**Week2 Progress**

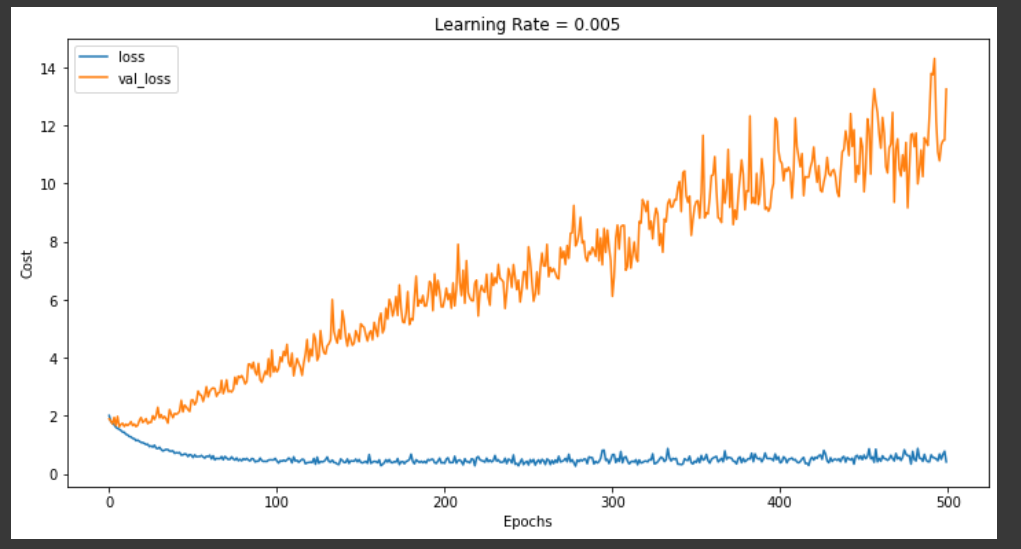
**Problem Statement:**

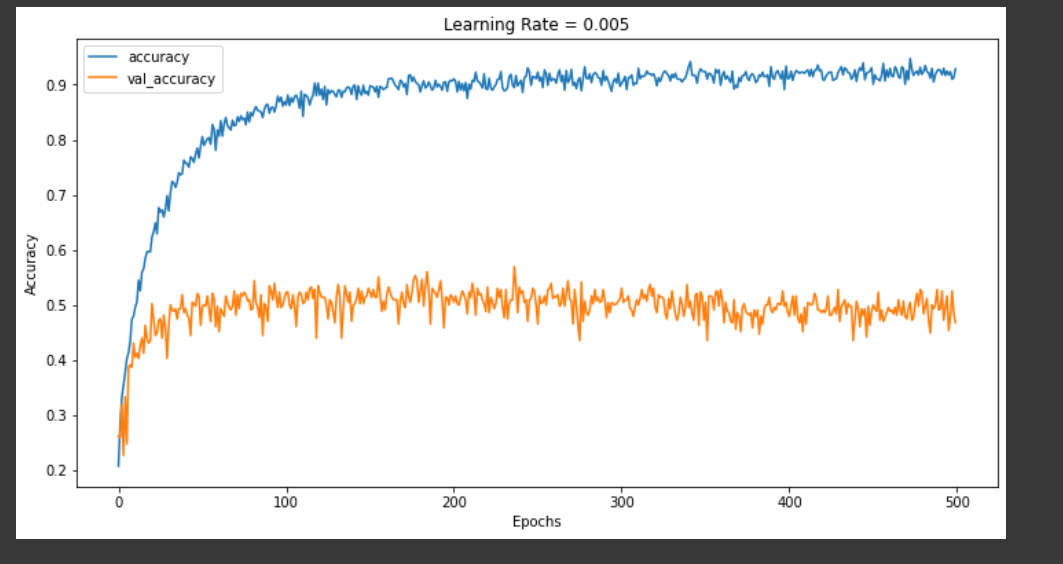
There are many accidents that happen every day and it can be also be caused by the state of mind of the driver. Evaluating the state of mind of the driver, the emotions that the driver exhibits can help manufacturers adapt vehicle control to  mitigate the accident, insurance companies to understand the state of mind of the driver at the time of the accident to make better decisions.

