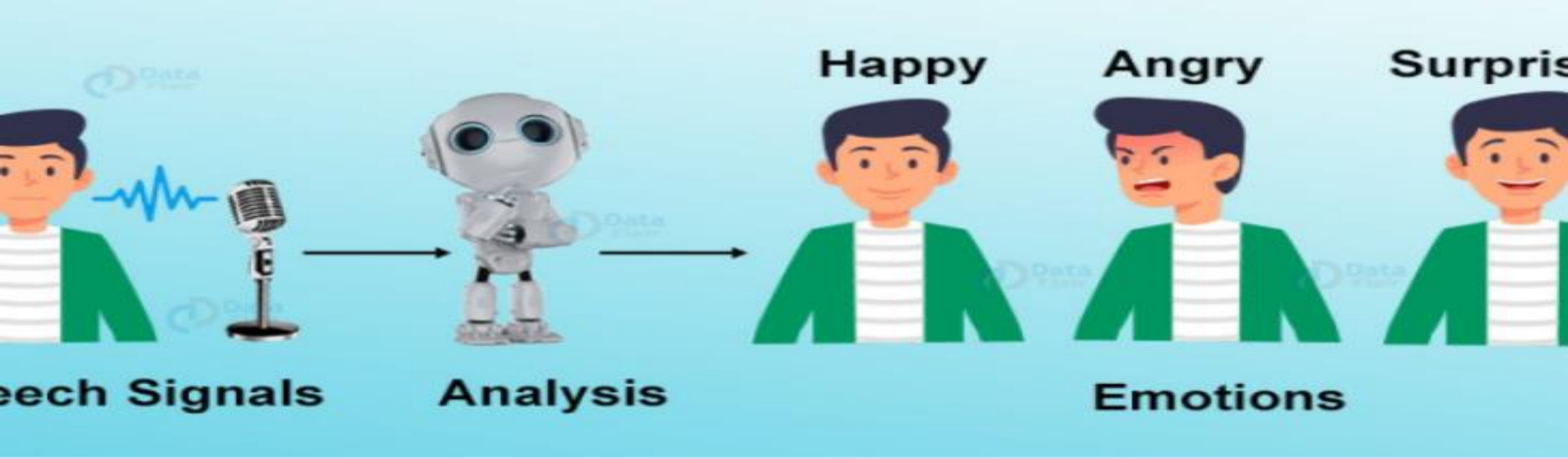
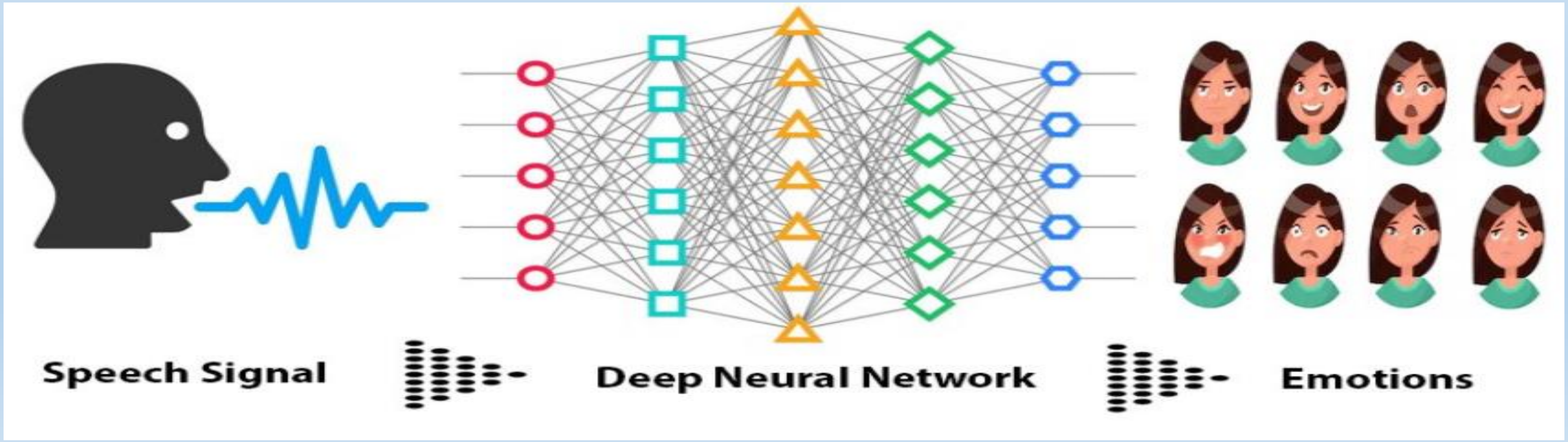


