



# Spoken Language Identification

Group 30

- Himanshu Kumar (2018147)
- Jaspreet Saka (2018237)
- Nishant Mishra (2018165)

# Problem Statement



## Problem Statement

We will use the audio of human speech to recognize the language they are speaking in it. In this project, we will be classifying three languages English, German and Spanish.

## Motivation

The phenomenon of globalization has brought together people from around the world. However one barrier to this increase in global communication is that many people speak different languages and effectively lack a common communication medium. Spoken Language Identification System is a first step toward providing such common medium.

# Dataset & Evaluation Metrics Used



## DataSet

- Dataset for this project is from Kaggle which contains 73080 Training and 540 Testing speech samples of 10 sec each.
- Now since the Dataset is very Large so we have used a part of it approx (1/3)rd of It, But we have ensured one thing that the presentation of all languages, genders, speakers and transformations (like speed, noise and pitch) is well maintained as in original dataset.

**Number of Training Samples=26618, Number of Test Samples=540, Number of Validation Samples= 5308**

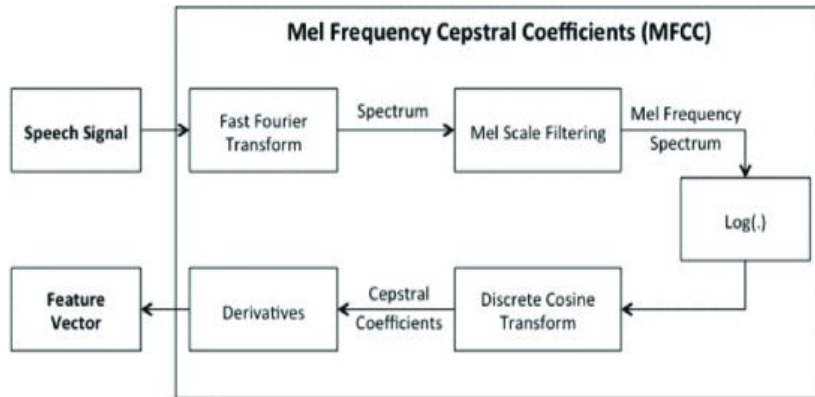
## Evaluation Metrics

We have Used Following Evaluation Metrics

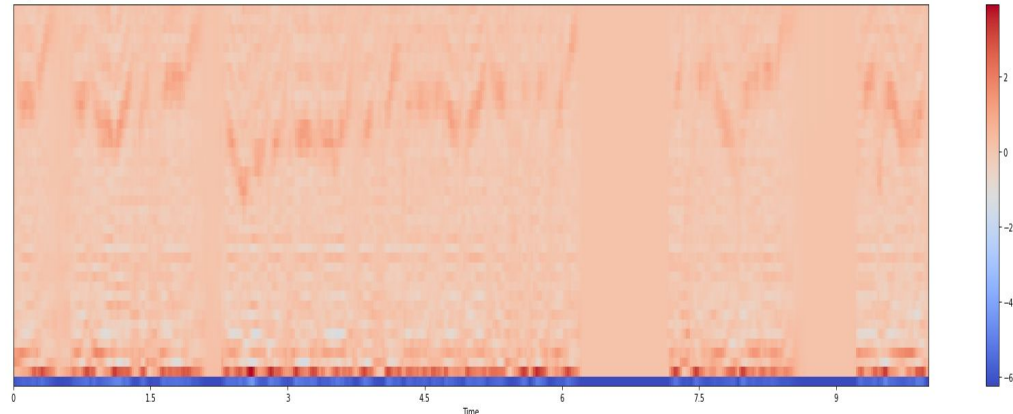
1. Accuracy
2. F1 Score
3. Confusion Matrix
4. ROC Curve

# Data Preprocessing

- In this Project, we have used MFCC (Mel Frequency Cepstral Coefficients) method for the Feature Extraction.
- MFCC mimics parts of speech perception and production to extract features having details about the linguistic message spoken by the speaker. In the end of process the MFCC method provide us a feature matrix of audio signal.



