1. **AdaBoost in Music Emotion Recognition**

AdaBoost (Freund, Yoav; Schapire, Robert E (1997). "A decision-theoretic generalization of on-line learning and an application to boosting". *Journal of Computer and System Sciences*. **55**: 119–139. [CiteSeerX](https://en.wikipedia.org/wiki/CiteSeerX_(identifier)" \o "CiteSeerX (identifier)) [10.1.1.32.8918](https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.8918). [doi](https://en.wikipedia.org/wiki/Doi_(identifier)):[10.1006/jcss.1997.1504](https://doi.org/10.1006%2Fjcss.1997.1504)) or Adaptive Boosting, is one of the popular boosting algorithms used in solving variety of machine learning problems. As a the name implies this algorithm relies on boosting methods were a sequence of weak learners collectively operate to generate an optimal prediction outcome. As a ensemble method, AdaBoost also offer flexibility to apply any classification algorithm as the underlying learning algorithm that forms the backbone of the weak learners. Despite the popularity, flexibility and robustness, AdaBoost is rarely used in Music Emotion Recognition (MER) to generate predictive emotional classifications. This chapter illustrates application of AdaBoost in MER and to use the model to determine the emotion conveyed by Carnatic music compositions tuned to ragas Mayamalavagaula, Bhairavi, Kalyani, Kedaragaula, Khamboji, Mohanam, Shankarabharanam and Todi, as illustrated in section 4.8: Validation datasets of this document.

In order to ascertain the capability of AdaBoost, three learning algorithms were supplied to the boosting model viz Decision Tree Classifier, Support Vector Machine Classifier and Multi Level Perceptron classifier, to evaluate the performance of each classifier and to determine most suitable classifier for weak leaners.

* 1. **AdaBoost Intuition**

The process of generating an outcome in AdaBoost consists of three stages viz – Scouting, Drafting and Weight assignment ( [Rojas, Raúl](https://en.wikipedia.org/wiki/Ra%C3%BAl_Rojas)*(2009).*["AdaBoost and the super bowl of classifiers a tutorial introduction to adaptive boosting"](http://www.inf.fu-berlin.de/inst/ag-ki/adaboost4.pdf)(Tech. Rep.)*. Freie University, Berlin.*

Scouting primarily involves testing the classifiers by evaluating the predictive outcome of each classifier. Correct and wrong predictions are penalized with a non-zero cost (β >0), e-β for correct predictions and eβ for wrong prediction so that wrong predictions are penalized heavily than right predictions. Once the classifiers are listed with appropriate penalty, a pool of classifiers are available for AdaBoost to make a systematic prediction. AdaBoost then systematically proceeds through the pool by extracting one classifier from the pool during each iteration. The elements in the data set are weighted according to their current relevance (or urgency) at each iteration. At the beginning, all elements are assigned the same weight (just 1, or 1/N if we want to have a total sum of 1 for all weights). As the drafting progresses, the more difficult examples, that is, those where the committee still performs badly, are assigned larger and larger weights thereby reducing the overall predictive variance of the model.

Drafting involves ranking the classifiers and choosing the best classifier from the pool such that by the end of drafting process, a linear combination of classifiers is available to performing the prediction process. Once the drafting is complete each drafted classifier is assigned a weight.

The pseudocode of the AdaBoost Algorithm is illustrated below:

***TBD***

**Explain about multi-class classification using AdaBoost and SAMME & SAMME.R boosting algorithms along with pseudocode.**

* 1. **Training, Test and Validation datasets**

The dataset obtained as a result of adopting influential dataset strategy, explained in chapter 4, was used to develop the MER model. The class variable in this dataset (N = 6287), *sound\_file\_class*, which is categorical variable, was converted into an ordinal variable and renamed as *sound\_file\_class\_num*. This conversion was specifically done to aid in the MER modelling. Further to this, the dataset was divided into two sets viz. *model data set* and *validation data set* with partition ratio of 0.9 and stratified on *sound\_file\_class\_num* such that 90% of the observations are available for modelling along with 10% of observations as validation set.

