classifier to train and test the model's accuracy.

```
rf_clf = RandomForestClassifier()
rf_clf.fit(X_training, y_training)
pred_rf = rf_clf.predict(X_valid)
report=classification_report(pred_rf,y_valid)
con=confusion_matrix(pred_rf,y_valid)
acc_rf = accuracy_score(y_valid, pred_rf)
print(acc_rf)
# pred_rf
```

Accuracy obtained - 0.421875

WORKING

- Checking and finding the optimal parameters for model.

```
rf_clf = RandomForestClassifier()

parameters = {"n_estimators": [4, 5, 6, 7, 8, 9, 10, 15],
              "criterion": ["gini", "entropy"],
              "max_features": ["auto", "sqrt", "log2"],
              "max_depth": [2, 3, 5, 10],
              "min_samples_split": [2, 3, 5, 10],
              "min_samples_leaf": [1, 5, 8, 10]
             }

grid_cv = GridSearchCV(rf_clf, parameters, scoring = make_scorer(accuracy_score))
grid_cv = grid_cv.fit(X_training, y_training)

print("Our optimized Random Forest model is:")
grid_cv.best_estimator_
```

```
Our optimized Random Forest model is:

RandomForestClassifier(criterion='entropy', max_depth=5, min_samples_split=5,
                       n_estimators=7)
```

WORKING

- Predicting accuracy on the basis of obtained optimal parameters.

```
rf_clf = grid_cv.best_estimator_
rf_clf.fit(X_train, y_train)
pred_rf = rf_clf.predict(X_valid)
report=classification_report(pred_rf,y_valid)
con=confusion_matrix(pred_rf,y_valid)
acc_rf = accuracy_score(y_valid, pred_rf)
print(acc_rf)
```

Accuracy obtained - 0.4921875

	precision	recall	f1-score	support
1	0.74	0.39	0.51	87
2	0.17	0.33	0.23	15
3	0.38	0.57	0.46	23
4	0.16	1.00	0.27	3
accuracy			0.43	128
macro avg	0.36	0.57	0.37	128
weighted avg	0.59	0.43	0.46	128

KNN CLASSIFIER

	precision	recall	f1-score	support
1	0.41	0.74	0.53	95
2	0.38	0.25	0.30	59
3	0.34	0.20	0.25	65
4	0.22	0.05	0.09	37
accuracy			0.39	256
macro avg	0.34	0.31	0.29	256
weighted avg	0.36	0.39	0.34	256

PCA – PRINCIPAL COMPONENT ANALYSIS

- PCA is the process of computing the principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest.

- It is used to explain the variance-covariance structure of a set of variables through linear combinations.

- It is often used as a dimensionality-reduction technique.