Steps to Find MFCC matrix



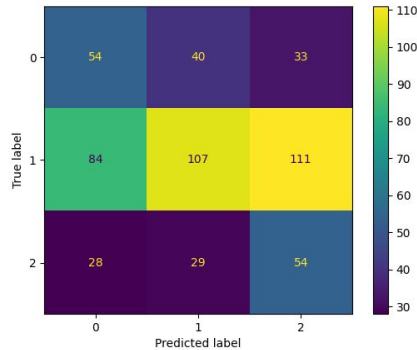
MFCC Matrix plot

# Approaches Used

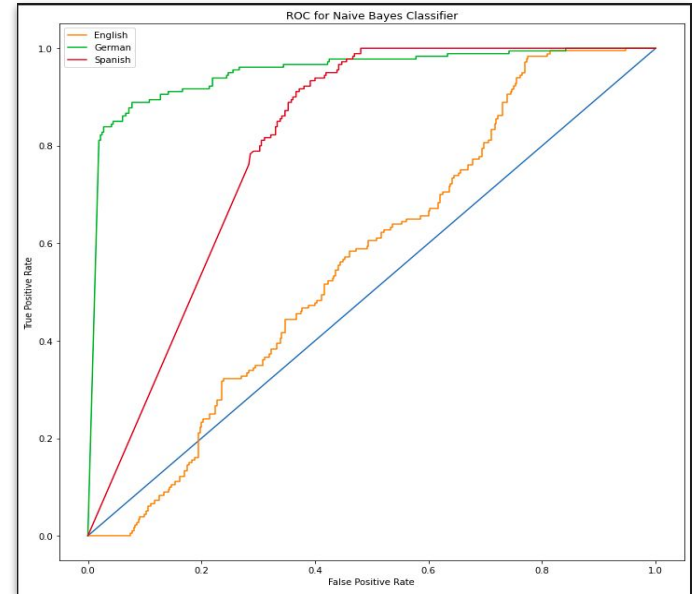
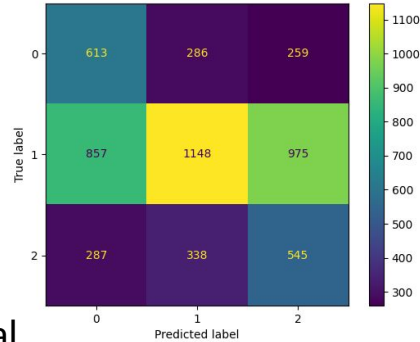
- **Baseline Model - Gaussian Naive Bayes**

- Our Baseline model was Gaussian Naive Bayes. Following this we have done our analysis for this model with following evaluation metrics
- Train Accuracy = 53.56 %
- Testing Accuracy = 46.3%
- We have plotted the ROC curve for each class and noticed that the model was working pretty good for German but performing poorly on English class.

Test



Val



# Approaches Used (cont.)



- **Three Advanced Models**

- **Convolution Neural Network**

- CNN models are generally used on images for classification, in our case the MFCC matrix is also come out to 2D matrix of  $40 \times 431$ .
- CNN can do prediction. Analyzing the speech data, CNN can not only learn from images but can also learn from speeches. CNN can do analyze the data, learn from this data and able to identify words, utterances.

- **Support Vector Machine**

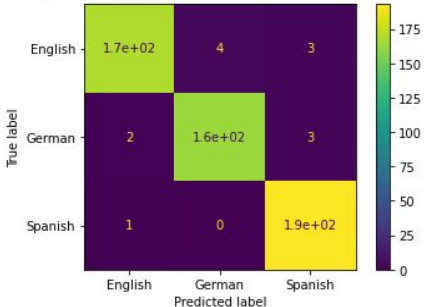
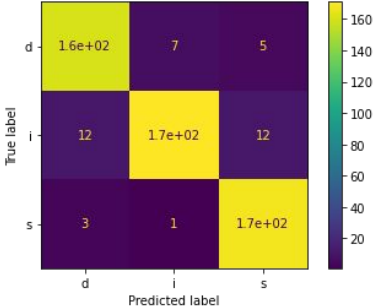
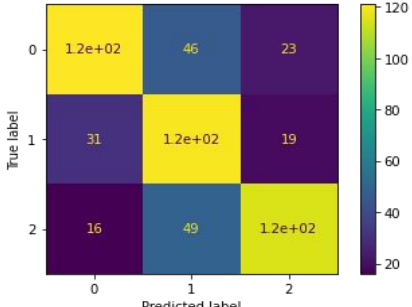
- We have used SVM because it provides us a set of Kernels by tuning which we can set a perfect hyper-plane and hence able to classify data with very good accuracy.
- As SVM does not provide any `partial_fit` method during training and our dataset is large so I have used Feature reduction methods to reduce the number of feature which help in training model efficiently.