**CNN:**

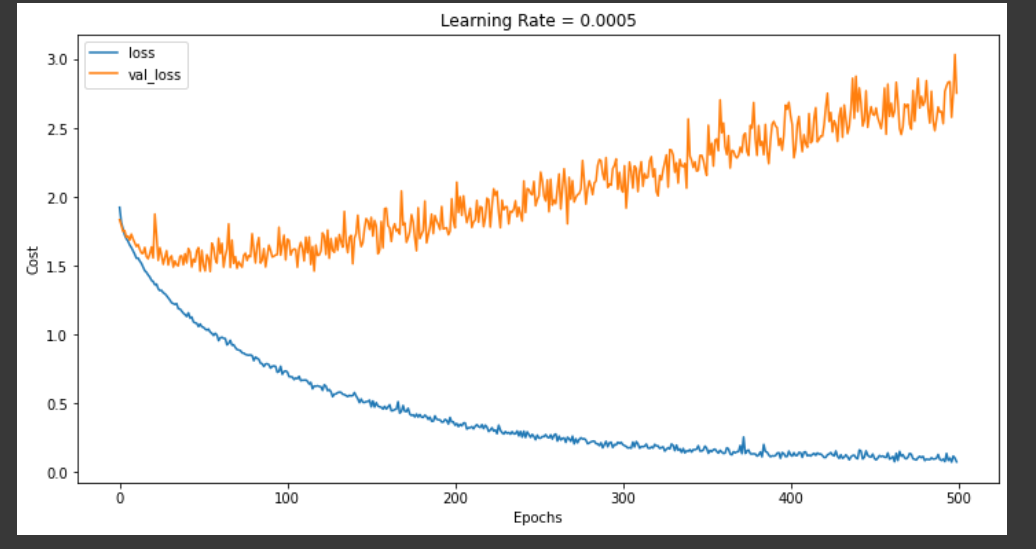
* Extracted the mfcc and mel spectrogram features from the dataset
* Split the dataset into dependent and independent variables and encoded the label using one hot encoding but this did not work (unimplemented graph error => cannot convert string to int64)
* Tried out LabelEncoding instead of one hot encoding which solved the issue
* Split the data into train and test set and added an extra dimension(3rd dimension ) as CNN usually works with 2 dimensions and a third which represents the number of components of the input
* Used keras from tensor flow to make the network architecture
* Initialised a sequential model as there are no intermediate inputs -> there is one input and one output tensor
* Added a Conv1D layer with a 128 filters with kernel\_size = 5 and then added a relu activation function after the conv1D layer to make the feature map output of the first convolution output non linear (Relu is max(0,value) so it removes all the negative values thus introducing non linearity which is necessary since all problems are not linear and producing linear results when data cannot be mapped using a linear model Is not optimal)
* Added a max pooling layer to reduce the size and complexity of the input tensor
* Added another conv1D layer and a relu activation layer
* Added a flatten() layer which flattens the input tensor into a 1d input and then a dense fully connected layer that helps to predict the emotion
* Initialize an optimizer which helps compile and run the model and trains the model towards the minima of the cost (initialize a loss function to help minimize the cost)
* experimented between categoricalcrossentropy and sparsecategoricalcrossentropy (loss functions) and had better accuracy with sparsecategoricalcrossentropy .
* Metric(usually used to judge the progress of the model) which was used to train the model is the accuracy
* Had to come up with a way to reduce and overcome overfitting which is a common issue in any machine learning model=>Used dropout layers to help with this
* Figuring out the right learning rate of the model using trial and error:

