ECE 763
Project 2

Adaptive Boosting (AdaBoost) for Face Detection

Spring 2022

Sahil Avinash Deshpande **200375597**

Project 2.1

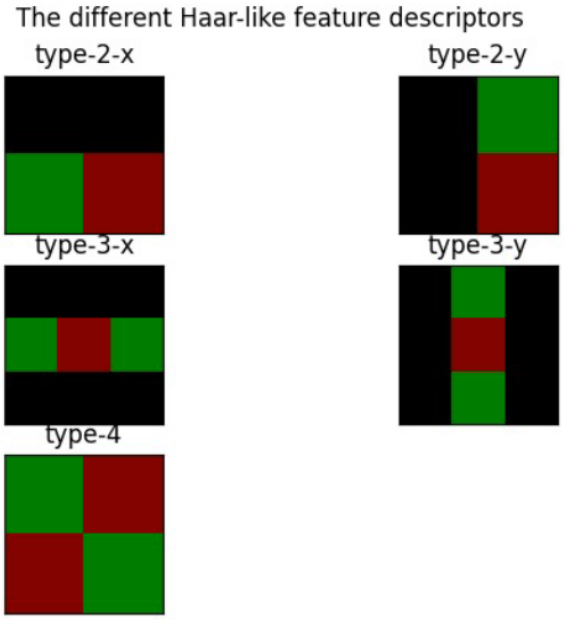
- ### Data:

I used the same dataset used in project 1 for training and testing this model which contains 16x16 grayscale images.

I used 800 images for the training data (400 face and 400 nonface) and
200 images for the testing (100 face and 100 nonface).

- ### Features:

The following 5 types of haar features were used as weak classifiers. This includes edge, line and four rectangle features.



The AdaBoost algorithm was implemented to produce a strong classifier from the combinations of all the weak classifiers.

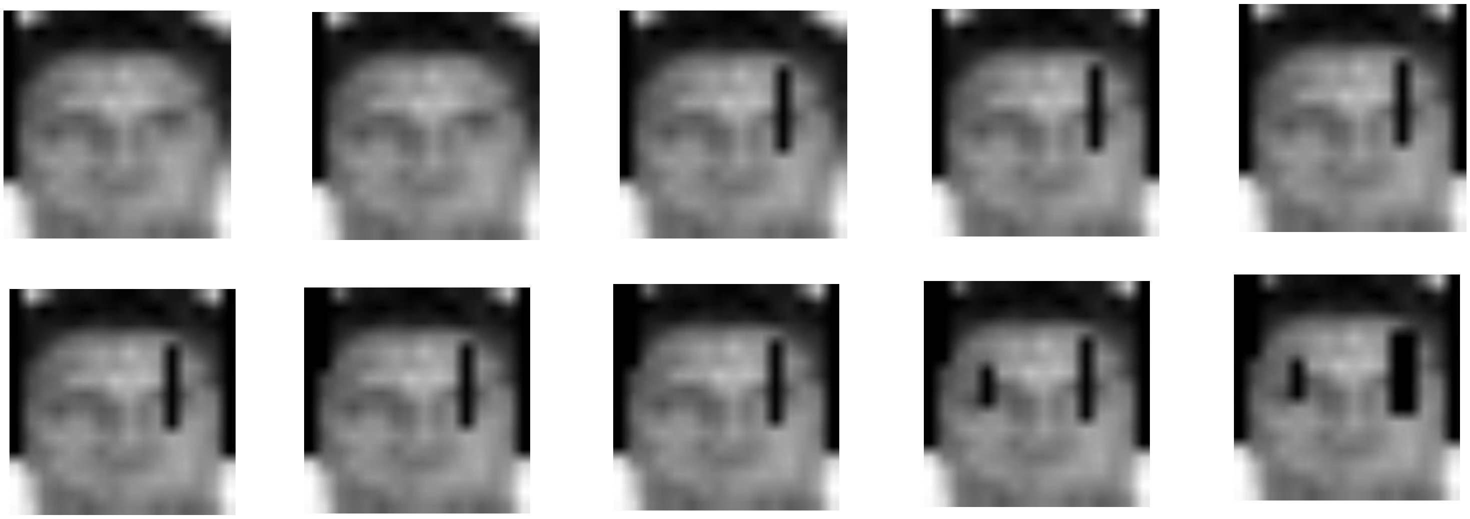
Threshold value for any weak learner is given by taking multiple values from the minimum and maximum of all feature values

- **AdaBoost Algorithm:**

  AdaBoost is used to boost the classification performance of a weak learning algorithm. AdaBoost adds together the effect of all weak classifiers to make a strong classifier. In the AdaBoost algorithm, first the weights are initialized, and error of each weak classifier is calculated. The weak classifier with minimum error is chosen for the AdaBoost iteration. Then the weights are updated after every iteration and training accuracy of the strong classifier is calculated.