- **Decision Tree**

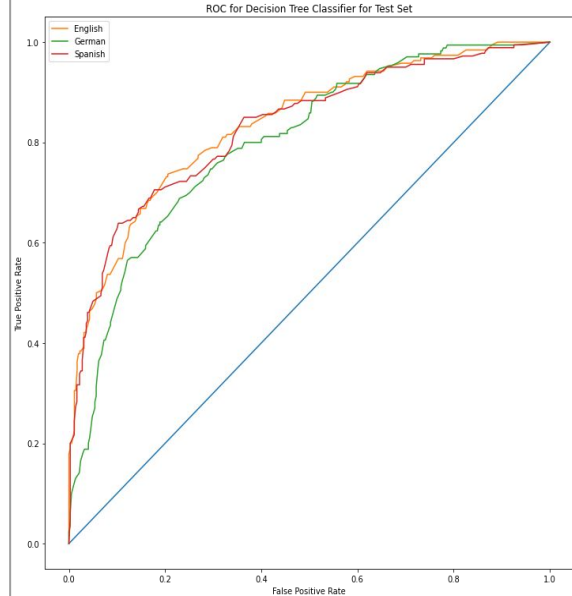
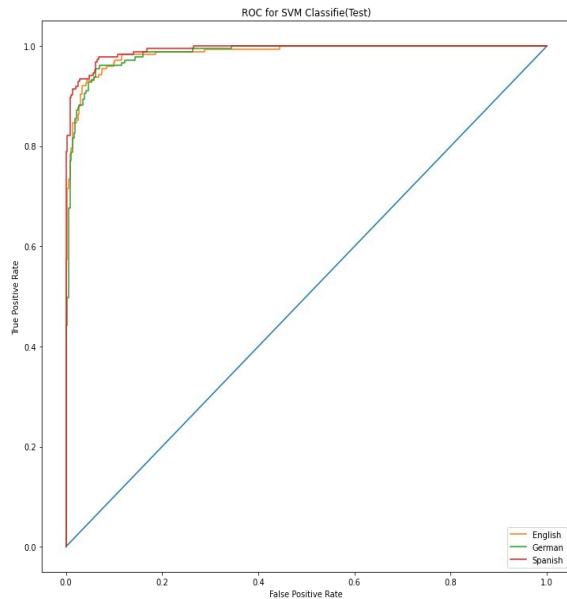
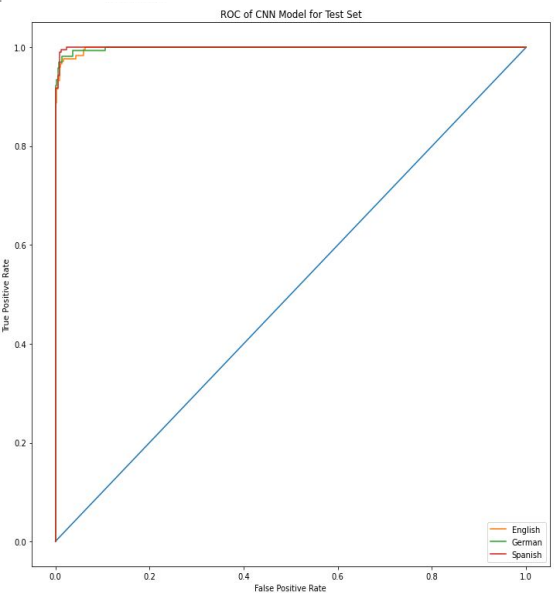
- Our problem is a multiclass classification and decision tree can easily solve a multiclass classification without high computation cost.

