# Mental State Recognition using EEG Signal

## GROUP - 1

**Group Members :**

| | |
|---|---|
| Harsh Goyal | IIT2018114 |
| Aaditya Gadhave | IIT2018144 |
| Sourabh Gupta | IIT2018149 |
| Meet Singh Gambhir | IIT2018158 |
| Tushar Atrey | IIT2018159 |

**Instructed By : Dr.Anupam Agrawal, G.C. Jana**

# Introduction

Human machine interaction is often considered superior than that of the human Human interaction, as sound and visuals constitute voice recognition, human activity classification, facial recognition, can be used to get more information about an individual when the former one is used than the latter one . Though, with the availability of sensors to gather data that the human body cannot, interaction with machines can often exceed the abilities of the natural human experience.

An example of this is the consideration of electroencephalographic brainwaves. The brain, based on what a person is thinking, feeling, or doing, has a unique pattern of electrical activity that emerges as a consequence of the aggregate firing patterns of billions of individual neurons [1, 2]

These electrical signals can, in principle, be detected and processed to infer the state of the brain and, by extension, the mental state of a given subject. Besides clinical applications, this possibility is also useful, e.g., for brain-machine interfacing. This work focuses on the process of feature extraction, selection and formatting in order to achieve improved classification accuracy of EEG signals.

More specifically, the main contribution is a framework to perform classification of these signals, based on

(i) the extraction of a large number of static statistical features of the data,

(ii) automated feature selection

(iii) representation of the selected attributes as a 2D matrix

(iv) then used to predict for the new data of the same kind.

