First tried to know about various audio features and significance of those features in deciding the mood. As the number of features were huge, so to avoid the curse of dimensionality, started with those features which were important and easy to handle in the first place.  
  
Even after selecting important features, number of features were around 440. Now tried to reduce the more number of features and to select only important ones.

Tried various approach for feature selection:

1. Filter based using correlation/chi-square, which gave important features.
2. Used recursive feature elimination and step-wise model selection.

Getting important features and building classifier like non-liner svm, neighbourhood classifier, naïve-bayes, but all of these classifier did not give good result even on selected important features. Tried PCA to reduce the feature space but it did not work well on the above classifiers.  
  
Then I used embedded method for feature-selection, which uses built in feature-selection methods. I tried tree-based classifier for this purpose. In tree-based classifier, there is in-built feature selection strategy, using entropy/information-gain/Gini-coefficient. Therefore, in tree based classifier there in no need to do feature selection explicitly and there is no need to do the scaling and normalization of features as well.  
  
So I tried following tree based classifier:  
  
1. XGboost  
2. Gradient Boosting Classifier  
3. AdaBoost tree based classifier  
4. Random forest  
5. Bagging Classifier (tree as base estimator)  
  
Used regularization parameters to do feature elimination in above classifiers and give some weight to the minority classes as well. Some above-mentioned classifiers were facing overfitting issue, which was due to less data set. Tried to reduce overfitting using hyper-parameter tuning, but reducing overfitting using hyper-parameters tuning was leading to bad f1-score. This type of overfitting can be reduced using more training data.  
  
Some of the above-mentioned classifiers were good in predicting HAPPY classes while others were good in predicting SAD classes. Therefore, tried different approach to build a classifier, which is good at predicting both the classes.   
  
To reduce biases of each above mentioned classifier, **I used stacked ensemble method**, in which all the above classifier were stacked together and then there is another classifier (XGBoost in our case) which will take base classifer’s prediction as input and then make a balanced prediction over those inputs. The prediction of stacked classifier is good in sense that it is good in predicting both the classes.  
  
To do the hyper-parameter tuning, tried Grid search and Random search techniques along with cross validation.  
  
Another problem was the imbalanced training data. As in the data, around 600 entries were for SAD song and 1600 for HAPPY songs.

To handle this skewed class problem, I had few options, over-sample the minority class using the SMOTE (Synthetic Minority Oversampling Technique) or getting the more data for the minority class using manual methods but it did not work well as it was not able to capture true data pattern.  
  
Another option was to under-sample the majority class, but after doing under sampling, dataset became very small comprising 1200 data rows, so it was leading to over fitted model and giving the high variance.  
  
Another approach was to give higher weights for the minority class, I did this in current algorithm but it is not giving very significant improvement in the model performance.

If data set is more and balanced as well, then current algorithm will perform very well.