# Main Results

Results on Test Set for every learning technique

Metrics	CNN Model	SVM model	Decision Tree																																																
Accuracy	97.59 % (Test Set) 98.64 % (Validation Set)	92.6% (Test Set) 92.05% (Validation Set)	65.92 % (Test Set) 63.01 % (Validation Set)																																																
F1 Score	English - 0.9714285 German - 0.9732937 Spanish - 0.9821883	English - 0.9226361 German - 0.9144385 Spanish - 0.94117647	English - 0.67597765 German - 0.62337662 Spanish - 0.68249258																																																
Confusion Matrix	 <p>Confusion Matrix for CNN Model (True label vs Predicted label):</p> <table><thead><tr><th></th><th>English</th><th>German</th><th>Spanish</th></tr></thead><tbody><tr><th>English</th><td>1.7e+02</td><td>4</td><td>3</td></tr><tr><th>German</th><td>2</td><td>1.6e+02</td><td>3</td></tr><tr><th>Spanish</th><td>1</td><td>0</td><td>1.9e+02</td></tr></tbody></table>		English	German	Spanish	English	1.7e+02	4	3	German	2	1.6e+02	3	Spanish	1	0	1.9e+02	 <p>Confusion Matrix for SVM model (True label vs Predicted label):</p> <table><thead><tr><th></th><th>d</th><th>i</th><th>s</th></tr></thead><tbody><tr><th>d</th><td>1.6e+02</td><td>7</td><td>5</td></tr><tr><th>i</th><td>12</td><td>1.7e+02</td><td>12</td></tr><tr><th>s</th><td>3</td><td>1</td><td>1.7e+02</td></tr></tbody></table>		d	i	s	d	1.6e+02	7	5	i	12	1.7e+02	12	s	3	1	1.7e+02	 <p>Confusion Matrix for Decision Tree (True label vs Predicted label):</p> <table><thead><tr><th></th><th>0</th><th>1</th><th>2</th></tr></thead><tbody><tr><th>0</th><td>1.2e+02</td><td>46</td><td>23</td></tr><tr><th>1</th><td>31</td><td>1.2e+02</td><td>19</td></tr><tr><th>2</th><td>16</td><td>49</td><td>1.2e+02</td></tr></tbody></table>		0	1	2	0	1.2e+02	46	23	1	31	1.2e+02	19	2	16	49	1.2e+02
	English	German	Spanish																																																
English	1.7e+02	4	3																																																
German	2	1.6e+02	3																																																
Spanish	1	0	1.9e+02																																																
	d	i	s																																																
d	1.6e+02	7	5																																																
i	12	1.7e+02	12																																																
s	3	1	1.7e+02																																																
	0	1	2																																																
0	1.2e+02	46	23																																																
1	31	1.2e+02	19																																																
2	16	49	1.2e+02																																																

# Main Results





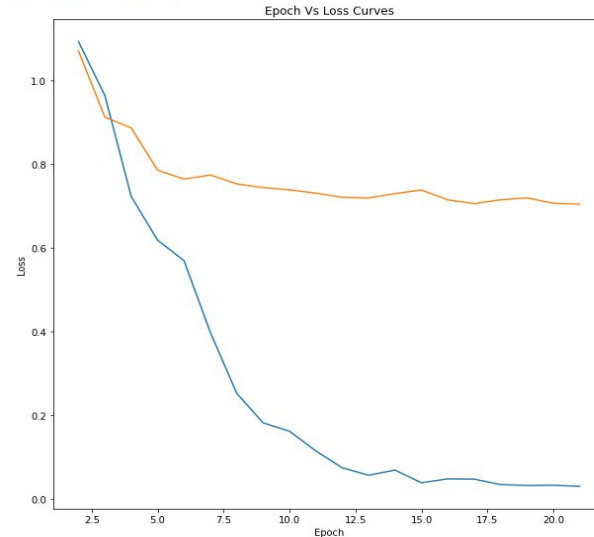
# Analysis

## CNN Model

- I have used the architecture for CNN model as shown in Fig 1.
- Learning rate = 0.0001 and Batch size = 64.
- Learning Curve for this architecture is shown in Fig 2.