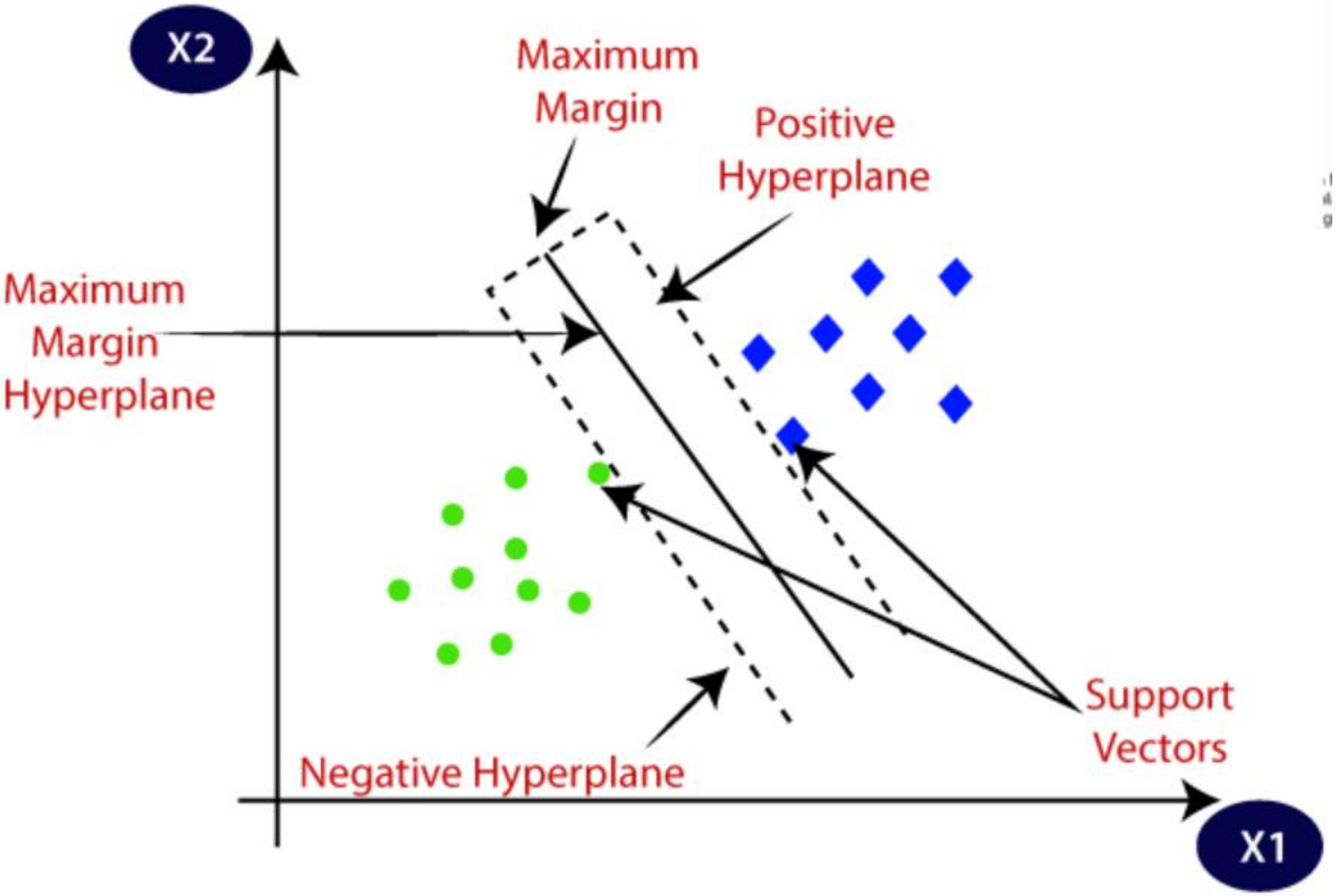
SVM CLASSIFIER

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems.

- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.

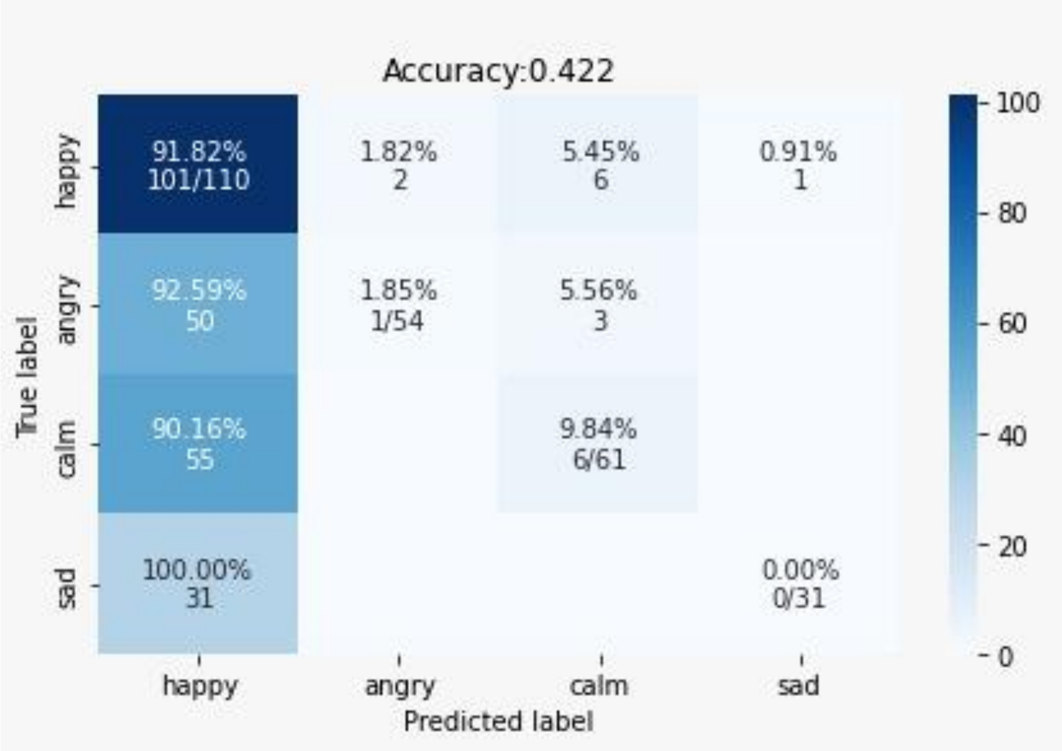
- This best decision boundary is called a hyperplane.