Speech Emotion Recognition
Machine Learning for Signal Processing
Term Project

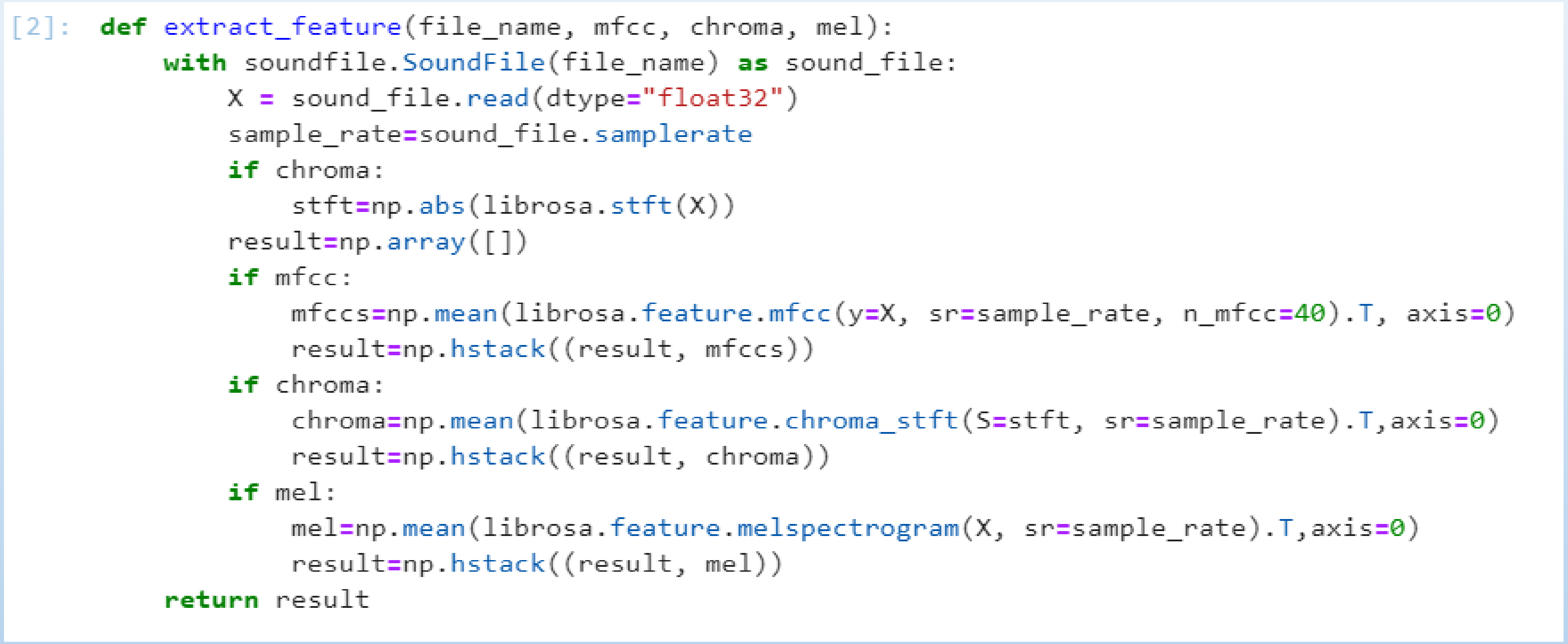
Osman Kaan Kurtça – 040160090
Aziz Gündoğdu - 040160112

What we have done:

- İmporting Human Speech Dataset including different emotions (RAVDESS)
- Feature Extraction (MFCC, Chroma, MEL)
- Data Preprocessing (Normalization, Dimensionality Reduction (PCA), One-Hot and Label Encoding)
- Determining the ML&DL models to be used. Training the dataset with models.
- Evaluation of Test results and Determination of the best model.



Feature Extraction



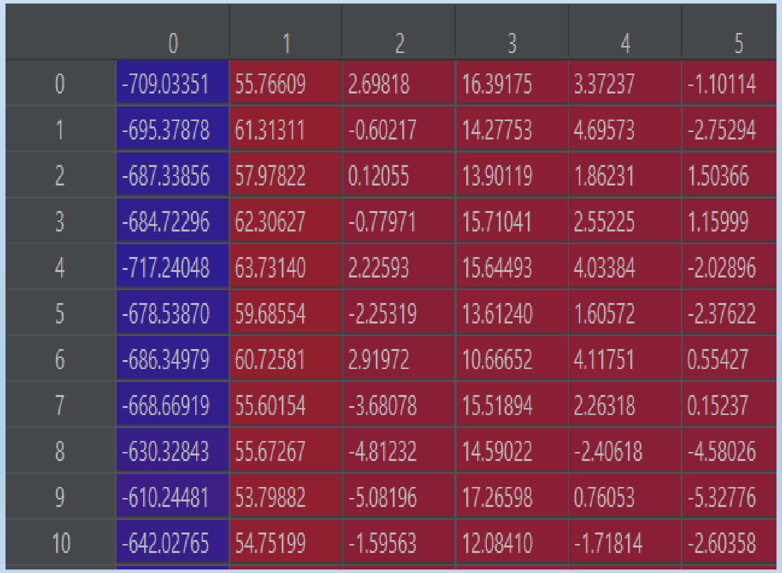
```
[2]:  def extract_feature(file_name, mfcc, chroma, mel):
          with soundfile.SoundFile(file_name) as sound_file:
              X = sound_file.read(dtype="float32")
              sample_rate=sound_file.samplerate
              if chroma:
                  stft=np.abs(librosa.stft(X))
              result=np.array([])
              if mfcc:
                  mfccs=np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=40).T, axis=0)
                  result=np.hstack((result, mfccs))
              if chroma:
                  chroma=np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T,axis=0)
                  result=np.hstack((result, chroma))
              if mel:
                  mel=np.mean(librosa.feature.melspectrogram(X, sr=sample_rate).T,axis=0)
                  result=np.hstack((result, mel))
          return result
```

We have handled the RAVDESS dataset and converted the audio files into a matrix by extracting features of audio files. While each row of the matrix represents a different human speech, each column except the last one represents a feature. The last column represents the class of the data.