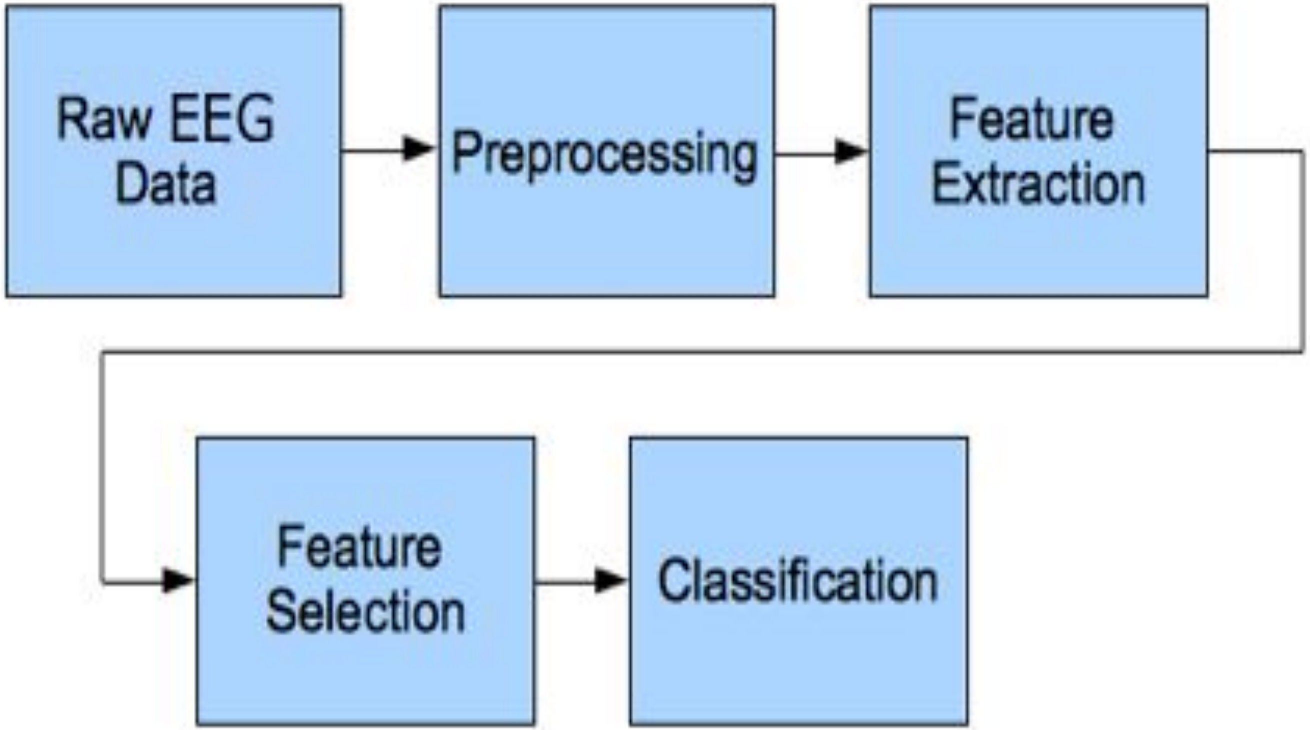
# Literature Review

| Author Name | Paper Title | Conference/Journal Paper with year | Method/approach used | Application domain | Dataset used | Achieved Performance | Advantages and Disadvantages | Feature scope |
|---|---|---|---|---|---|---|---|---|
| Jodie Ashford Jordan J. Bird Felipe Campelo Diego R. Faria | Classification of EEG Signals Based on Image Representation of Statistical Features | UK Workshop on Computational Intelligence, 30 August 2019 | Convolutional Neural Networks, Electroencephalography | EEG-based | DEAP dataset | CNN achieved 81.41% accuracy for valence and 73.35% for arousal | Advantage: The extracted features were reduced to optimum number of features Disadvantage:EEG recorded during a cognitive task is relatively easy to separate from a baseline EEG than EEG signals recorded in two different cognitive tasks | Extending the application of the proposed method to various image recognition system for predicting emotions |

# Problem Statement and Scope

Our main motive is to predict the three-class mental state of any subject in which they experience either relaxation, concentration, or neutral states.  Therefore we first extract features from the pre-processed EEG signals data, then we select the important features from it using 5 different methods of 4 subjects.These selected features are then normalised and reshaped, which can be expressed as a grayscale image Then we make it pass through a SVM model which allows us to classify the mental state of the subject.

# Methodology and Flowchart



# Implementation

## 1. Feature Generating :

Firstly, an available training set of EEG signals is preprocessed. The data is assumed to contain the time series related to one or more electrodes, within a given experimental time frame, labelled in terms of **three distinct mental states (relaxed, concentrating, and neutral)** that the subjects were keeping during data collection . From these signals a number of statistical features are extracted , resulting in a high dimensional attribute space.

## 2. Feature Extraction :

Due to the temporal, auto-correlated nature of the EEG waves, single-point features cannot generally provide enough information for good rules to be generated by machine learning models. In this work we follow the approach of extracting statistical features based on sliding time windows [3, 4]. More specifically, the EEG signal is divided into a sequence of windows of length one second, with consecutive windows overlapping by 0.5 seconds, e.g., $[(0s - 1s), [0.5s - 1.5s), [1s - 2s), \ldots])$.

**The following statistical features were generated for each time window and for this we have implemented the following functions:**

## Functions used :

- feature_mean : Takes in the matrix and returns the mean value of each signal for the full time window
- Feature_mean_d : Takes in two matrices denoting first half and second half of any time slice and computes the change in the means (backward difference) of all signals between the first and second half-windows, i.e mean(arg1)-mean(arg2)
- feature_mean_q : Takes in four matrices denoting four quarters of any time slice and computes the mean values of each signal for each quarter-window, plus the paired differences of means of each signal for the quarter-windows i.e Σ(feature_mean(arg(i)) - (feature_mean(arg(j)) for all 1<=i,j<=4
- feature_stddev : Takes in the matrix and returns the standard deviation of each signal for the full time window
- feature_stddev_d :Takes in two matrices denoting first half and second half of any time slice and computes the change in standard deviation (backward difference) of all signals between the first and second half-windows, i.e std(arg1)-std(arg2).
- feature_moments : Takes in the matrix and computes the 3rd and 4th standardised moments about the mean (i.e., skewness and kurtosis) of each signal, for the full time window.
- feature_max : Takes in the matrix and returns the max value of each signal for the full time window.

- Feature_max_d : Takes in two matrices denoting first half and second half of any time slice and computes the change in the max value (backward difference) of all signals between the first and second half-windows, i.e mean(arg1)-mean(arg2).
- feature_max_q : it will take four quarter windows as input,computes the maximum value for each quarter window and then paired the difference of maximum values for the quarter-windows.
- feature_min : Returns the minimum value of each signal for the full time window.
- feature_min_d: Computes the change in min values of all signals between the first and second half-windows, min(h2) - min(h1)
- feature_min_q: Computes the min values for each quarter-window and paired differences of min values for the quarter-windows.
- feature_covariance_matrix: Computes the lower triangular elements of the covariance matrix of the signals because the covariance matrix is symmetric.
- feature_eigenvalues : Computes the eigenvalues of the covariance matrix.
- feature_logcov: Computes the matrix logarithm of the covariance matrix.
- calc_feature_vector: Calculates all previously defined features and concatenates everything into a single feature vector.
- generate_feature_vectors_from_samples: Reads data from CSV file and extracts statistical features for each time window of width "period".
- Gen_training_matrix: This is the main function which calls on all the calculating features and generate the final matrix

### 3. Feature Selection:

Based on all the features we get, we select only the relevant ones. Scikit-learn has made it pretty much easy for us to make the feature selection. There are a lot of ways in which we can think of feature selection, but most feature selection methods can be divided into three major buckets:

- ***Filter based:*** We specify some metric and based on that filter features. An example of such a metric could be correlation/chi-square.
- ***Wrapper-based:*** Wrapper methods consider the selection of a set of features as a search problem. Example: Recursive Feature Elimination.
- ***Embedded:*** Embedded methods use algorithms that have built-in feature selection methods. For instance, Lasso and RF have their own feature selection methods.