SVM CLASSIFIER



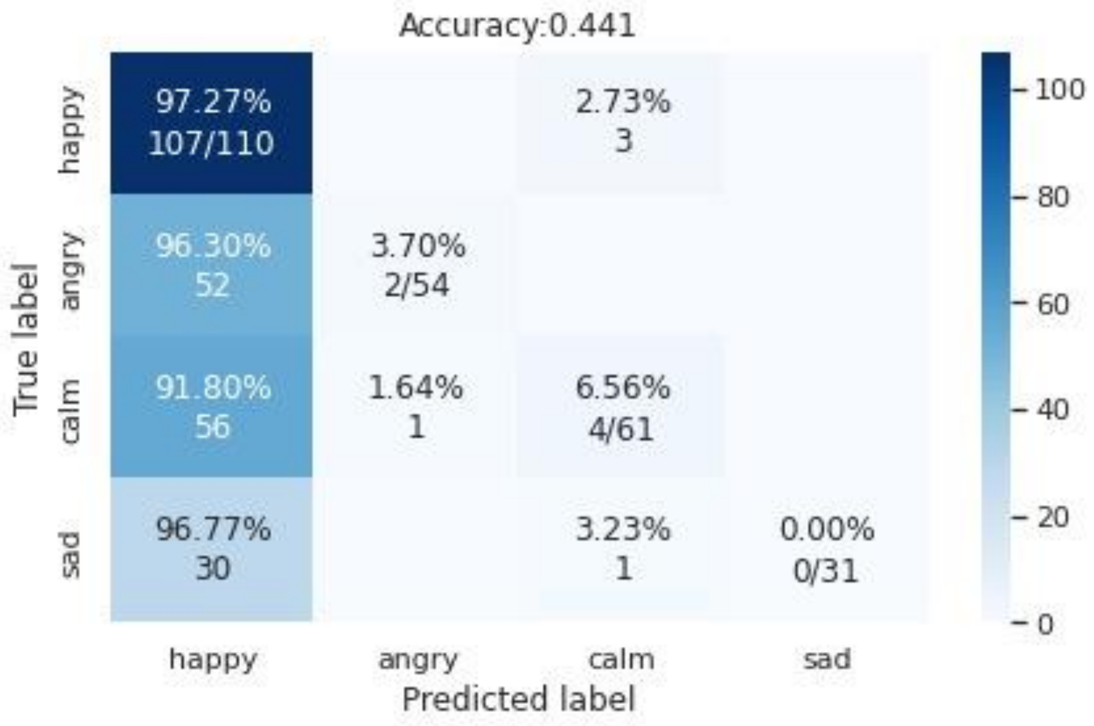
SVM CLASSIFIER – RBF kernel

	precision	recall	f1-score	support
1	0.43	0.92	0.58	110
2	0.33	0.02	0.04	54
3	0.40	0.10	0.16	61
4	0.00	0.00	0.00	31
accuracy			0.42	256
macro avg	0.29	0.26	0.19	256
weighted avg	0.35	0.42	0.30	256