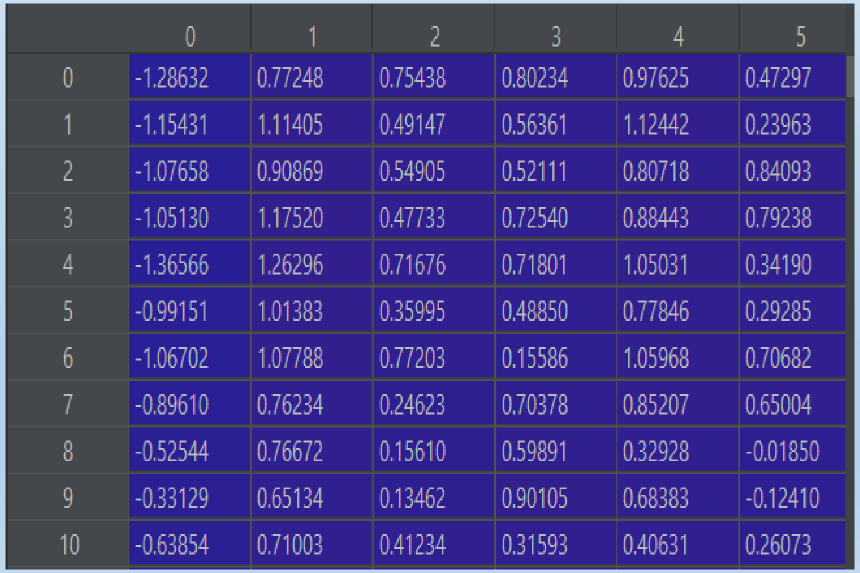
Data Preprocessing

Standard Scaler: The Standard Scaler assumes your data is normally distributed within each feature and will scale them such that the distribution is now centered around 0, with a standard deviation of 1. The mean and standard deviation are calculated for the feature and then the feature is scaled based on:

$$\frac{x_i - mean(x)}{stdev(x)}$$



	0	1	2	3	4	5
0	-709.03351	55.76609	2.69818	16.39175	3.37237	-1.10114
1	-695.37878	61.31311	-0.60217	14.27753	4.69573	-2.75294
2	-687.33856	57.97822	0.12055	13.90119	1.86231	1.50366
3	-684.72296	62.30627	-0.77971	15.71041	2.55225	1.15999
4	-717.24048	63.73140	2.22593	15.64493	4.03384	-2.02896
5	-678.53870	59.68554	-2.25319	13.61240	1.60572	-2.37622
6	-686.34979	60.72581	2.91972	10.66652	4.11751	0.55427
7	-668.66919	55.60154	-3.68078	15.51894	2.26318	0.15237
8	-630.32843	55.67267	-4.81232	14.59022	-2.40618	-4.58026
9	-610.24481	53.79882	-5.08196	17.26598	0.76053	-5.32776
10	-642.02765	54.75199	-1.59563	12.08410	-1.71814	-2.60358



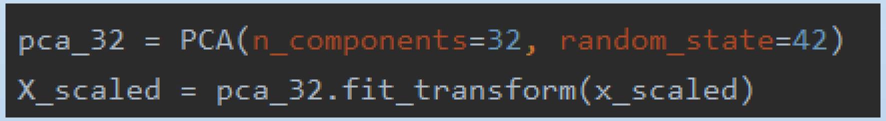
	0	1	2	3	4	5
0	-1.28632	0.77248	0.75438	0.80234	0.97625	0.47297
1	-1.15431	1.11405	0.49147	0.56361	1.12442	0.23963
2	-1.07658	0.90869	0.54905	0.52111	0.80718	0.84093
3	-1.05130	1.17520	0.47733	0.72540	0.88443	0.79238
4	-1.36566	1.26296	0.71676	0.71801	1.05031	0.34190
5	-0.99151	1.01383	0.35995	0.48850	0.77846	0.29285
6	-1.06702	1.07788	0.77203	0.15586	1.05968	0.70682
7	-0.89610	0.76234	0.24623	0.70378	0.85207	0.65004
8	-0.52544	0.76672	0.15610	0.59891	0.32928	-0.01850
9	-0.33129	0.65134	0.13462	0.90105	0.68383	-0.12410
10	-0.63854	0.71003	0.41234	0.31593	0.40631	0.26073

First 5 features of dataset before and after Normalization

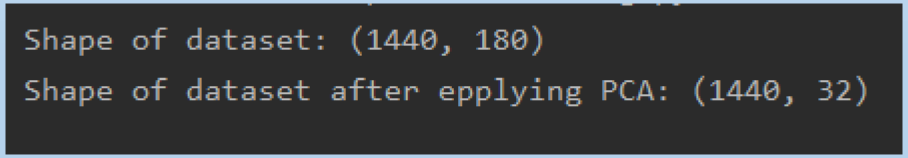
Dimensionality Reduction (PCA), One-Hot and Label Encoding

We applied principal component analysis (PCA) to reduce the dimension of our dataset.