We used 5 methods for feature selection and we then have chosen the one with highest accuracy :

1. **Pearson Correlation :**This is a filter-based method.We check the ***absolute value of the Pearson's correlation*** between the target and numerical features in our dataset. We keep the top n features based on this criterion.

$$r = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2 \sum(y-\bar{y})^2}}$$

2. **Chi-Squared :** This is another filter-based method.In this method, we calculate the chi-square metric between the target and the numerical variable and only select the variable with the maximum chi-squared values. We therefore calculate the chi-squared value. To do this, we first find out the values we would expect to be falling in each bucket if there was indeed independence between the two categorical variables. We then multiply the row sum and the column sum for each cell and divide it by total observations. We use the following formula :

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

3. **Recursive Feature Elimination :** This is a wrapper based method. As I said before, wrapper methods consider the selection of a set of features as a search problem. The goal of recursive feature elimination (RFE) is to select features by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features and the importance of each feature is obtained either through a `coef_` attribute or through a `feature_importances_` attribute. Then, the least important features are pruned from the current set of features. That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached.

4. **Random Forest :** This is an Embedded method.We can also use RandomForest to select features based on feature importance.We calculate feature importance using node impurities in each decision tree. In Random forest, the final feature importance is the average of all decision tree feature importance.

5. **LightGBM :** We have also used a LightGBM as it has a feature_importances_ attribute.

# Classification Algorithms

## 1. Support vector machine

**Fundamental reason for using the support vector machine is to classify/predict , .i.e. Firstly using the training data (combination of sample input and output ) , in order to train the svm and then , for some set of input , get the output predicted technically also known as the supervised learning.**
**The SVM is used to get classification among the different classes among the output and thus is the discrete classification and not the continuous one.**

Saying that there are **n features** in the input and then the main objective of the **svm algorithm is to find a plane in n dimensional that classifies the data points , which are themself n dimensional. The classification is taken as either the data point belongs inside or outside of the plane.**

In order to separate the classes of data , there can be many possible different planes that can be used in doing so. Task comes to get that plane that has the maximum margin , so the large margin classifier, that is  distance between data points of both classes should be maximum w.r.t the plane .As we are maximizing marginal distance so as to get some extra reinforcement so that on the future data points we will be getting less error. And thus will be classified with more amount of confidence.

## 2. KNN

K- Nearest Neighbors is a Supervised machine learning algorithm as the target variable is known Non parametric as it does not make an assumption about the underlying data distribution pattern Lazy algorithm as KNN does not have a training step. All data points will be used only at the time of prediction. With no training step, prediction step is costly. An eager learner algorithm eagerly learns during the training step. It is used for both Classification and Regression. It uses feature similarity to predict the cluster that the new point will fall into.

# Experimental Data Description

**EEG-Feature -Generation**

https://github.com/jordan-bird/eeg-feature-generation/tree/master/dataset/original_data