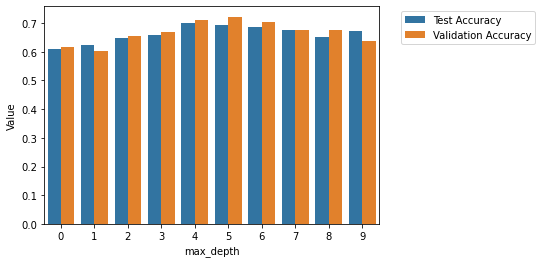
During the modelling process *model data set* is used and was split into *train* and *test* datasets with partition ratio of 0.8 and stratified on *sound\_file\_class\_num.*

* 1. **Appraising base estimator, Decision Tree Classifier**

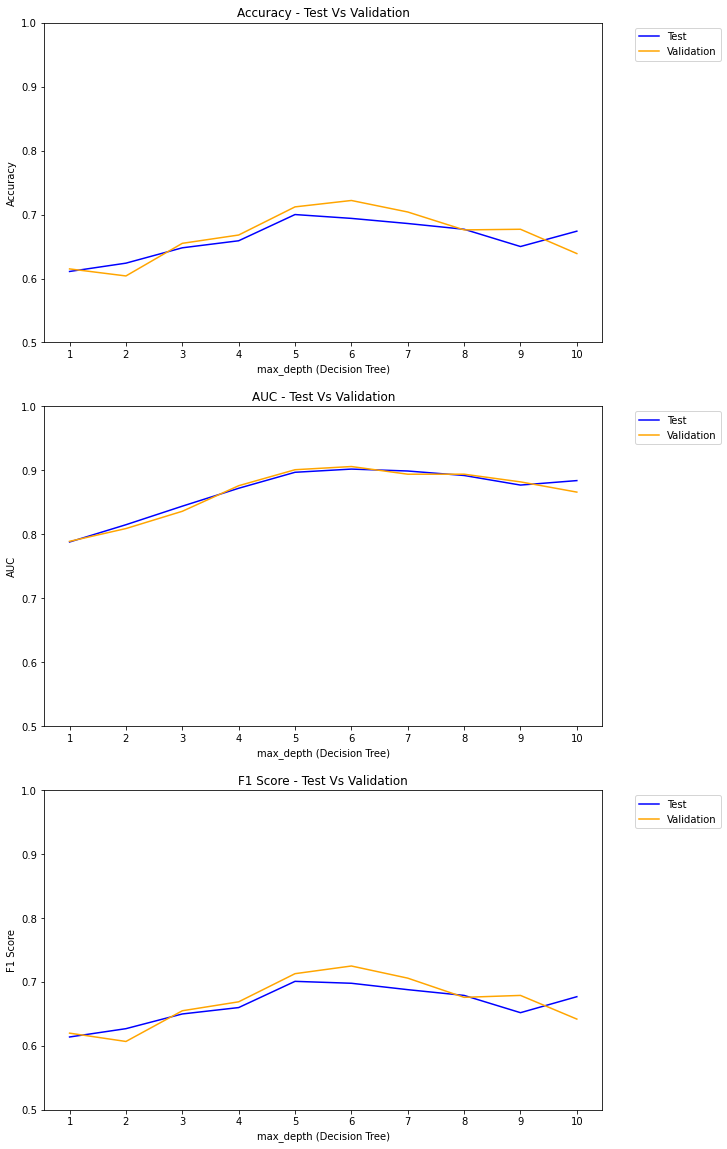
With the dataset chosen and split into appropriate train, test and validation datasets, it was time to start developing the base model. The fundamental assumption taken during this step was that the base estimator passed to AdaBoost would be Decision Tree Classifier. Moreover, during the development of “influential dataset strategy”, it was observed that the optimal values of AdaBoost parameters – *n\_estimators* and *learning\_rate* would be 1500 and 0.1 respectively (Refer to section: tbd - Finding optimal dataset for modelling in chapter 4). The modelling process, explained here, utilizes this finding in further optimizing/fine tuning the MER model.

The assumption of using base estimator as Decision Tree Classifier invokes a critical task to be performed i.e. to determine the optimal depth of the Decision Tree so that it doesn’t overfit. Hence to determine the most appropriate value for the parameter *max\_depth,* a simulation of 10 trials was performed by passing various values to *max\_depth* parameter, starting from 1 to 10 during each trial, and by keeping all other parameters constant (default values as assigned by the Decision Tree Classifier of sklean library in Python). The training and test datasets were kept constant during each trial so as to avoid any sampling bias that may vary the optimal parameter determination. During each trial the model with varying max\_depth parameter for Decision Tree Classifier, was evaluated against the validation dataset – created and kept anonymous as hold out, as explained in section 5.2. The metrics considered for model evaluation and optimal parameter determination were accuracy, AUC and F1 Score (*elaborate: TBD*).