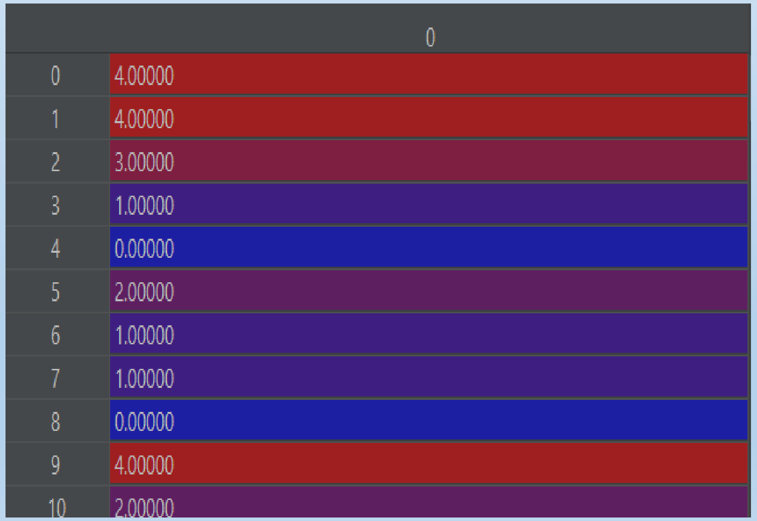
Since our labels types are string, we should encode these as a numerical value. We will use one-hot or label encoding according to our model.



```
pca_32 = PCA(n_components=32, random_state=42)
X_scaled = pca_32.fit_transform(x_scaled)
```

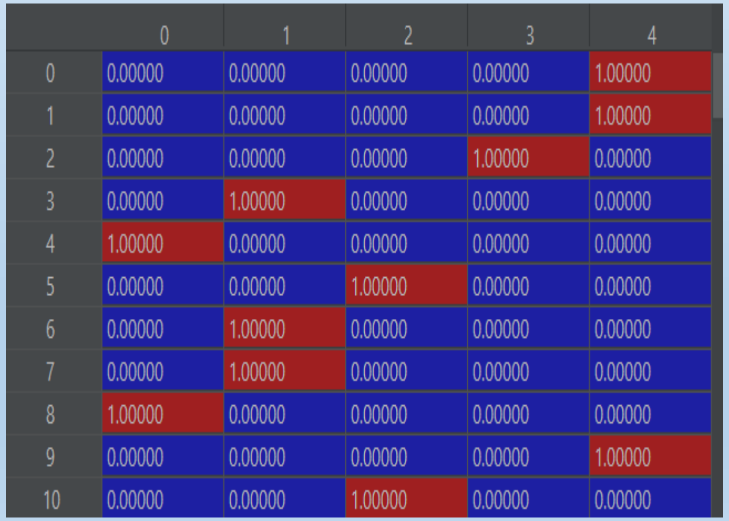


```
Shape of dataset: (1440, 180)
Shape of dataset after epplying PCA: (1440, 32)
```



	0
0	4.00000
1	4.00000
2	3.00000
3	1.00000
4	0.00000
5	2.00000
6	1.00000
7	1.00000
8	0.00000
9	4.00000
10	2.00000

Label Encoding

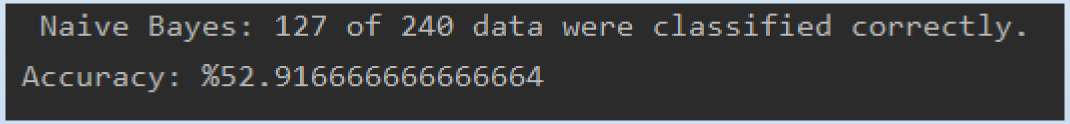


	0	1	2	3	4
0	0.00000	0.00000	0.00000	0.00000	1.00000
1	0.00000	0.00000	0.00000	0.00000	1.00000
2	0.00000	0.00000	0.00000	1.00000	0.00000
3	0.00000	1.00000	0.00000	0.00000	0.00000
4	1.00000	0.00000	0.00000	0.00000	0.00000
5	0.00000	0.00000	1.00000	0.00000	0.00000
6	0.00000	1.00000	0.00000	0.00000	0.00000
7	0.00000	1.00000	0.00000	0.00000	0.00000
8	1.00000	0.00000	0.00000	0.00000	0.00000
9	0.00000	0.00000	0.00000	0.00000	1.00000
10	0.00000	0.00000	1.00000	0.00000	0.00000