***Timestamps :-*** *is that instance of me where the data is collected*

***TP9,AF7,AF8,TP10, :-*** *These are the four sensors used up for collecting dataset*

***Right Aux :-*** *RAW brainwaves for the auxiliary USB sensor (MU-02 and MU-03 only)*

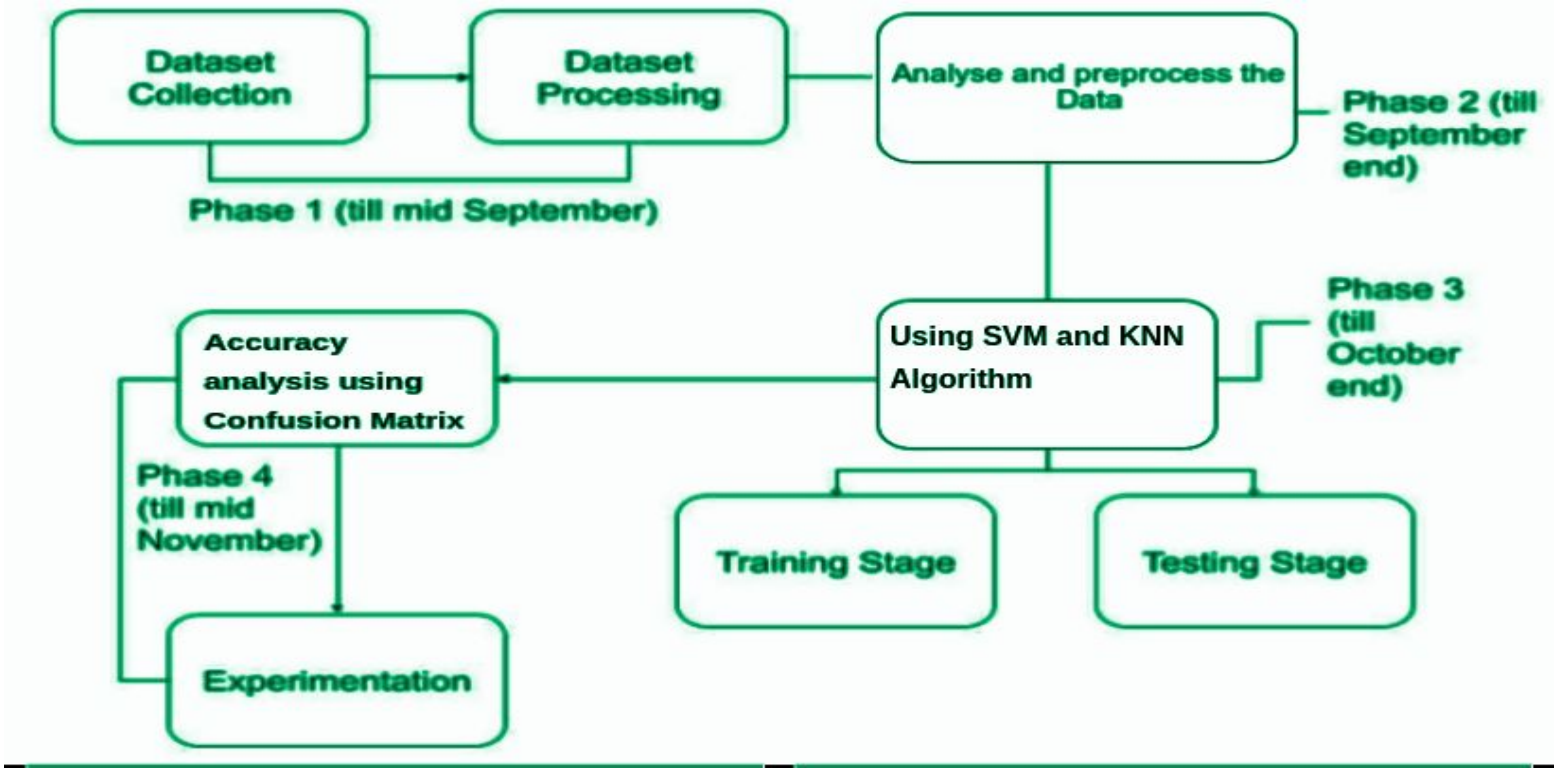
```
timestamps,TP9,AF7,AF8,TP10,Right AUX
1533222559.839,59.105,28.320,15.137,12.207,54.199
1533222559.843,62.012,30.273,43.945,11.719,79.102
1533222559.847,44.922,30.273,-97.656,11.230,32.715
1533222559.851,28.809,27.832,-110.352,9.277,29.785
1533222559.855,36.156,28.809,-73.242,11.230,50.781
1533222559.859,57.617,36.133,-17.090,16.113,37.109
1533222559.862,74.219,38.574,42.480,27.832,20.020
1533222559.866,57.129,33.691,-30.273,34.180,28.320
1533222559.870,23.949,29.785,-101.074,13.184,64.941
1533222559.874,20.020,33.203,-92.285,9.277,40.527
1533222559.878,41.504,33.203,-24.902,15.625,-0.488
1533222559.882,51.758,29.785,68.359,14.160,37.598
1533222559.886,47.852,34.180,-29.297,21.973,55.176
1533222559.890,34.180,36.133,-122.070,17.578,59.570
```

# Language and Libraries

**Choice of Language :** python

**Used Libraries :** numpy, scipy, os, sys,matplotlib,sklearn, re , pandas , csv ,lightgbm, Pillow = 2.2.2 , nltk

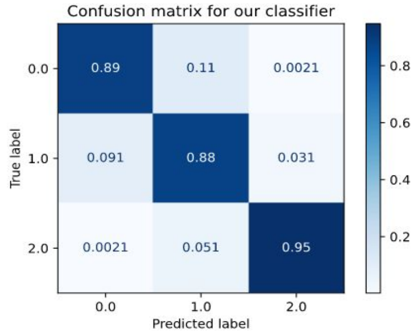
# Activity Chart



# Result : SVM Classifier

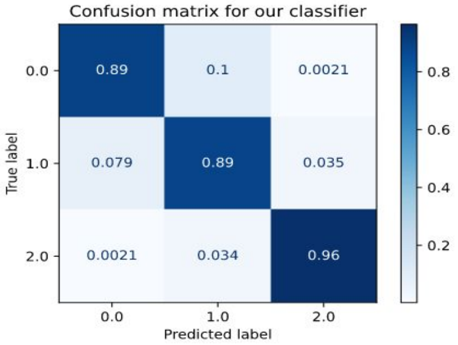
## 1. Pearson Correlation

```
-------------------------------------- Pearson Correlation ------------------------
(2479, 850)
2479
Training Accuracy: 100.0 %
Test Accuracy: 90.34289713086075 %
         TP     FP      TN      FN

0.0     417     54     913      45
1.0     424     59     869      77
2.0     450     25     938      16

Classification Report

              precision    recall  f1-score   support

         0.0       0.90      0.89      0.89       471
         1.0       0.85      0.88      0.86       483
         2.0       0.97      0.95      0.96       475

    accuracy                           0.90      1429
   macro avg       0.90      0.90      0.90      1429
weighted avg       0.90      0.90      0.90      1429
```