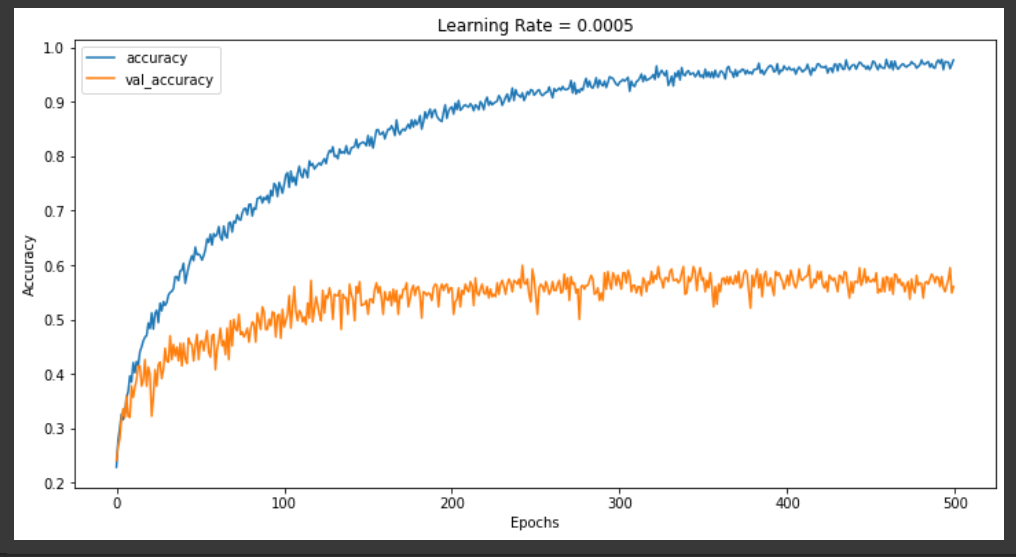
Lr=0.005 500 epochs (not optimal as the loss seems to be increasing with a decrease in training loss and accuracy seems to be consistent with increase in the training accuracy which indicates overfitting)