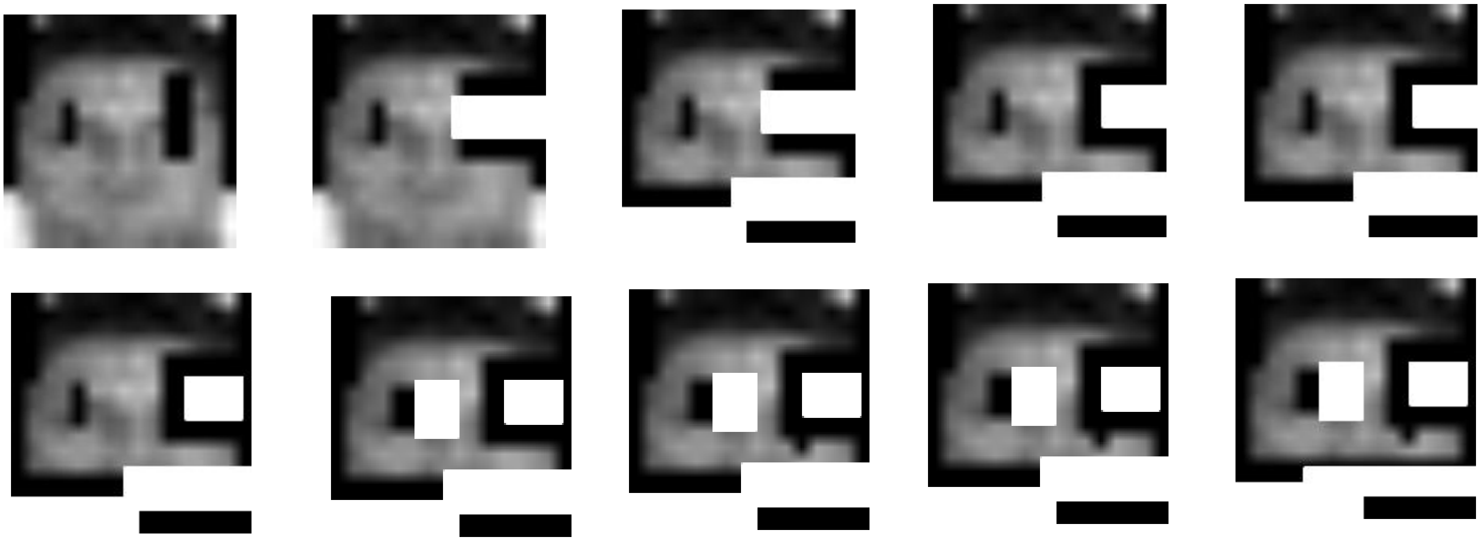
  I have used 10 iterations of the AdaBoost algorithm in my code.

- **Top 10 features before AdaBoost:**



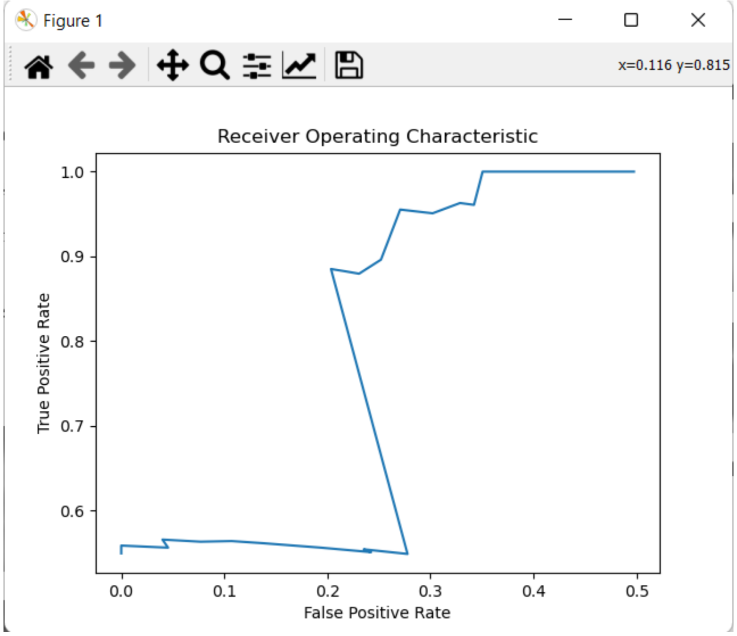
Training accuracy after 10 iterations of AdaBoost: 89.25%

- **Top 10 features after AdaBoost:**



Test Accuracy: 83.5%

ROC Curve:



Project 2.2

After refactoring my own code in the provided reference code framework, the following points were observed:

1. Bigger functions are combined inside individual methods which makes the code easy to understand.
2. The two utility files to check the dataset paths and visualizing mean, covariance matrix and plotting the ROC curve, which can be imported into the main function make the code modular.
3. Individual python files created for the each of the models makes the code modular.
4. Separate python file created for the EM Algorithm makes it easy to reuse it in each of the models made and thus saves a lot of time.