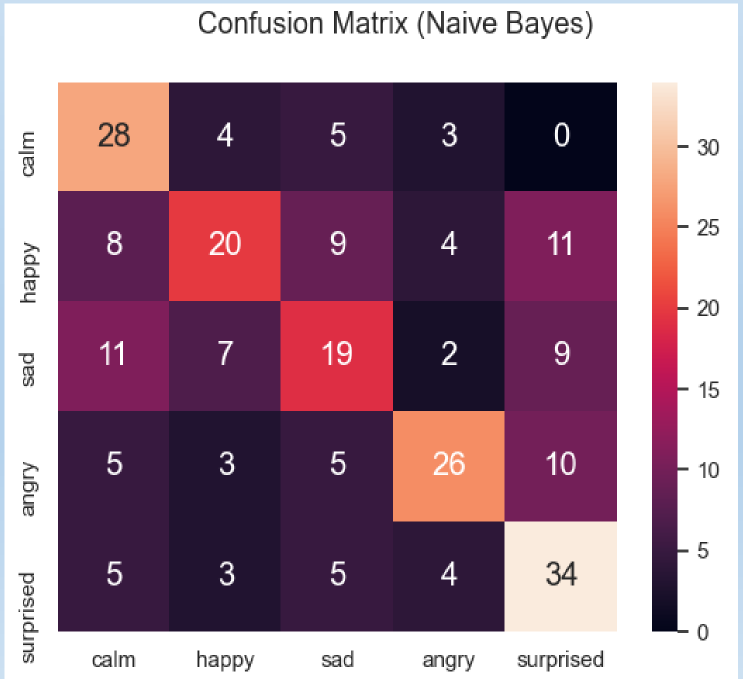
One-Hot Encoding

Naive Bayes Classifier

- Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.



```
 Naive Bayes: 127 of 240 data were classified correctly.
Accuracy: %52.916666666666664
```



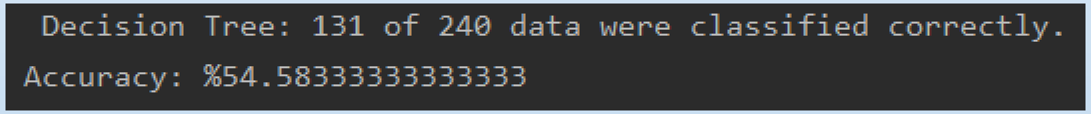
	calm	happy	sad	angry	surprised
calm	28	4	5	3	0
happy	8	20	9	4	11
sad	11	7	19	2	9
angry	5	3	5	26	10
surprised	5	3	5	4	34

We got an accuracy rate of just over %50, and we observed that some emotions were likened to each other ('calm' vs. 'sad') and wrong predictions were made. It's hard to say it's a good estimate rate.

In Confusion Matrix, while column names refer to the actual classes, row names refer to the predicted classes.

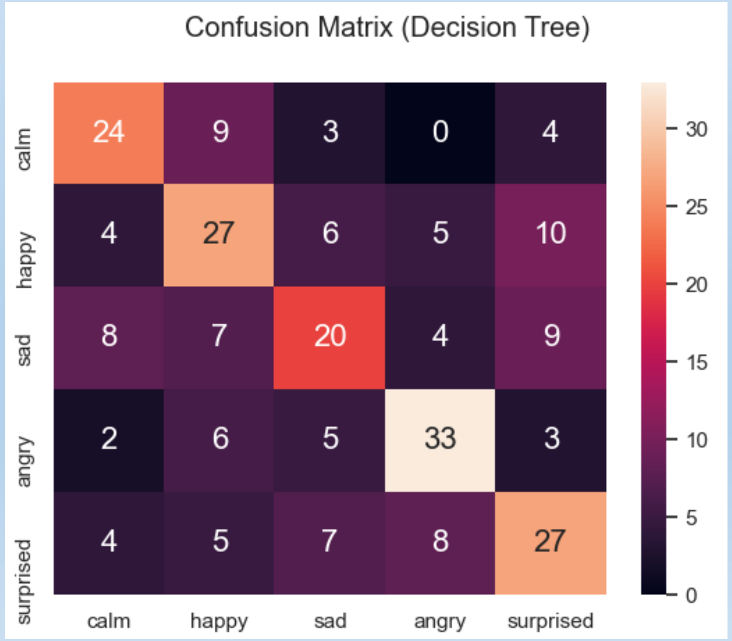
Decision Tree Classifier

- Decision tree learning is one of the predictive modelling approaches used in statistics, data mining and machine learning. It uses a decision tree (as a predictive model) to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.



```
 Decision Tree: 131 of 240 data were classified correctly.
Accuracy: %54.58333333333333
```

Confusion Matrix (Decision Tree)

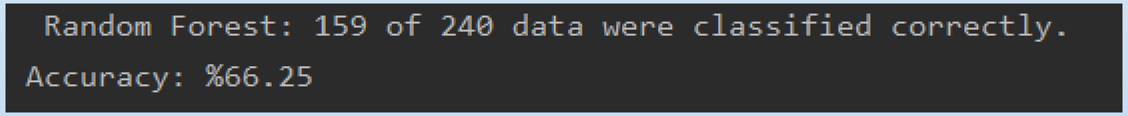


	calm	happy	sad	angry	surprised
calm	24	9	3	0	4
happy	4	27	6	5	10
sad	8	7	20	4	9
angry	2	6	5	33	3
surprised	4	5	7	8	27