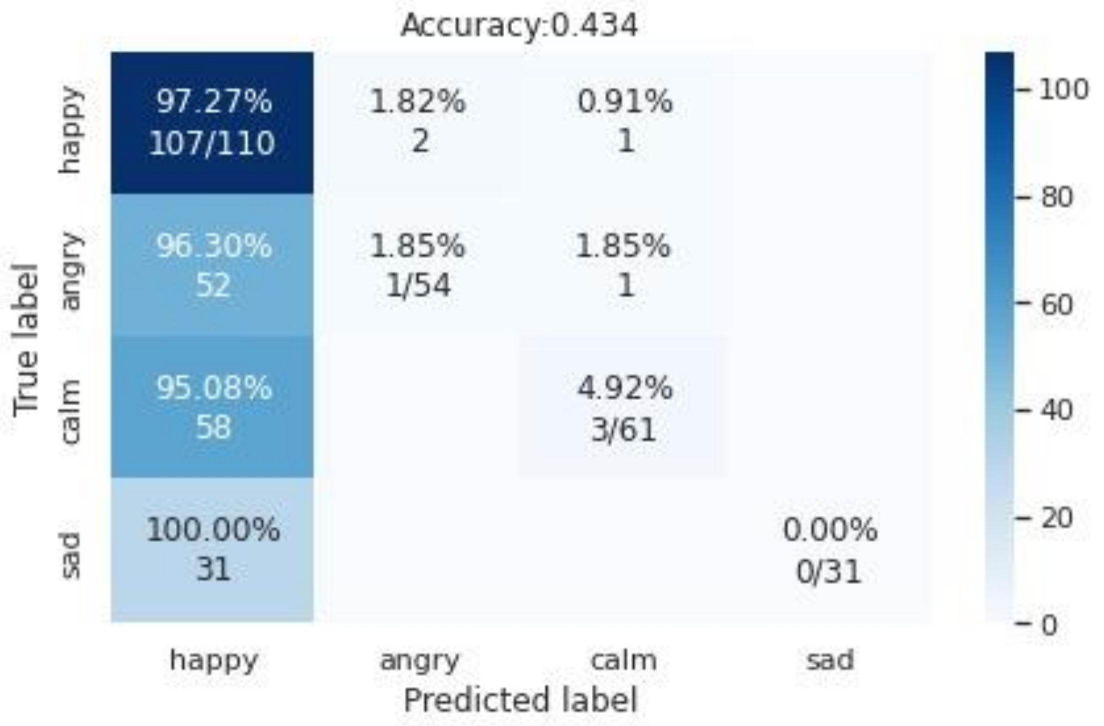
SVM CLASSIFIER – polynomial kernel

	precision	recall	f1-score	support
1	0.44	0.97	0.60	110
2	0.67	0.04	0.07	54
3	0.50	0.07	0.12	61
4	0.00	0.00	0.00	31
accuracy			0.44	256
macro avg	0.40	0.27	0.20	256
weighted avg	0.45	0.44	0.30	256



SVM CLASSIFIER – linear kernel

	precision	recall	f1-score	support
1	0.43	0.97	0.60	110
2	0.33	0.02	0.04	54
3	0.60	0.05	0.09	61
4	0.00	0.00	0.00	31
accuracy			0.43	256
macro avg	0.34	0.26	0.18	256
weighted avg	0.40	0.43	0.29	256



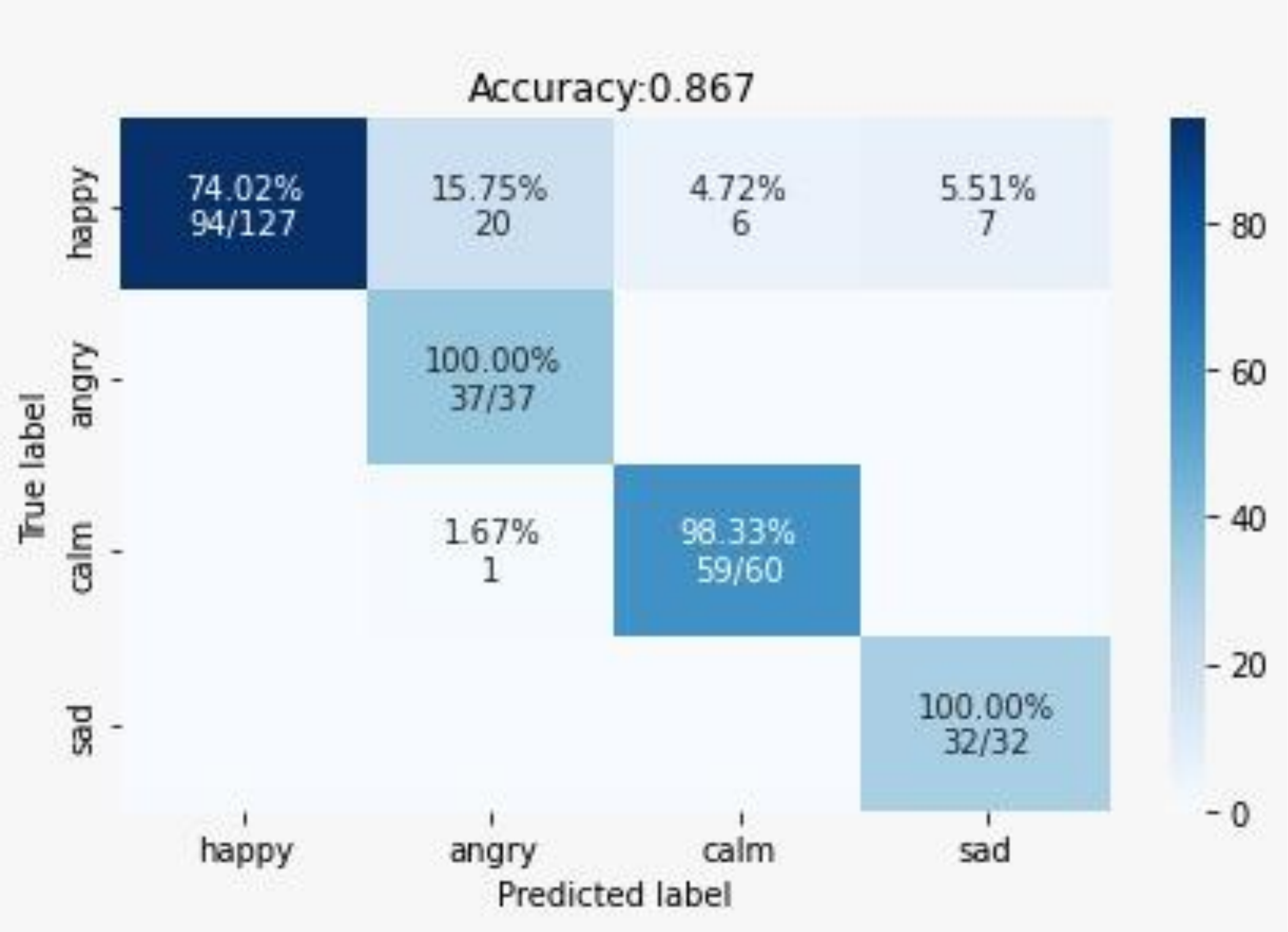
Random Forest – third stage optimized parameters