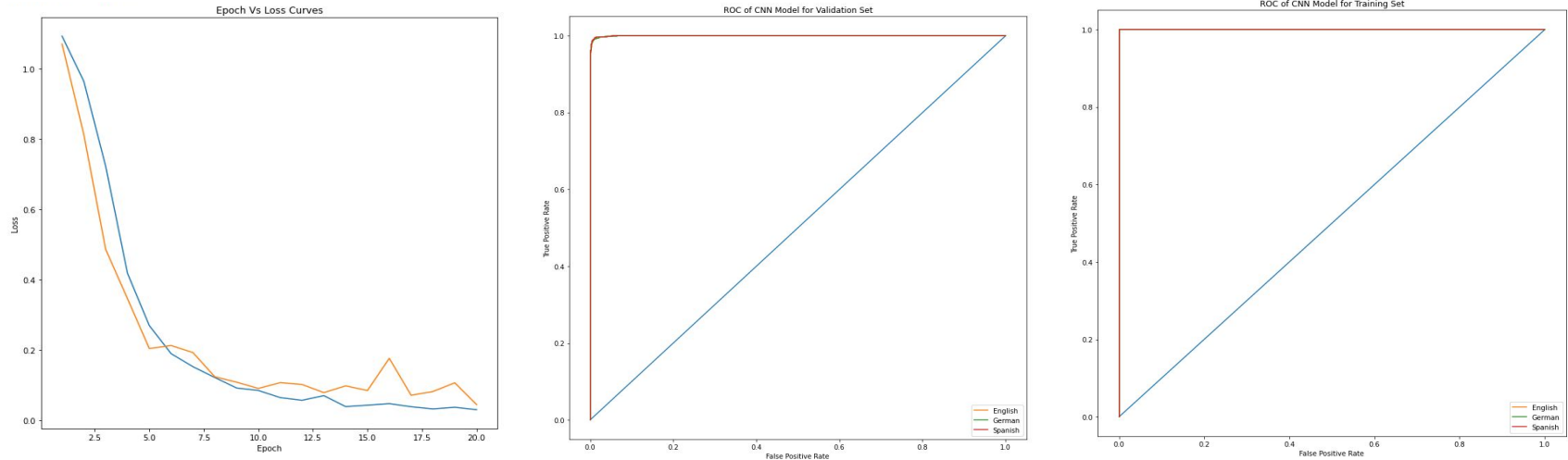
Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 16, 42, 433]	160
MaxPool2d-2	[-1, 16, 21, 216]	0
Conv2d-3	[-1, 32, 23, 218]	4,640
MaxPool2d-4	[-1, 32, 11, 108]	0
Conv2d-5	[-1, 64, 13, 110]	18,496
MaxPool2d-6	[-1, 64, 6, 54]	0
Conv2d-7	[-1, 128, 8, 56]	123,008
MaxPool2d-8	[-1, 128, 4, 27]	0
Conv2d-9	[-1, 256, 6, 29]	491,776
MaxPool2d-10	[-1, 256, 3, 14]	0
AvgPool2d-11	[-1, 256, 1, 4]	0
Linear-12	[-1, 64]	65,600
Linear-13	[-1, 3]	195

Total params: 703,875  
Trainable params: 703,875  
Non-trainable params: 0



# Analysis (cont.)

- As, we can take insights from learning curve which implies model was overfitting.
- The accuracy for the validation set was 73.65% and for training set 97.46%. This also shows that model is overfitting.
- Now, to regularize the model a dropout layer is added between layer 12 and layer 13 with drop probability of 0.5.
- Learning Curve after adding Dropout and increasing the Learning rate to 0.001 is shown in Fig 1.