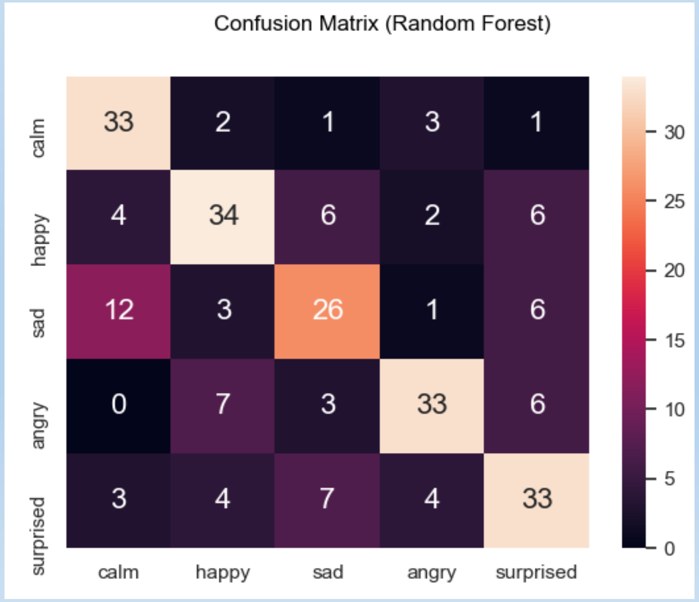
We see that our accuracy rate has increased and there are more accurate predictions in the error matrix than the naïve bayes classifier but our accuracy rate is still just over %50.

Random Forest Classifier

- Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees.



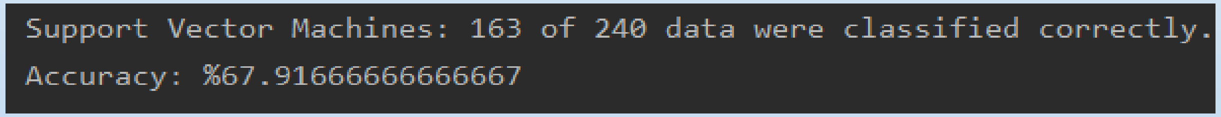
```
 Random Forest: 159 of 240 data were classified correctly.
Accuracy: %66.25
```



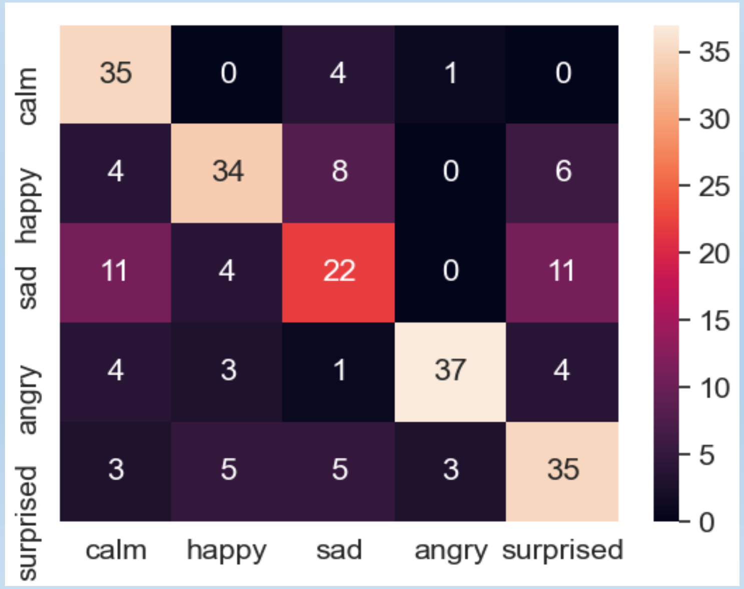
We see that we have significantly increased our accuracy rate but we can say that the model still does not predict ‘sad’ and ‘calm’ emotions well.

Support Vector Machines

- The objective of the support vector machine algorithm is to find a hyperplane in an N-dimensional space (N — the number of features) that distinctly classifies the data points. To separate the two classes of data points, there are many possible hyperplanes that could be chosen. Our objective is to find a plane that has the maximum margin. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.



```
Support Vector Machines: 163 of 240 data were classified correctly.
Accuracy: %67.91666666666667
```

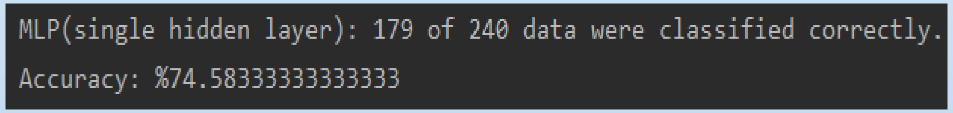


	calm	happy	sad	angry	surprised
calm	35	0	4	1	0
happy	4	34	8	0	6
sad	11	4	22	0	11
angry	4	3	1	37	4
surprised	3	5	5	3	35

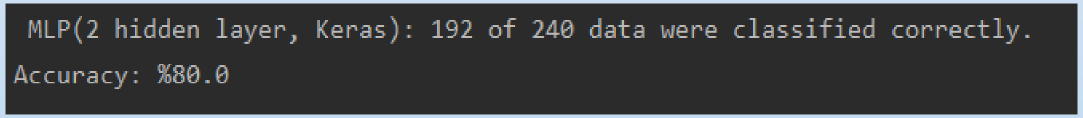
We got a slightly better result than the Random Forest Classifier but when we examine the error matrix, we observe the same problem here. Now, we will use deep learning model for better results.