- Checking and finding the optimal parameters for model.

```
Our optimized Random Forest model is:
RandomForestClassifier(max_depth=10, max_features='sqrt', min_samples_split=5,
                       n_estimators=30)
```

	precision	recall	f1-score	support
1	1.00	0.74	0.85	127
2	0.64	1.00	0.78	37
3	0.91	0.98	0.94	60
4	0.82	1.00	0.90	32
accuracy			0.87	256
macro avg	0.84	0.93	0.87	256
weighted avg	0.90	0.87	0.87	256

CONFUSION MATRIX



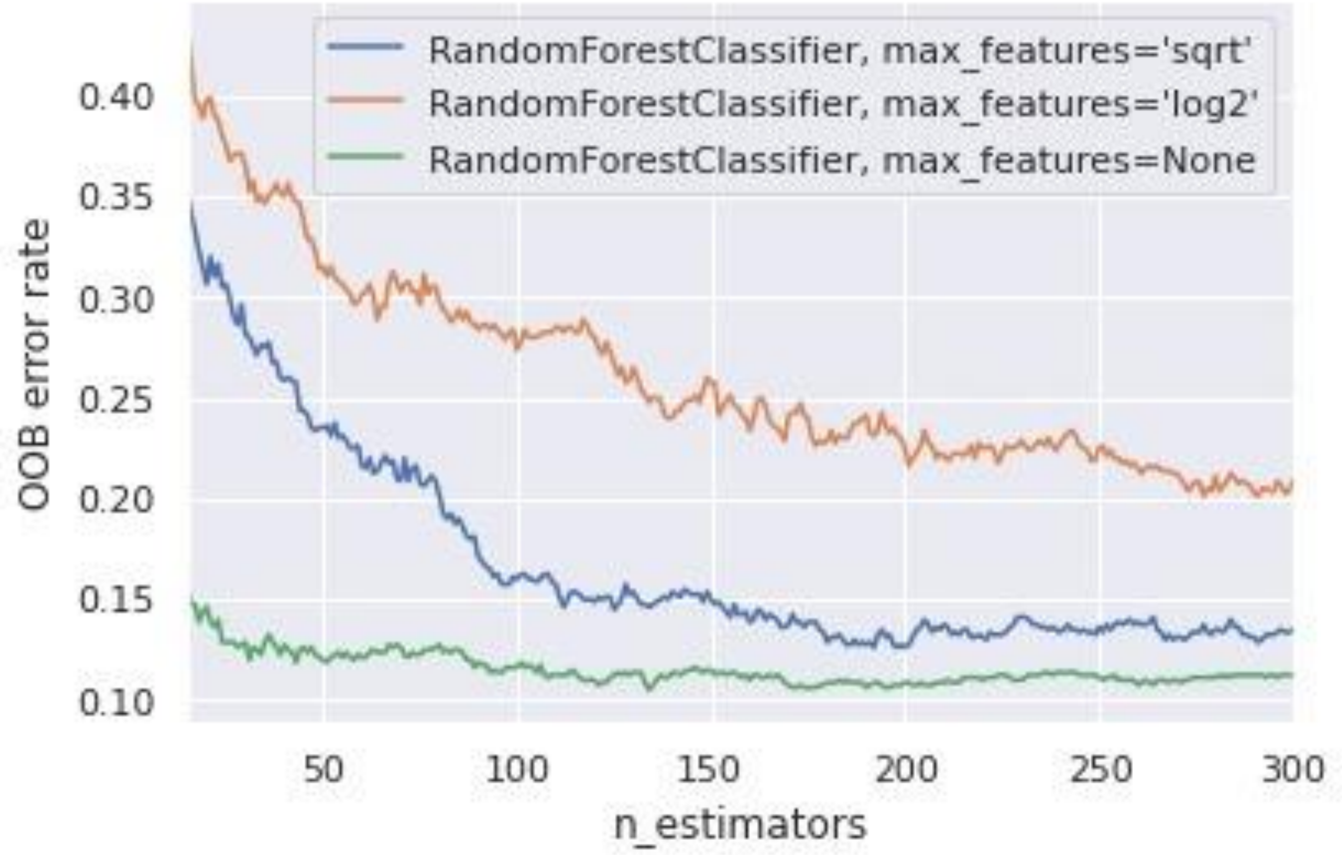
OOB ERROR

- Out-of-bag error is one of the methods for validating the machine learning model.

- OOB means they are the error estimates obtained by predicting on data that was not (or atleast should not be) part of the learning phase.

OOB ERROR

OOB error estimates with the changes in the no. of trees. for classification task.