# Analysis (Using SVM Model)

- SVM only support binary classification so I have used one versus rest approach to train the Model.
- I have tried different hyperparameters to tune my model and fit a hyper-plane which will able to classify our dataset with higher accuracy, figure 1 show the ROC curve which I got for linear Kernels.
- As observe from the ROC curve for 'Linear' kernel, the dataset we have used isn't linearly separable, and hence it is given very bad accuracy(38% which is even lower than our baseline model).
- As the Number of feature is very large (170000 approx) so I have used feature reduction to reduce the Number of feature, I have tried different number of feature to reduce to (figure 2) but got good results(Validation Accuracy) for 100 features.

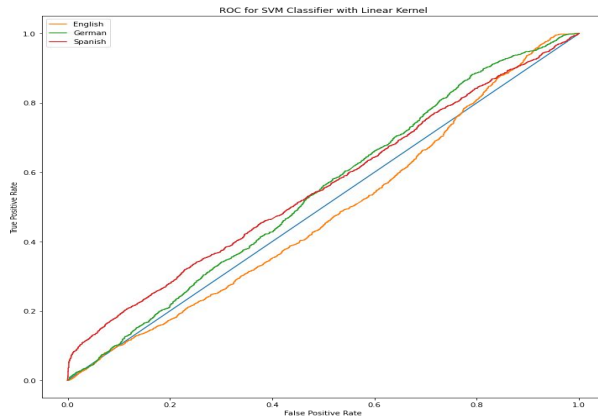
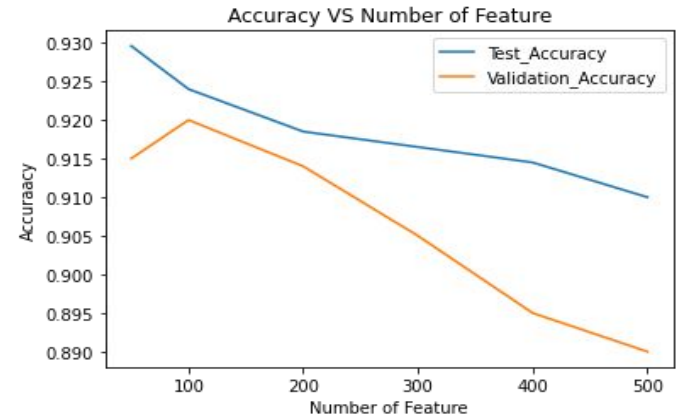


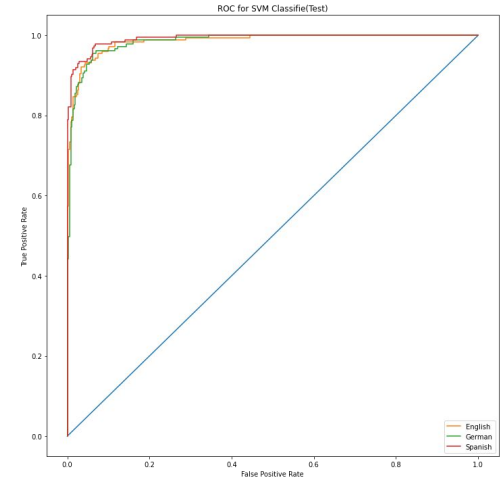
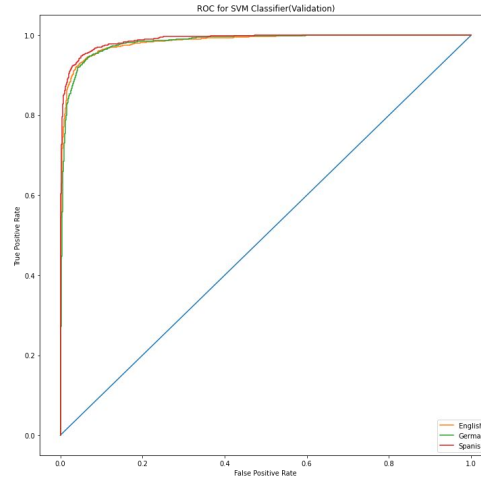
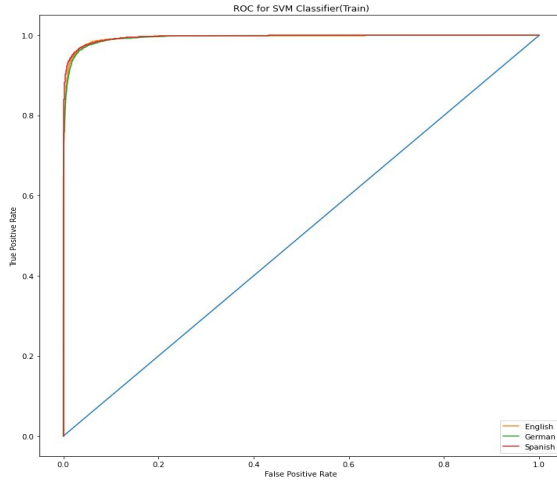
Figure1

