Adaboost Report:

(1) a description of how you formulated the problem

I used the algorithm from Artificial Intelligence: A Modern Approach, Stuart Russell, Peter Norvig which is explained on canvas.

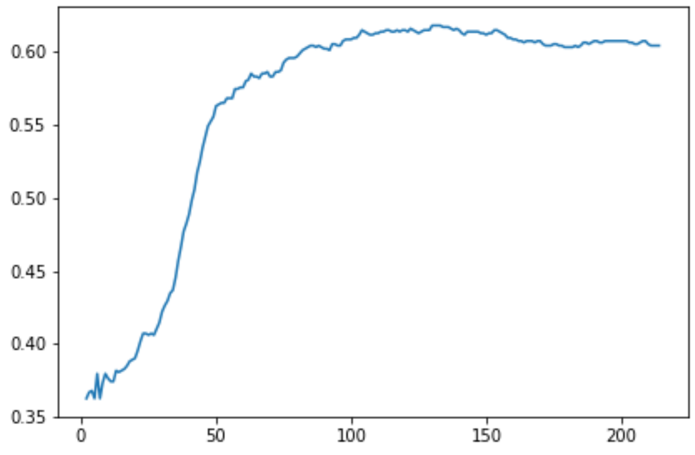
I also referred this paper for weights updating and overall decision stump weight(a)

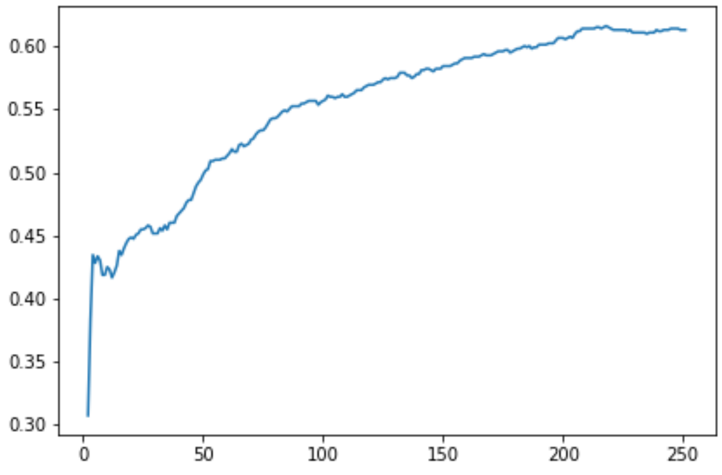
https://web.stanford.edu/~hastie/Papers/samme.pdf

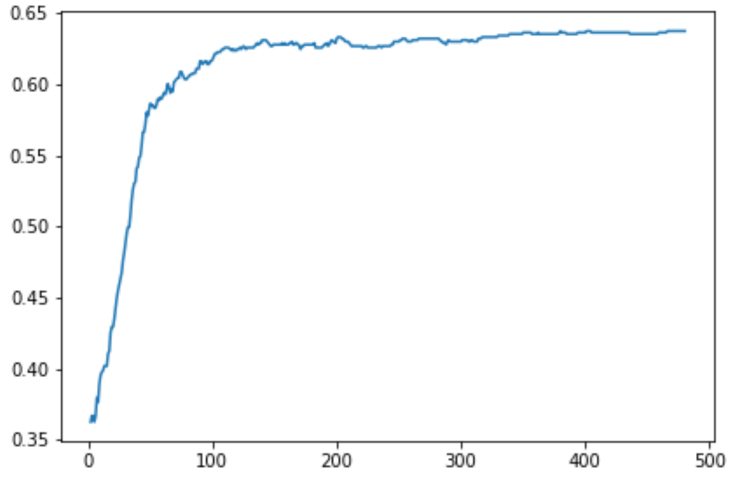
The train_pixels contains both the pixels and the orientation. Similarly the test_pixels.

As suggested in the given assignment itself, I generated random indices and compared the pixel values at those indices in a given image.

So, first I created all the possible indices and generated 500 random indices from this list. I tried with different number of decision stumps. The x axis is number of decision stumps and y axis is accuracy







These are the different graphs I got.

By taking last one as the reference, I used 500 decision stumps.

2) Brief description of how your program works:

So, every decision stump is used to classify all the images into positive and negative depending on whether the pixel at first index is greater than or less than the second.

Then the orientation with maximum mode in positive is assigned to all the images in this class. Similarly for negative.

Based on this I calculated the error. For better accuracy I ignored the decision stumps which give error>0.75

Then I calculated the weight for each decision stump
a = np.log((1-error)/error) + np.log(3)

Updated the weights of the misclassified images.

Store the decision stump, positive, negative angle and the a value in a dictionary and saved it in model file.

Testing:

For every test images I used the decision stumps for the model file and compared the pixels at those indices. If the image comes under positive class i.e if the pixel at first index is greater than the second then, I increase positive angle weight by w. Otherwise negative. After classifying a single test image with all the decision stumps, the one with maximum weight will be the final classification.

3) Discussion of any problems, assumptions or simplifications.

Initially the program was really slow, then I used numpy arrays. First I tried to build 6 binary classifiers, the accuracy was almost same, but it took lot of time. So I buit a multi class classifier.