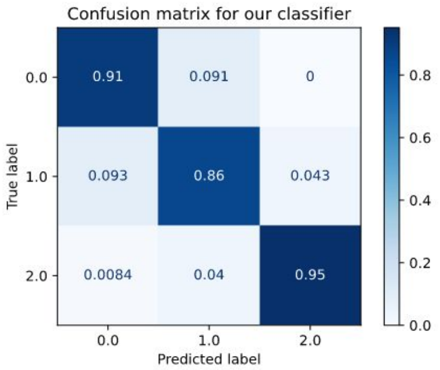

## 2. Chi - Squared

```
----------------------------------- Chi-Squared ----------------------
(2479, 850)
2479
Training Accuracy: 100.0 %
Test Accuracy: 91.46256123163052 %
         TP     FP      TN      FN

0.0     421     50     919      39
1.0     428     55     881      65
2.0     458     17     936      18

Classification Report

              precision    recall  f1-score   support

         0.0       0.92      0.89      0.90       471
         1.0       0.87      0.89      0.88       483
         2.0       0.96      0.96      0.96       475

    accuracy                           0.91      1429
   macro avg       0.92      0.91      0.91      1429
weighted avg       0.91      0.91      0.91      1429
```



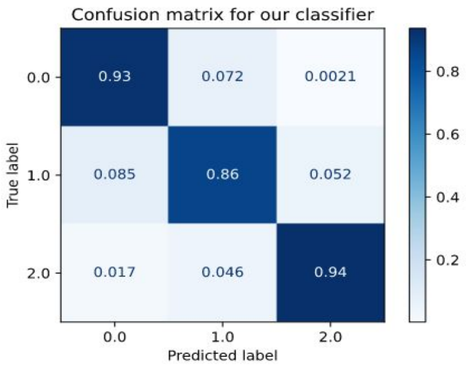
## 3. Recursive Feature Elimination

```
------------------------------------- Recursive Feature Elimination -----------------
(2479, 850)
2479
Training Accuracy: 100.0 %
Test Accuracy: 90.76277116864941 %
         TP      FP      TN      FN

0.0      428     43      909     49
1.0      417     66      884     62
2.0      452     23      933     21

Classification Report

              precision    recall  f1-score   support

         0.0       0.90      0.91      0.90       471
         1.0       0.87      0.86      0.87       483
         2.0       0.96      0.95      0.95       475

    accuracy                           0.91      1429
   macro avg       0.91      0.91      0.91      1429
weighted avg       0.91      0.91      0.91      1429
```



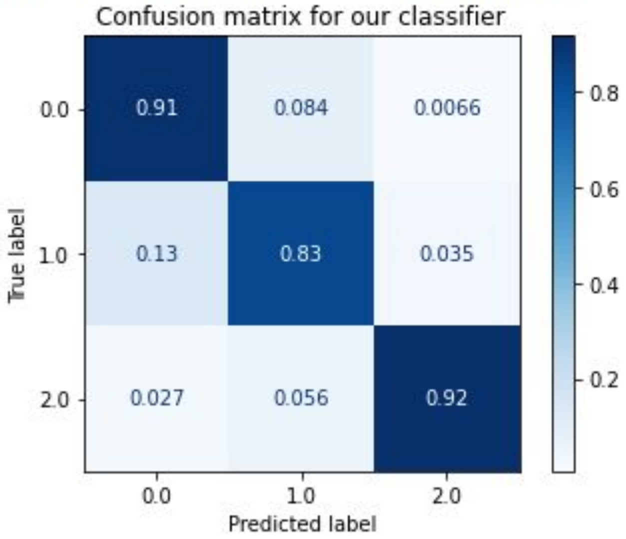

## 4. Random Forest

```
------------------------------------- Random Forest ---------------------
(2479, 206)
2479
Training Accuracy: 98.4747378455672 %
Test Accuracy: 90.83275017494752 %
         TP      FP      TN      FN

0.0      436     35      909     49
1.0      417     66      890     56
2.0      445     30      928     26

Classification Report

              precision    recall  f1-score   support

         0.0       0.90      0.93      0.91       471
         1.0       0.88      0.86      0.87       483
         2.0       0.94      0.94      0.94       475

    accuracy                           0.91      1429
   macro avg       0.91      0.91      0.91      1429
weighted avg       0.91      0.91      0.91      1429
```