The bar char below indicates the variation of Test and Validation accuracies during each trial.



It can be observed that after trial 5 & 6 (where max\_depth = 5 and 6) the validation accuracies were dropping. Hence the max\_depth = 6 was chosen as the optimal value that would be passed to Decision Tree Classifier. The trend charts below, for accuracy, AUC and F1 Score, also supports the determination that the optimal parameter value for *max\_depth* as it can be observed visually that the validation accuracies diminish after the value of 6.

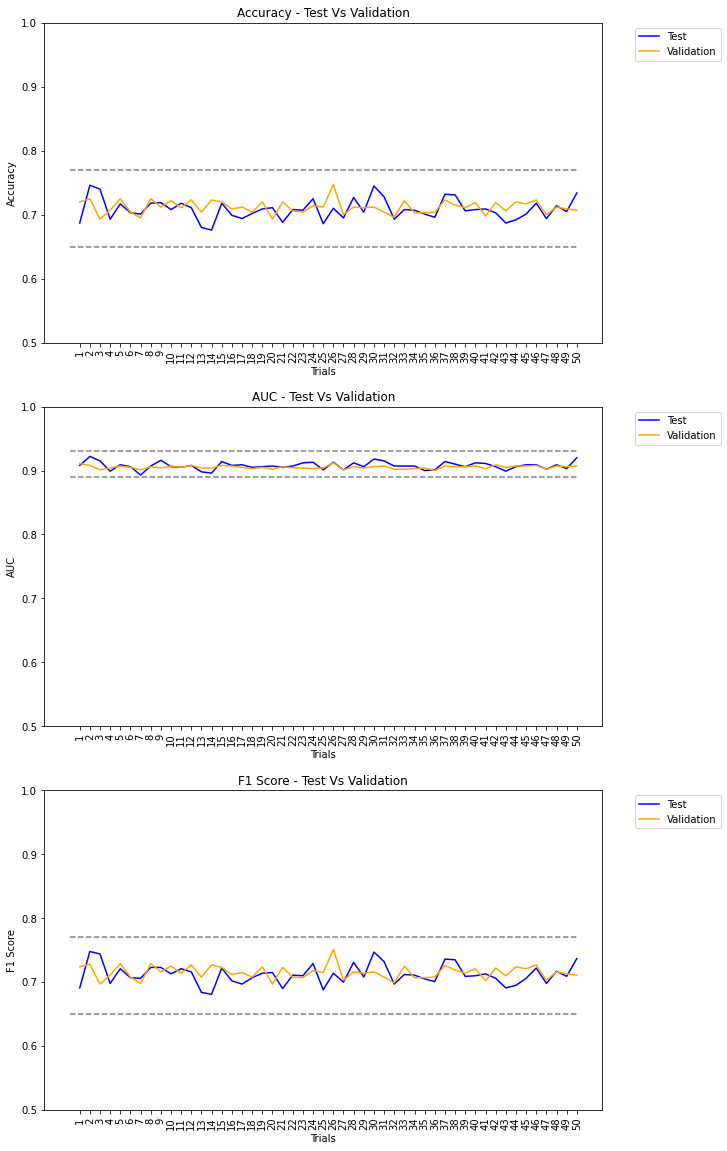


* 1. **Checking model robustness**

Having determined the optimal parameters for AdaBoost i.e. parameters that may potentially offer best model outcome viz. *base estimator* = Decision Tree Classifier with max\_depth = 6, *n\_estimators* = 1500 and *learning\_rate* = 0.1, its is pertinent to evaluate robustness of a model with these parameters by passing various samples of data to the model (contrary to earlier stage of determining optimal *max\_depth* parameter) and evaluating each model against the validation set generated earlier as explained in section 5.2. The evaluation, involving simulation containing 50 trials, was designed to develop separate model in each trial by performing a stratified random sampling on the model dataset to draw training and test data set. Each model was then evaluated against the validation data set to determine accuracy, AUC and F1 Score metrics as indicated in the table below.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Trial #** | **Accuracy** | | **AUC** | | **F1 Score** | |
| **Test** | **Validation** | **Test** | **Validation** | **Test** | **Validation** |
| 1 | 0.687 | 0.720 | 0.908 | 0.911 | 0.691 | 0.724 |
| 2 | 0.746 | 0.725 | 0.922 | 0.908 | 0.748 | 0.728 |
| 3 | 0.740 | 0.693 | 0.915 | 0.901 | 0.744 | 0.697 |
| 4 | 0.693 | 0.707 | 0.899 | 0.904 | 0.698 | 0.711 |
| 5 | 0.717 | 0.725 | 0.909 | 0.907 | 0.721 | 0.729 |
| 6 | 0.703 | 0.704 | 0.906 | 0.905 | 0.707 | 0.708 |
| 7 | 0.701 | 0.695 | 0.893 | 0.901 | 0.706 | 0.698 |
| 8 | 0.718 | 0.725 | 0.907 | 0.906 | 0.723 | 0.729 |
| 9 | 0.719 | 0.712 | 0.916 | 0.904 | 0.723 | 0.716 |
| 10 | 0.708 | 0.722 | 0.906 | 0.907 | 0.713 | 0.725 |