MORE FEATURES IN THE FEATURE MATRIX

OVERALL FEATURES

```
features(row,column)=mean(channel);
features(row,32+column)=var(channel);
features(row,64+column)=std(channel);
features(row,96+column)=kurtosis(channel);
features(row,128+column)=skewness(channel);
features(row,160+column)=zerocrossrate(channel);
```

NEW FEATURE MATRIX DIMENSIONS

1280 rows x 193 columns

(32 persons * 40 vides) x (6 features * 32 channels + encoded label)

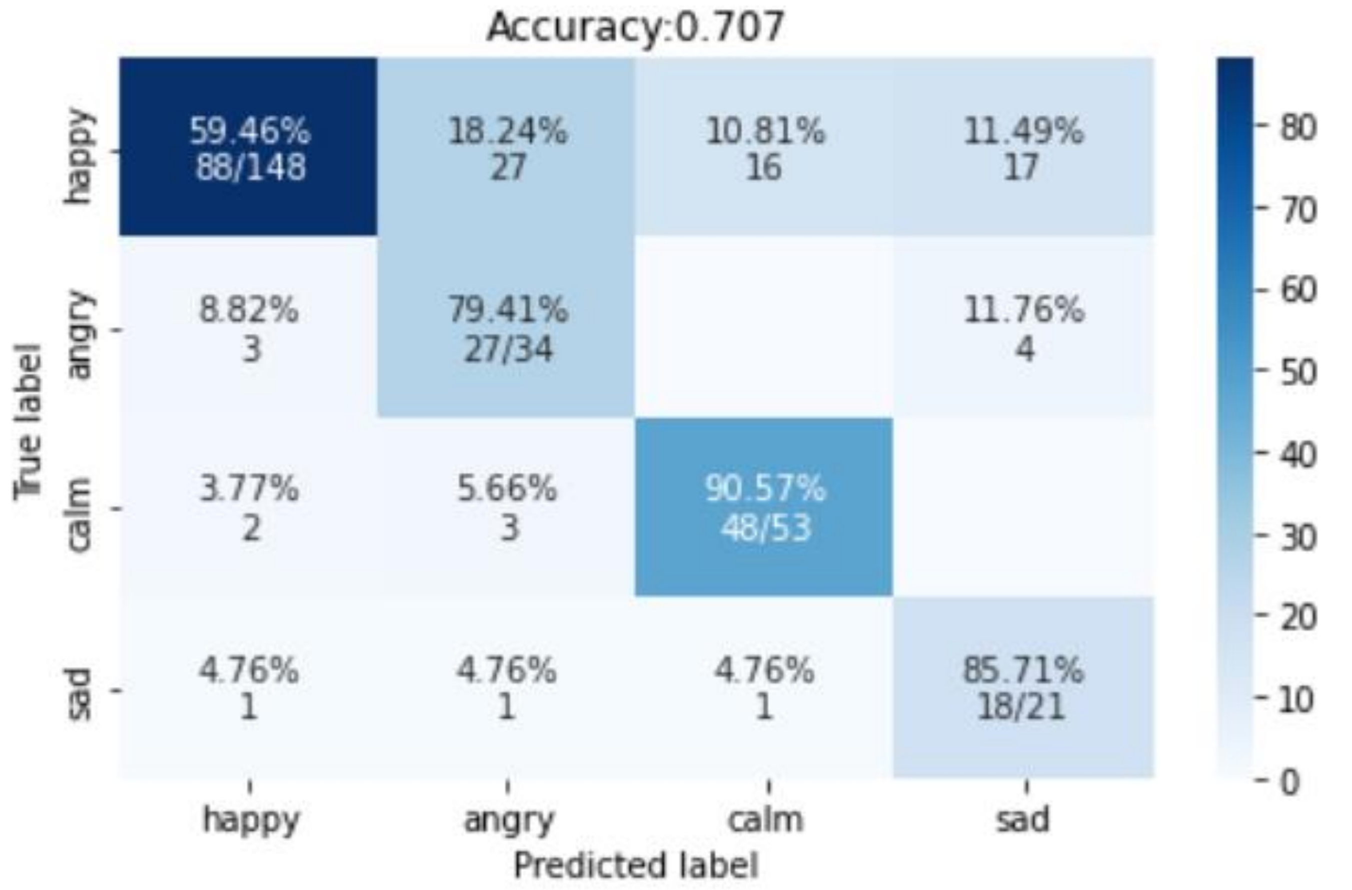
Random Forest – optimized parameters

- Checking and finding the optimal parameters for model.
- This time a new feature matrix is introduced.