## 5. Light GBM

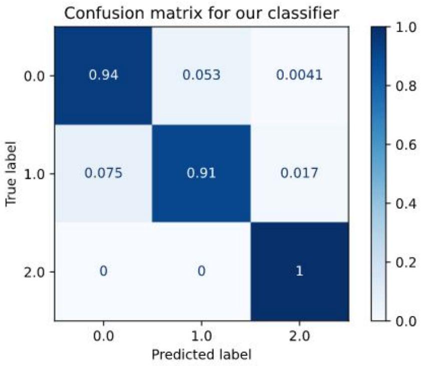
```
(2479, 199)
2479
Training Accuracy: 97.04480457578646 %
Test Accuracy: 88.52344296710987 %
```



# Result : KNN Classifier

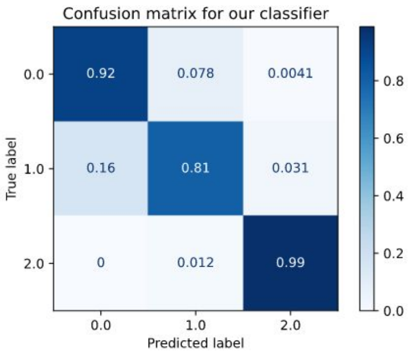
1. **Pearson Correlation**

```
--------------------------------------- Pearson Correlation ---------------
Test Accuracy: 95.16129032258065 %
         TP      FP      TN      FN

0.0      232     14      480     18
1.0      218     22      491     13
2.0      258     0       481     5

Classification Report

              precision    recall  f1-score   support

         0.0       0.93      0.94      0.94       246
         1.0       0.94      0.91      0.93       240
         2.0       0.98      1.00      0.99       258

    accuracy                           0.95       744
   macro avg       0.95      0.95      0.95       744
weighted avg       0.95      0.95      0.95       744
```



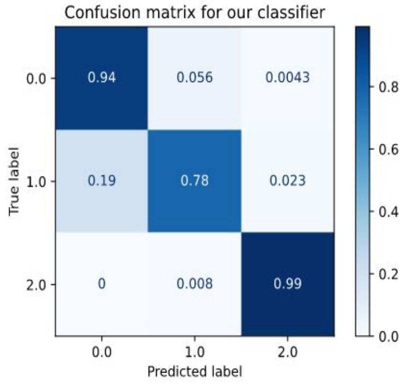

2. **Chi - Squared**

```
--------------------------------------- Chi-Squared ----------------------
Test Accuracy: 90.18817204301075 %
         TP      FP      TN      FN

0.0      224     20      458     42
1.0      208     50      464     22
2.0      239     3       493     9

Classification Report

              precision    recall  f1-score   support

         0.0       0.84      0.92      0.88       244
         1.0       0.90      0.81      0.85       258
         2.0       0.96      0.99      0.98       242

    accuracy                           0.90       744
   macro avg       0.90      0.90      0.90       744
weighted avg       0.90      0.90      0.90       744
```



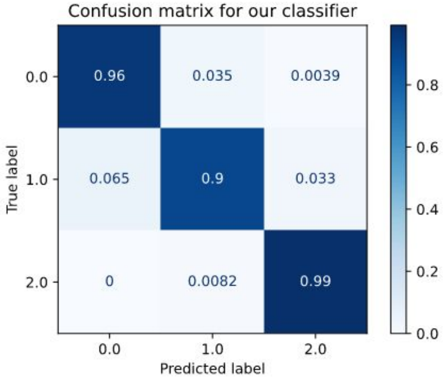
## 3. Recursive Feature Elimination

```
------------------------------------- Recursive Feature Elimination -----------------
Test Accuracy: 90.18817204301075 %
         TP     FP      TN      FN

0.0      217    14      462     51
1.0      206    57      466     15
2.0      248    2       487     7

Classification Report

              precision    recall  f1-score   support

         0.0       0.81      0.94      0.87       231
         1.0       0.93      0.78      0.85       263
         2.0       0.97      0.99      0.98       250

    accuracy                           0.90       744
   macro avg       0.90      0.90      0.90       744
weighted avg       0.91      0.90      0.90       744
```



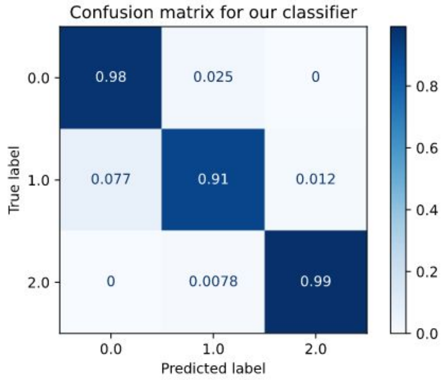

## 4. Random Forest

```
----------------------------------------- Random Forest --------------------------
Test Accuracy: 95.16129032258065 %
         TP     FP      TN      FN

0.0      244    10      474     16
1.0      222    24      487     11
2.0      242    2       491     9

Classification Report

              precision    recall  f1-score   support

         0.0       0.94      0.96      0.95       254
         1.0       0.95      0.90      0.93       246
         2.0       0.96      0.99      0.98       244

    accuracy                           0.95       744
   macro avg       0.95      0.95      0.95       744
weighted avg       0.95      0.95      0.95       744
```