Multilayer Perceptron

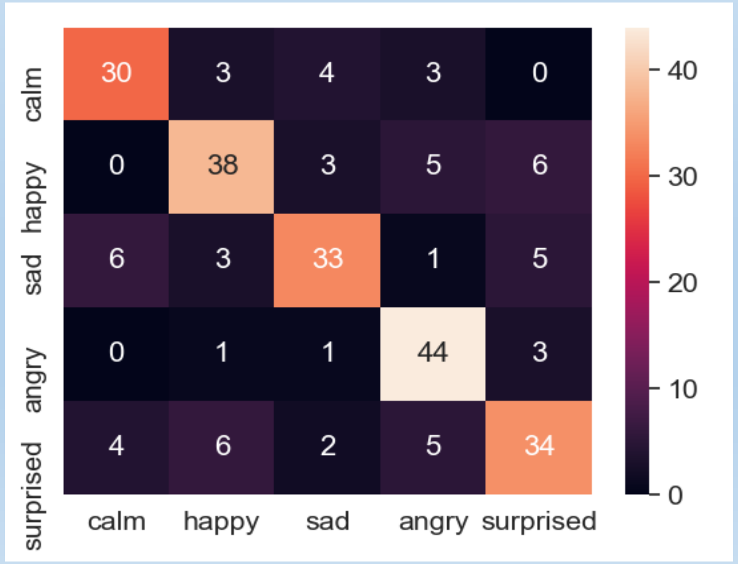
- A multilayer perceptron (MLP) is a class of feedforward artificial neural network (ANN). An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

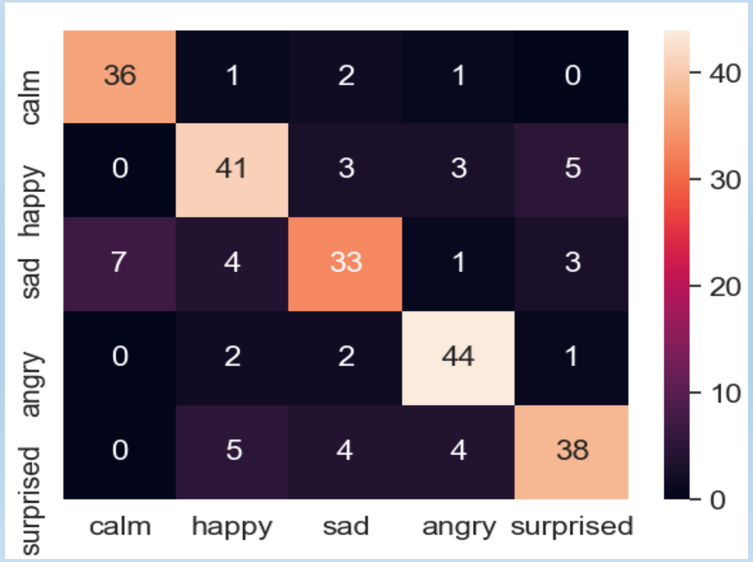


```
MLP(single hidden layer): 179 of 240 data were classified correctly.
Accuracy: %74.58333333333333
```



```
 MLP(2 hidden layer, Keras): 192 of 240 data were classified correctly.
Accuracy: %80.0
```

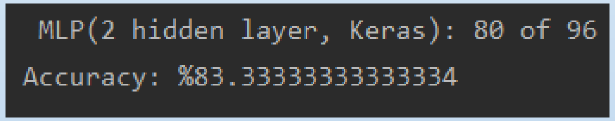




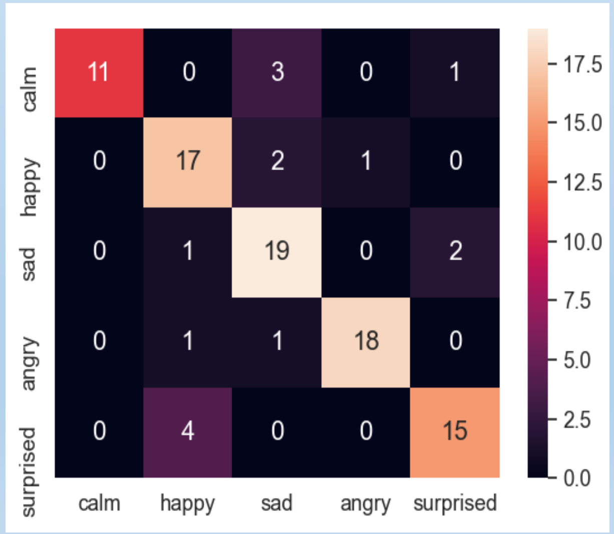
Conclusion

- Our final accuracy score is 0.80. This is not a perfect accuracy rate, but we can say the this is best result compared to other models we use. We are able to predict 4 out of every 5 test data correctly.
- As a result, we obtained an accuracy rate better than the accuracy of the study taken as the source (0.71) and we classified for more emotion. 5 emotions were used as label instead of 4.