| 11 | 0.718 | 0.711 | 0.905 | 0.905 | 0.721 | 0.714 |
| 12 | 0.711 | 0.723 | 0.908 | 0.908 | 0.716 | 0.727 |
| 13 | 0.680 | 0.704 | 0.898 | 0.904 | 0.684 | 0.708 |
| 14 | 0.676 | 0.723 | 0.896 | 0.904 | 0.681 | 0.727 |
| 15 | 0.718 | 0.720 | 0.914 | 0.908 | 0.722 | 0.723 |
| 16 | 0.699 | 0.709 | 0.908 | 0.907 | 0.702 | 0.712 |
| 17 | 0.694 | 0.712 | 0.909 | 0.905 | 0.697 | 0.715 |
| 18 | 0.702 | 0.704 | 0.905 | 0.903 | 0.707 | 0.708 |
| 19 | 0.709 | 0.720 | 0.906 | 0.905 | 0.714 | 0.724 |
| 20 | 0.711 | 0.693 | 0.907 | 0.902 | 0.715 | 0.697 |
| 21 | 0.688 | 0.720 | 0.905 | 0.906 | 0.690 | 0.723 |
| 22 | 0.708 | 0.706 | 0.907 | 0.904 | 0.711 | 0.708 |
| 23 | 0.707 | 0.704 | 0.912 | 0.904 | 0.710 | 0.707 |
| 24 | 0.725 | 0.714 | 0.913 | 0.903 | 0.729 | 0.718 |
| 25 | 0.686 | 0.712 | 0.901 | 0.904 | 0.688 | 0.715 |
| 26 | 0.710 | 0.747 | 0.913 | 0.912 | 0.714 | 0.751 |
| 27 | 0.695 | 0.700 | 0.901 | 0.901 | 0.700 | 0.703 |
| 28 | 0.727 | 0.712 | 0.912 | 0.906 | 0.731 | 0.716 |
| 29 | 0.704 | 0.711 | 0.906 | 0.904 | 0.708 | 0.714 |
| 30 | 0.745 | 0.712 | 0.918 | 0.906 | 0.747 | 0.716 |
| 31 | 0.728 | 0.704 | 0.915 | 0.907 | 0.732 | 0.708 |
| 32 | 0.693 | 0.696 | 0.907 | 0.902 | 0.697 | 0.699 |
| 33 | 0.708 | 0.722 | 0.907 | 0.902 | 0.712 | 0.725 |
| 34 | 0.707 | 0.703 | 0.907 | 0.903 | 0.711 | 0.707 |
| 35 | 0.701 | 0.703 | 0.900 | 0.904 | 0.705 | 0.707 |
| 36 | 0.696 | 0.704 | 0.901 | 0.900 | 0.701 | 0.708 |
| 37 | 0.732 | 0.723 | 0.914 | 0.907 | 0.736 | 0.726 |
| 38 | 0.731 | 0.715 | 0.910 | 0.906 | 0.735 | 0.719 |
| 39 | 0.706 | 0.711 | 0.906 | 0.906 | 0.709 | 0.714 |
| 40 | 0.708 | 0.719 | 0.912 | 0.907 | 0.710 | 0.721 |
| 41 | 0.709 | 0.698 | 0.911 | 0.903 | 0.713 | 0.702 |
| 42 | 0.703 | 0.719 | 0.906 | 0.909 | 0.706 | 0.722 |
| 43 | 0.687 | 0.706 | 0.899 | 0.905 | 0.691 | 0.710 |
| 44 | 0.692 | 0.720 | 0.906 | 0.907 | 0.695 | 0.724 |
| 45 | 0.701 | 0.717 | 0.909 | 0.907 | 0.706 | 0.721 |
| 46 | 0.718 | 0.723 | 0.909 | 0.908 | 0.722 | 0.727 |
| 47 | 0.694 | 0.700 | 0.902 | 0.902 | 0.698 | 0.703 |
| 48 | 0.714 | 0.712 | 0.909 | 0.907 | 0.717 | 0.716 |
| 49 | 0.705 | 0.709 | 0.903 | 0.906 | 0.709 | 0.713 |
| 50 | 0.734 | 0.707 | 0.920 | 0.907 | 0.737 | 0.711 |