# 5. Light GBM

```
------------------------------------- LightGBM ----------------------------
Test Accuracy: 95.96774193548387 %
         TP     FP      TN      FN

0.0      237    6       482     19
1.0      224    22      490     8
2.0      253    2       486     3

Classification Report

              precision    recall  f1-score   support

         0.0       0.93      0.98      0.95       243
         1.0       0.97      0.91      0.94       246
         2.0       0.99      0.99      0.99       255

    accuracy                           0.96       744
   macro avg       0.96      0.96      0.96       744
weighted avg       0.96      0.96      0.96       744
```



# Comparison and Remaining Work

Comparison Between KNN and SVM

| FeatureSelection\Classifier | SVM (%) | KNN (%) |
|---|---|---|
| | | |
| Pearson Correlation | 91.39258223 | 93.01075269 |
| Chi-Squared | 91.81245626 | 92.06989247 |
| Recursive Feature Elimination | 92.51224633 | 90.99462366 |
| Random Forest | 91.1126662 | 93.9516129 |
| LightGBM | 88.52344297 | 95.56451613 |

# Base Paper Comparison

**Base Paper Result : 89.38%**

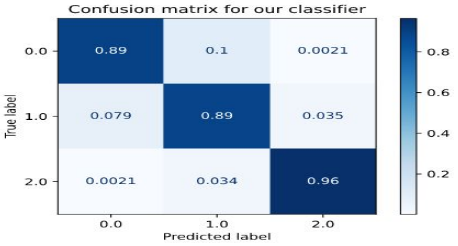
| Method | Validation | Focus | Accuracy |
|---|---|---|---|
| Inf. Gain Selection, CNN | 70/30 Split | Accuracy | 89.38% [86.94, 91.50] |
| OneR Selection, Random Forest | 10-fold | Accuracy | 87.2% [85.7, 88.6] |
| Evol. Selection, DEvoMLP | 5-fold | Accuracy, Resource Usage | 79.8% [78.1, 81.5] |

# Comparison

## Best Accuracy using SVM

## Our Result : 91.46%

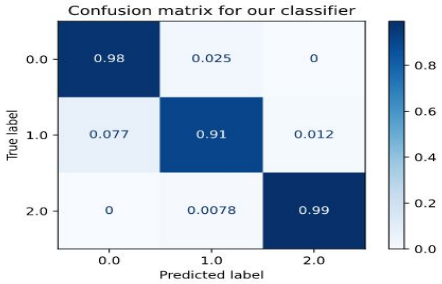
```
----------------------------------------- Chi-Squared -------------------------
(2479, 850)
2479
Training Accuracy: 100.0 %
Test Accuracy: 91.46256123163052 %
         TP      FP       TN      FN

0.0      421     50       919     39
1.0      428     55       881     65
2.0      458     17       936     18

Classification Report

              precision    recall  f1-score   support

         0.0       0.92      0.89      0.90       471
         1.0       0.87      0.89      0.88       483
         2.0       0.96      0.96      0.96       475

    accuracy                           0.91      1429
   macro avg       0.92      0.91      0.91      1429
weighted avg       0.91      0.91      0.91      1429
```




## Best Accuracy using KNN

## Our Result : 95.96%

```
--------------------------------------- LightGBM ------------------------------
Test Accuracy: 95.96774193548387 %
         TP      FP       TN      FN

0.0      237     6        482     19
1.0      224     22       490     8
2.0      253     2        486     3

Classification Report

              precision    recall  f1-score   support

         0.0       0.93      0.98      0.95       243
         1.0       0.97      0.91      0.94       246
         2.0       0.99      0.99      0.99       255

    accuracy                           0.96       744
   macro avg       0.96      0.96      0.96       744
weighted avg       0.96      0.96      0.96       744
```



# References

[1] Mental Emotional Sentiment Classification with an EEG-based Brain-machine Interface by Jordan J.Bird , Anikó Ekárt , Christopher D. Buckingham and Diego R. Faria

[2] DEAP: A Database for Emotion Analysis Using Physiological Signals Sander Koelstra, Student Member, IEEE, Christian Mühl, Mohammad Soleymani, Student Member, IEEE, Jong-Seok Lee, Member, IEEE, Ashkan Yazdani, Touradj Ebrahimi, Member, IEEE, Thierry Pun, Member, IEEE, Anton Nijholt, Member, IEEE, and Ioannis (Yiannis) Patras, Senior Member, IEEE

[3] A Deep Evolutionary Approach to Bioinspired Classifier Optimisation for Brain-Machine Interaction Jordan J. Bird , Diego R. Faria, Luis J. Manso, Anikó Ekárt, and Christopher D. Buckingham