Bonus

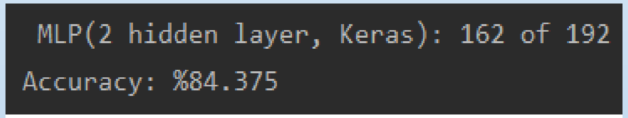


```
MLP(2 hidden layer, Keras): 80 of 96
Accuracy: %83.33333333333334
```

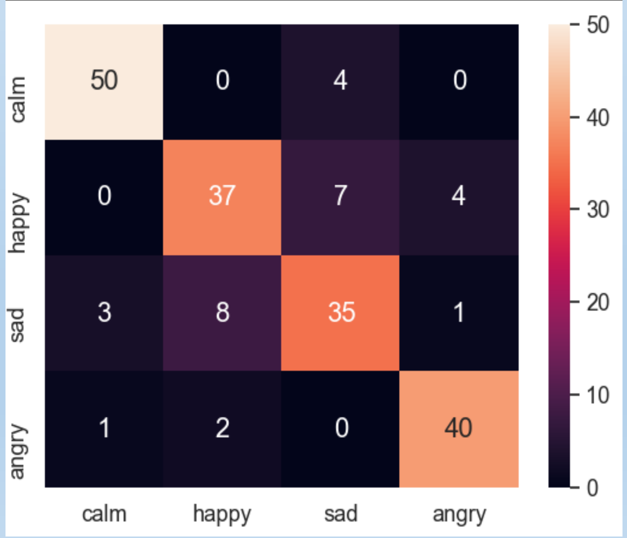


	calm	happy	sad	angry	surprised
calm	11	0	3	0	1
happy	0	17	2	1	0
sad	0	1	19	0	2
angry	0	1	1	18	0
surprised	0	4	0	0	15

Confusion matrix for 0.1 Test set rate

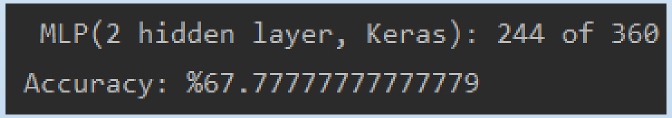


```
MLP(2 hidden layer, Keras): 162 of 192
Accuracy: %84.375
```

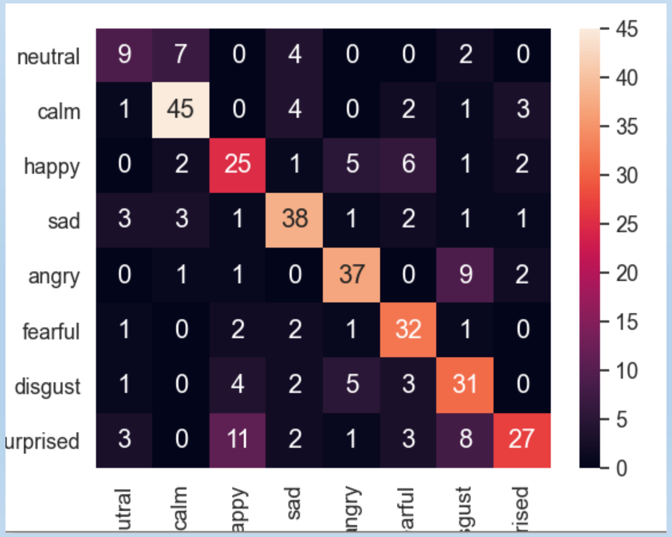


	calm	happy	sad	angry
calm	50	0	4	0
happy	0	37	7	4
sad	3	8	35	1
angry	1	2	0	40

Confusion matrix for 4 emotions



```
MLP(2 hidden layer, Keras): 244 of 360
Accuracy: %67.77777777777779
```



	utral	calm	appy	sad	ngry	arful	gust	rised
neutral	9	7	0	4	0	0	2	0
calm	1	45	0	4	0	2	1	3
happy	0	2	25	1	5	6	1	2
sad	3	3	1	38	1	2	1	1
angry	0	1	1	0	37	0	9	2
fearful	1	0	2	2	1	32	1	0
disgust	1	0	4	2	5	3	31	0
urprised	3	0	11	2	1	3	8	27

Confusion matrix for all emotions (8)

References

- https://data-flair.training/blogs/python-mini-project-speech-emotion-recognition/
- https://www.geeksforgeeks.org/naive-bayes-classifiers/
- https://en.wikipedia.org/wiki/Decision_tree
- https://towardsdatascience.com/understanding-random-forest-58381e0602d2
- https://en.wikipedia.org/wiki/Training,_validation,_and_test_sets
- https://en.wikipedia.org/wiki/Multilayer_perceptron
- https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
- https://www.wikiwand.com/en/Support-vector_machine