Our optimized Random Forest model is:
RandomForestClassifier (max_depth=10, max_features='log2', min_samples_leaf=5, min_samples_split=5, n_estimators=30)

	precision	recall	f1-score	support
1	0.94	0.59	0.73	148
2	0.47	0.79	0.59	34
3	0.74	0.91	0.81	53
4	0.46	0.86	0.60	21
accuracy			0.71	256
macro avg	0.65	0.79	0.68	256
weighted avg	0.79	0.71	0.72	256

Random Forest – optimized parameters



MORE FEATURES IN THE FEATURE MATRIX

OVERALL FEATURES

```
% these are the column vectors for collecting the colummn
channel = squeeze(data(j, k, :));
features(row,column)=mean(channel);
features(row,32+column)=std(channel);
features(row,64+column)=kurtosis(channel);
features(row,96+column)=skewness(channel);
%new features
features(row,128+column)=median(channel);
features(row,160+column)=var(channel);
features(row,192+column)=max(channel, [], 'all');
features(row,224+column)=min(channel, [], 'all');
features(row,256+column)=range(channel, 'all');
```

NEW FEATURE MATRIX DIMENSIONS

1280 rows x 289 columns

(32 persons * 40 vides) x (9 features * 32 channels + encoded label)

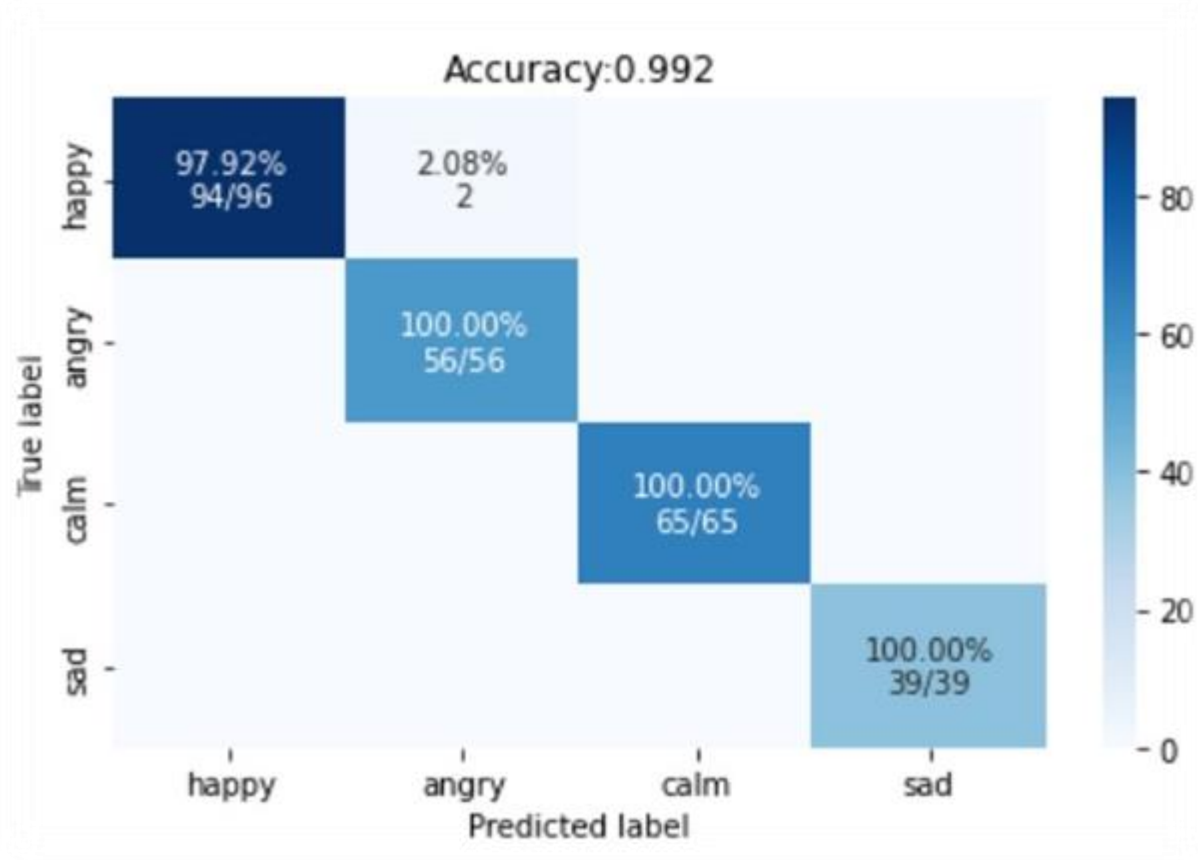
PSD EXTRACTION

```
column=288;
for k=1:32

    [C,L] = wavedec(squeeze(data(j,k,:)),7,'db1');
    [ccD1,ccD2,ccD3,ccD4,ccD5]=detcoef(C,L,2:6);
    column=column+1;

    PSD=pburg(ccD1,4);
    features(row,column)=mean(PSD);
    column=column+1;

    PSD=pburg(ccD2,4);
    features(row,column)=mean(PSD);
    column=column+1;

    PSD=pburg(ccD3,4);
    features(row,column)=mean(PSD);
    column=column+1;

    PSD=pburg(ccD4,4);
    features(row,column)=mean(PSD);
    column=column+1;

    PSD=pburg(ccD5,4);
    features(row,column)=mean(PSD);
end
```