[4] Classification of EEG Signals Based on Image Representation of Statistical Features Jodie Ashford * , Jordan J. Bird * , Felipe Campelo, and Diego R. Faria

[5] Amin HU, Mumtaz W, Subhani AR, Saad MNM and Malik AS (2017) Classification of EEG Signals Based on Pattern Recognition Approach. *Front. Comput. Neurosci*. 11:103. doi: 10.3389/fncom.2017.00103

[6]Amorim, P., Moraes, T., Fazanaro, D., Silva, J. and Pedrini, H., 2017. Electroencephalogram signal classification based on shearlet and contourlet transforms. Expert Systems with Applications, 67, pp.140-147.

[7] Luo Tian-jian, Fan Yachao, Chen Lifei, Guo Gongde, Zhou Changle. (2020). EEG Signal Reconstruction Using a Generative Adversarial Network With Wasserstein Distance and Temporal-Spatial-Frequency Loss. Frontiers in Neuroinformatics, doi: 10.3389/fninf.2020.00015

[8] S. Hwang, K. Hong, G. Son and H. Byun, "EZSL-GAN: EEG-based Zero-Shot Learning approach using a Generative Adversarial Network," 2019 7th International Winter Conference on Brain-Computer Interface (BCI), Gangwon, Korea (South), 2019, pp. 1-4, doi: 10.1109/IWW-BCI.2019.8737322.

[9] Classification of epileptic seizures in EEG signals based on phase space representation of intrinsic mode functions Author links open overlay panel Rajeev Sharma Ram Bilas Pachori

[10] W. Zheng, J. Zhu, Y. Peng and B. Lu, "EEG-based emotion classification using deep belief networks," 2014 IEEE International Conference on Multimedia and Expo (ICME), Chengdu, 2014, pp. 1-6, doi: 10.1109/ICME.2014.6890166.

[11]Classification Of Mental Tasks From Eeg Signals Using Extreme Learning Machine Nan-ying Liang, Paramasivan Saratchandran, Guang-bin Huang And Narasimhan Sundararajan

[12] K. Li, X. Li, Y. Zhang and A. Zhang, "Affective state recognition from EEG with deep belief networks," 2013 IEEE International Conference on Bioinformatics and Biomedicine, Shanghai, 2013, pp. 305-310, doi: 10.1109/BIBM.2013.6732507.

[13] Y. Ren and Y. Wu, "Convolutional deep belief networks for feature extraction of EEG signal," 2014 International Joint Conference on Neural Networks (IJCNN), Beijing, 2014, pp. 2850-2853, doi: 10.1109/IJCNN.2014.6889383.

[14] Hartmann, Kay Gregor et al. “EEG-GAN: Generative adversarial networks for electroencephalographic (EEG) brain signals.” *ArXiv* abs/1806.01875 (2018): n. pag.

[15] N. D. Truong, L. Kuhlmann, M. R. Bonyadi, D. Querlioz, L. Zhou and O. Kavehei, "Epileptic Seizure Forecasting With Generative Adversarial Networks," in IEEE Access, vol. 7, pp. 143999-144009, 2019, doi: 10.1109/ACCESS.2019.2944691.

https://github.com/nadzeri/Realtime-EEG-Based-Emotion-Recognition

https://github.com/Raghav714/EEG-Emotion-classification

https://github.com/Daisybiubiubiu/EEG-Emotion-Recognition

https://github.com/nasoboleva/EEG-Emotion-Recognition

https://github.com/jordan-bird/eeg-feature-generation

https://github.com/anumitgarg/Emotion-Classification-using-EEG

https://github.com/rishirdua/emotion-classification

https://github.com/Piyush-Bhardwaj/EEG-based-emotion-analysis-using-DEAP-dataset-for-Supervised-Machine-Learning

https://github.com/mahesh147/Support-Vector-Machine

https://towardsdatascience.com/andrew-ngs-machine-learning-course-in-python-support-vector-machines-435fc34b7bf9

https://medium.com/data-science-group-iitr/support-vector-machines-svm-unraveled-e0e7e3ccd49b

https://stackabuse.com/k-nearest-neighbors-algorithm-in-python-and-scikit-learn/

https://github.com/Piyush-Bhardwaj/EEG-based-emotion-analysis-using-DEAP-dataset-for-Supervised-Machine-Learning

https://github.com/robi56/emotion_classification

https://github.com/junmoan/eeg-feeling-emotions-LSTM

https://github.com/Mister5ive/emotion_classification

https://github.com/tongdaxu/EEG_Emotion_Classifier_DEAP

https://github.com/dweidai/DEAP-JRP-Emotion-Classification

https://github.com/yerzhan7orazayev/EEG-based-Emotion-Classification

https://github.com/PratikshyaM/Emotion-Recognition-From-EEG-brain-waves-