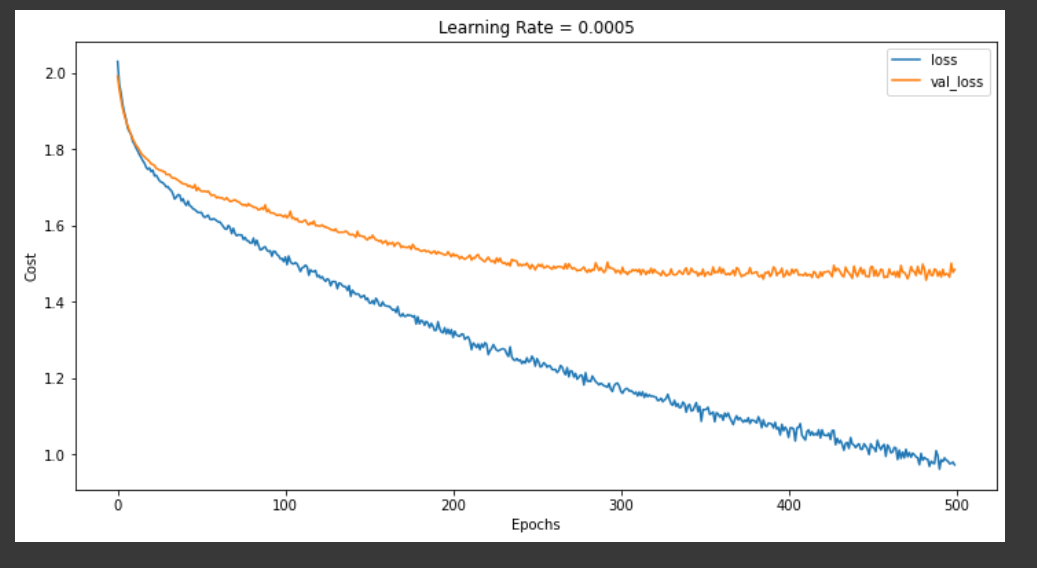


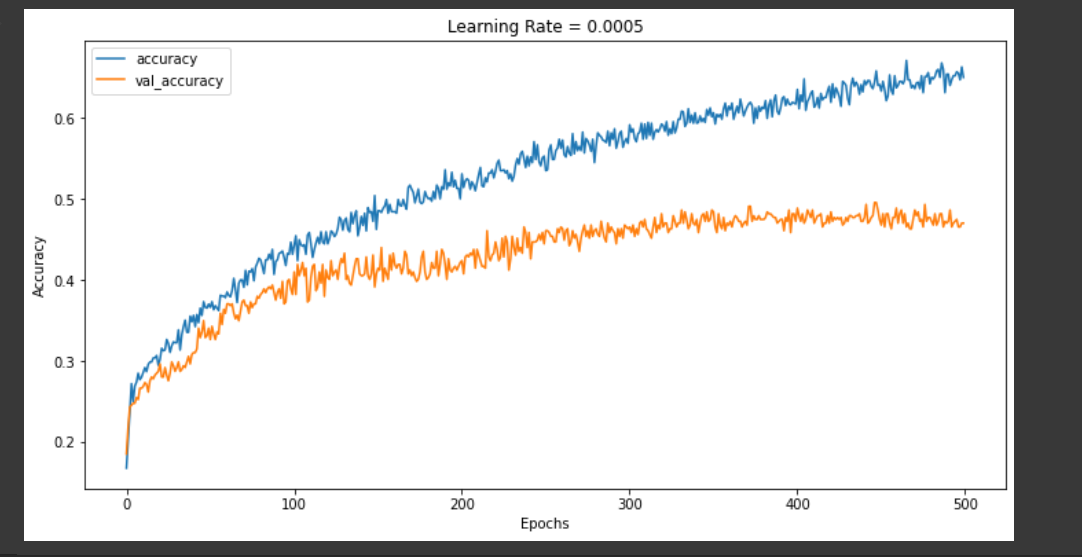


Lr = 0.0005 epochs = 500

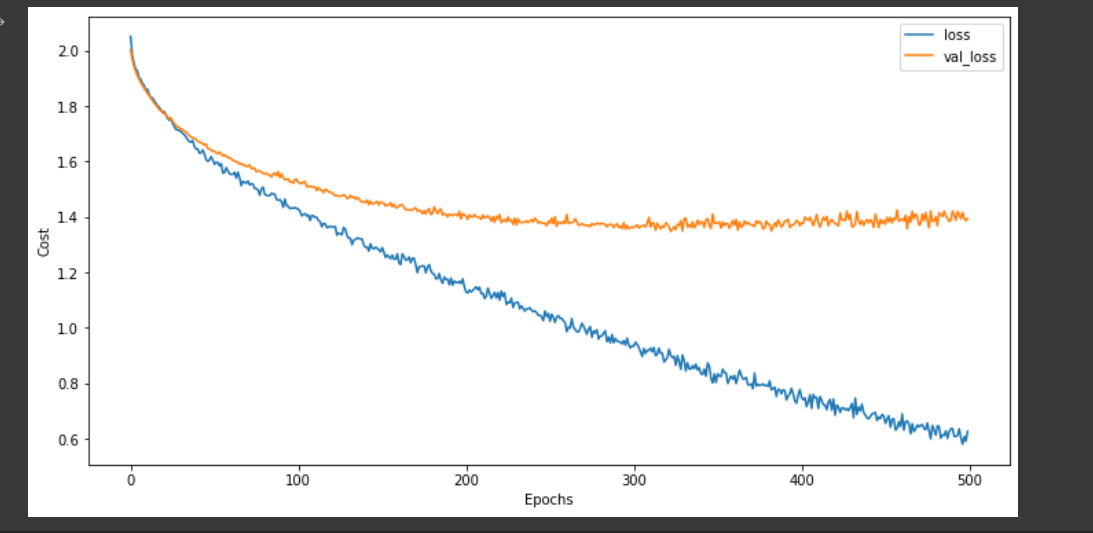


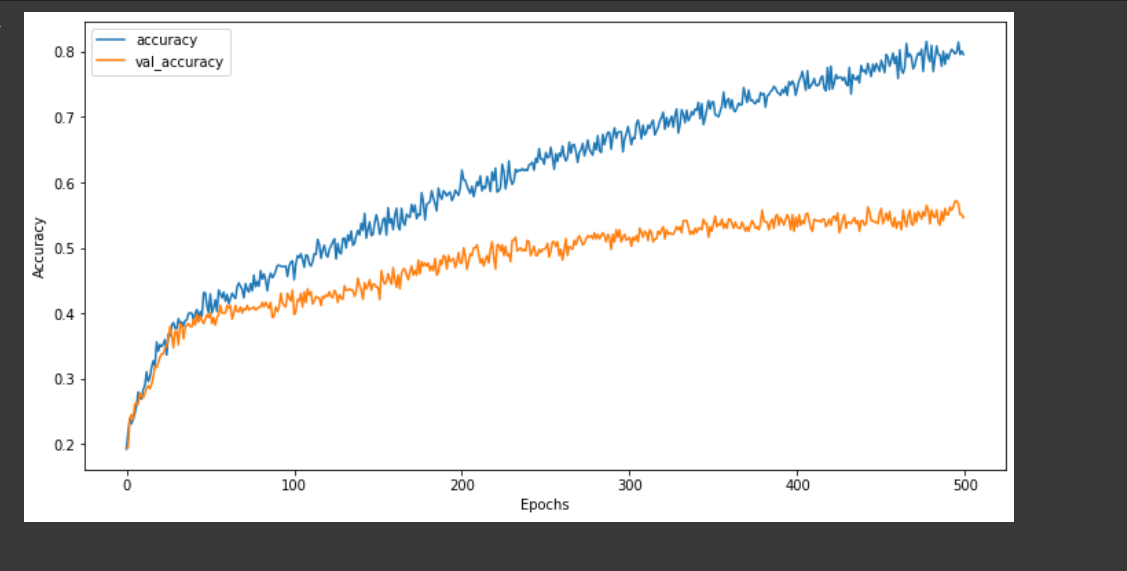


Lr = 0.0005 epochs = 500 produced better results than before but the cost still seems to saturate further so I reduced the lr further to check if its better