Figure 2



# Analysis (cont.)

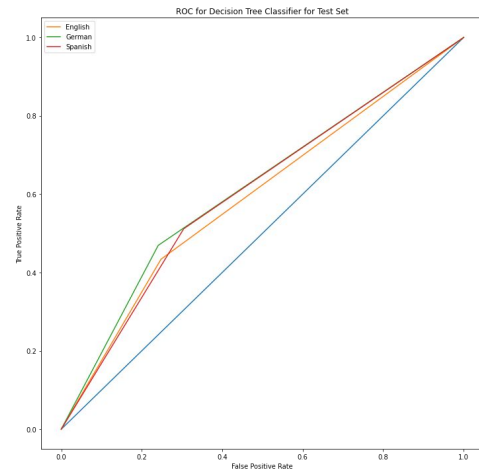
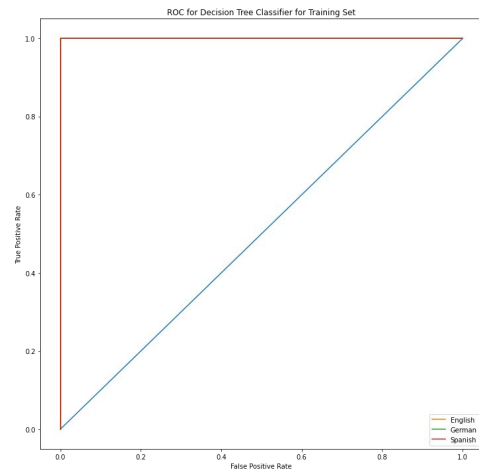
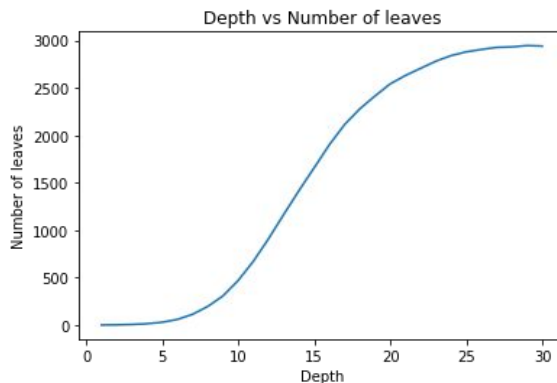
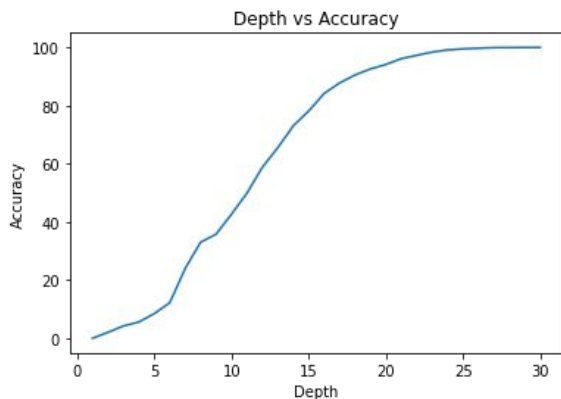
- I got good results when the kernel used for setting hyperplane is 'rbf'.
- The Accuracy which i got using 'rbf' kernel are 92.6% (on Test), 92.04% (on Validation) and 95.2% on Training Sample which clearly shows that the testing is done correctly and there is no case of Overfitting in the model training.
- When I have used 'polynomial' kernel with degree of 4 then i was also getting comparable results (Accuracy on test set 91.9%) to what i have got using 'rbf'.