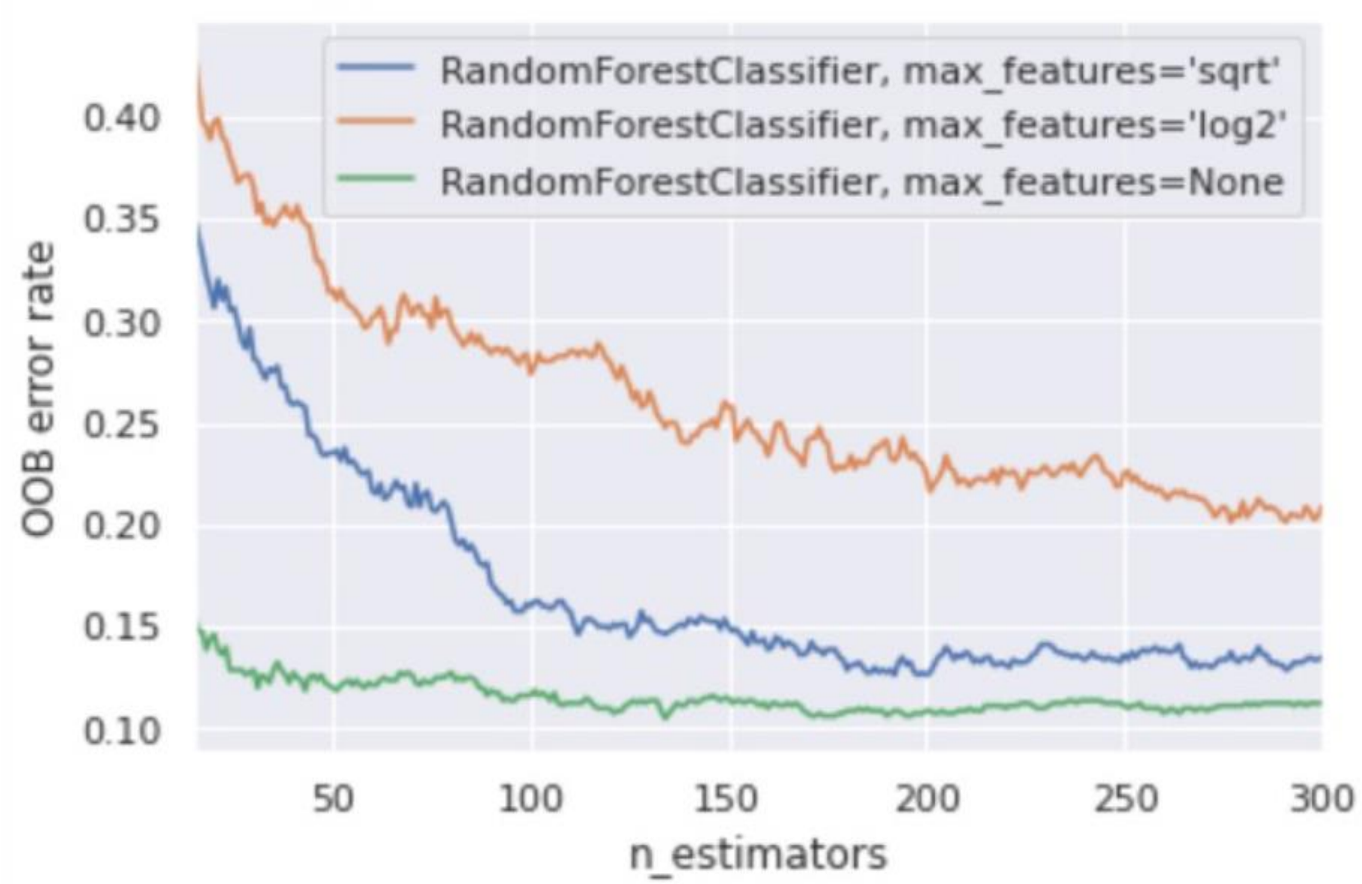
Random Forest – optimized parameters

	precision	recall	f1-score	support
1	1.00	0.98	0.99	96
2	0.97	1.00	0.98	56
3	1.00	1.00	1.00	65
4	1.00	1.00	1.00	39
accuracy			0.99	256
macro avg	0.99	0.99	0.99	256
weighted avg	0.99	0.99	0.99	256

Random Forest – optimized parameters



Random Forest – optimized parameters



THANK YOU !

Dr. Shyama Prasad Mukherjee International Institute of Information Technology, Naya Raipur