The trends of metrics considered is illustrated in the plots below.



The summary statistics of the metrics is show in the table below.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Summary  Statistic** | **Accuracy** | | **AUC** | | **F1 Score** | | |
| **Test** | **Validation** | **Test** | **Validation** | **Test** | **Validation** | |
| Mean (µ) | 0.708 | 0.712 | 0.908 | 0.905 | 0.712 | | 0.715 |
| Std. Dev (σ) | 0.016 | 0.010 | 0.006 | 0.003 | 0.016 | 0.010 | |
| CI – UB  (µ+1.96 \* σ) | 0.740 | 0.732 | 0.919 | 0.910 | 0.743 | 0.736 | |
| CI – LB CI  (µ- 1.96 \* σ) | 0.677 | 0.692 | 0.896 | 0.900 | 0.681 | 0.695 | |

* 1. **Choosing the boosting algorithm for AdaBoost**

**S**tagewise **A**dditive **M**odeling using a **M**ulti-class **E**xponential loss function (SAMME) (Zhu, H. Zou, S. Rosset, T. Hastie, “Multi-class AdaBoost”, 2009) and **S**tagewise **A**dditive **M**odeling using a **M**ulti-class **E**xponential loss function for **R**eal (SAMME.R) (Zhu, H. Zou, S. Rosset, T. Hastie, “Multi-class AdaBoost”, 2009) are commonly used multi-class classification algorithm used in boosting methods. AdaBoost can also be enhanced to perform multi-class classification using these algorithms.

In order to choose the best boosting algorithm to be selected two AdaBoost models were developed with the *algorithm* parameter as SAMME and SAMME.R respectively using a single trial. The outcome of both models is illustrated in the table below.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Algorithm** | **Metric** | **Emotion Class** | | | |
| **0 (Q1)** | **1 (Q2)** | **2 (Q3)** | **3 (Q4)** |
| Test | SAMME.R | Precision | 0.58 | 0.95 | 0.66 | 0.70 |
| Recall | 0.78 | 0.69 | 0.64 | 0.66 |
| F1 Score | 0.66 | 0.80 | 0.65 | 0.68 |
| Accuracy | 0.693 | | | |
| SAMME | Precision | 0.70 | 0.95 | 0.72 | 0.78 |
| Recall | 0.80 | 0.81 | 0.73 | 0.77 |
| F1 Score | 0.75 | 0.88 | 0.72 | 0.77 |
| Accuracy | 0.778 | | | |
| Validation | SAMME.R | Precision | 0.60 | 0.94 | 0.70 | 0.74 |
| Recall | 0.81 | 0.73 | 0.67 | 0.68 |
| F1 Score | 0.69 | 0.83 | 0.69 | 0.70 |
| Accuracy | 0.722 | | | |
| SAMME | Precision | 0.66 | 0.92 | 0.71 | 0.77 |
| Recall | 0.79 | 0.82 | 0.70 | 0.71 |
| F1 Score | 0.72 | 0.87 | 0.70 | 0.74 |
| Accuracy | 0.754 | | | |

Inferences: **TBD**

* 1. **Modelling with the original emotion categorization (4 classes)**

With the analysis conducted so far, it was observed that the following combination of parameters and algorithm would be most optimal while explaining MER using AdaBoost. Those are:

* n\_estimators = 1500
* learning\_rate = 0.1
* base\_estimator = Decision Tree Classifier
* boosting algorithm = SAMME

A final model with these options was developed with the independent variable as the *sound\_file\_class\_num* which is an ordinal transformation of *sound\_file\_class* which contains the original emotion categorization viz Q1, Q2, Q3 and Q4 (refer to chapter 4: Data Processing). The classification report of the model outcome is illustrated as follows.

* 1. **Inferences**
  2. **Predictions**