# Analysis (cont.)

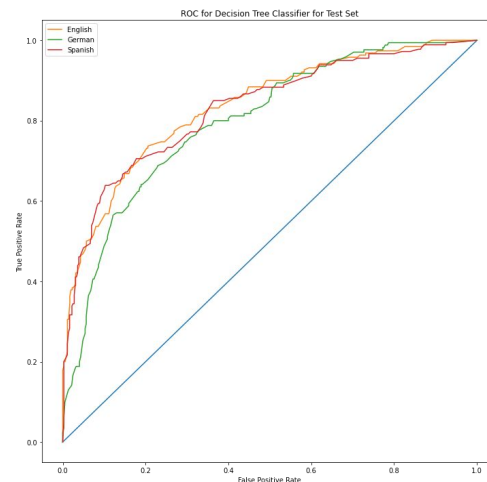
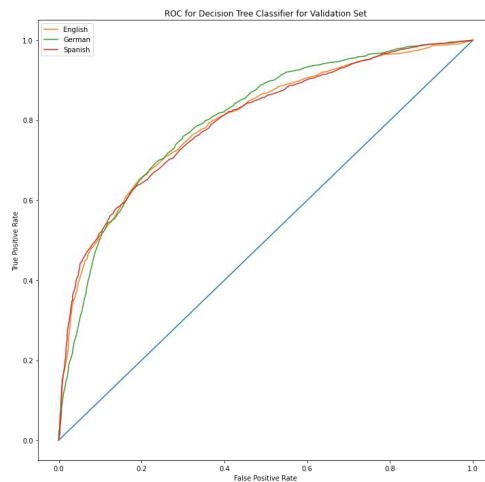
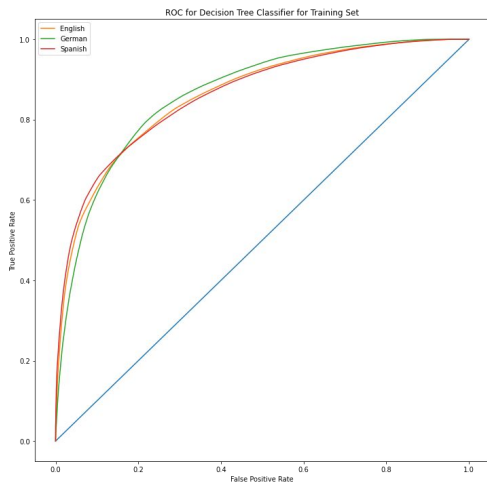
## Decision Tree Model

- As we run Model we get depth of Tree equals to 30 and number of leaf nodes equals to 2941.
- we get 100% accuracy in training set & 47% accuracy in Training set. It's clearly observed our model is Overfitting and cross checked with ROC graph below.
- To counter the Overfitting issue we have used the hyperparameters such as max\_depth, max\_leaf\_nodes and max\_features to tune the model.
- Max\_depth is most flexible and also have small value. I create models with max\_depth 0 to 30 and observed the accuracy of model and max\_leaf\_nodes and plot graph as shown below.



# Analysis (cont.)

- We observed after accuracy start to saturate around depth equals to 17. Then we selected `max_depth=17` and `max_leaf_nodes=251`
- Large numbers of features are also a reason to get overfitted model. So we also reduced the max features to 100 using `ipca`.
- Our tuned model gives 40.15% better accuracy than the previous model.
- Accuracy of Training: 70.6% , Accuracy of Validation: 63% & Accuracy of Test: 65.9%.



# Conclusion



- The CNN Model has performed best on the given dataset followed by SVM.
- The decision tree classifier overfits on the most of the datasets.

## **What we learned:**

- We learned that CNN model can also work on data other than images like audio data.
- SVM Model is not overfits Easily as it have gamma as a parameter which manage the model learning in such a way that it not overfits.
- The Audio Signal data we have used is not linearly separable.