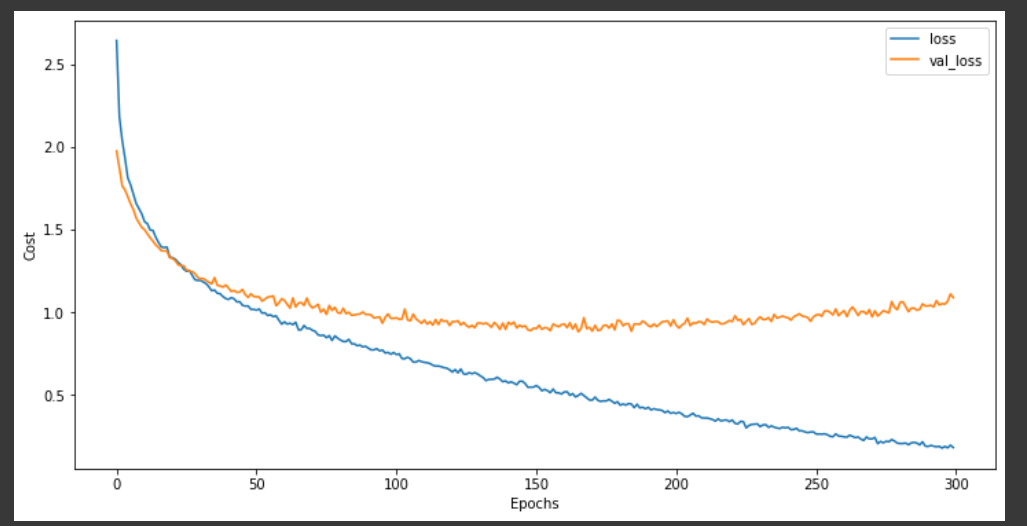


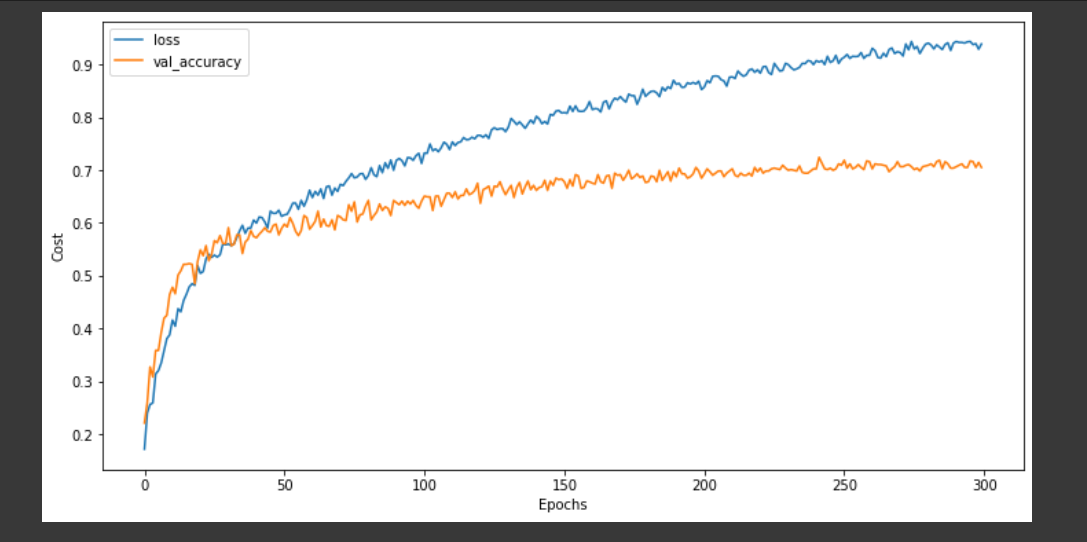
* Changed the entire network architecture to help get a better result and to tackle plateaus used a new method called ReduceLRonPlateau to reduce the learning rate when the learning hits a plateau and thus ensure that the model does not stop learning until a defined minimum learning rate. This improved the training process but was counterproductive as the learning rate got really small fast which hindered with the training process(making it really slow prematurely).
* Tried out different architectures and learning rates to help get a better accuracy.
* Added dropouts to help reduce the overfitting problem
* Experimented with the hyperparameters to get a better accuracy.
* Got an accuracy of 44% with data consisting of MFCC and Mel Spectrogram features.
* Added in more spectrogram features like Chroma and Tonnetz to get more features for better results which increased the accuracy of the model to 54.5%.
* Error and accuracy melspec+mfcc+chroma





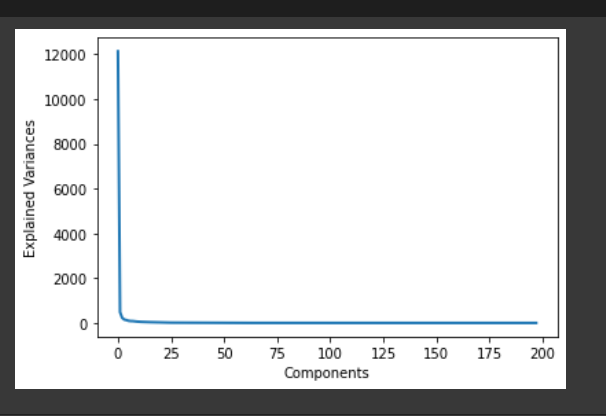
* Researched and appended more data (data from the same dataset but song data with emotions labelled) and tested the models accuracy. The model performed better and achieved an accuracy of 70.5%.





**SVM:**

* Extracted the mfcc and mel spectrogram features from the dataset
* Split the dataset into dependent and independent variables and encoded the label using labelencoding
* Trained a Support Vector Classifier with the data consisting of these extracted features and got an accuracy of 45%
* Trained the SVM with additional features like Chroma and tonnetz and got a testing accuracy of 51.3%
* Performed a general pca to find the optimal number of components that the feature set can be reduced to



* Performed PCA on the same dataset with number of components=50 and got an accuracy of 52.35% which is not a significant increase.
* Researched and appended more data (data from the same dataset but song data with emotions labelled) and tested the models accuracy. The model performed better and achieved an accuracy of 60% with pca and 58.9% without.

**Tasks for next week:**

* Explore more sources of data to help improve the accuracy of the model.
* Explore more models to see if a better accuracy can be achieved.
* Explore concepts like data augmentation to improve the test accuracy of the model
* Implement with other pre-trained model and also explore ensemble machine learning methods.
* Once all these have been explored, move on to the final steps of the project(after model finalization).