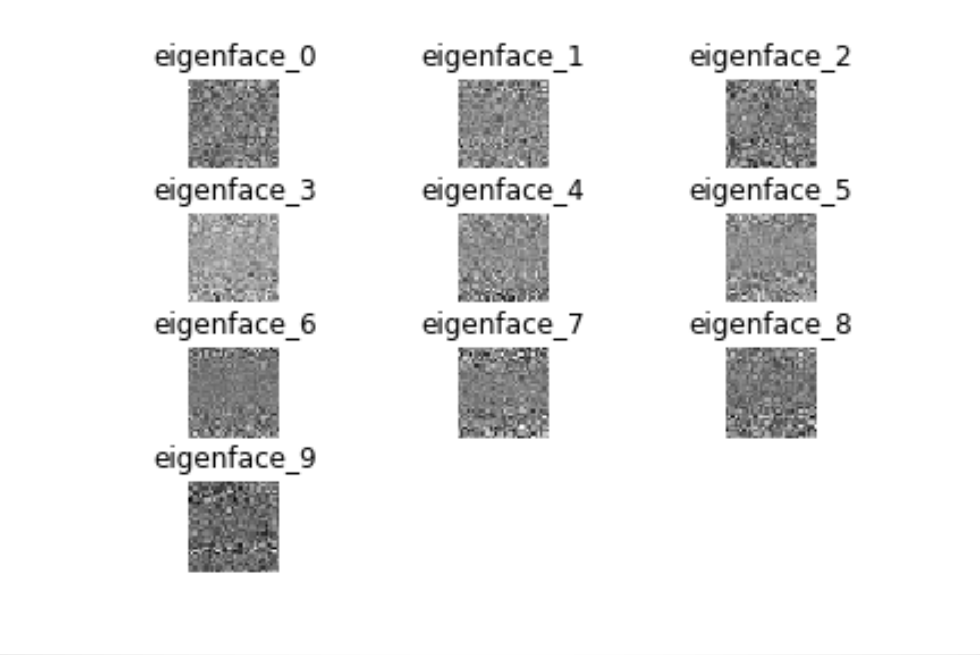
Computer Vision
Fall-2019
Problem Set #6

Sahil Dhingra
sdhingra31@gatech,edu

1a: Average face

ps6-1-a-1

1b: Eigenvectors



ps6-1-b-1

1c: Analysis

Analyze the accuracy results over multiple iterations. Do these “predictions” perform better than randomly selecting a label between 1 and 15? Are there any changes in accuracy if you try low values of k? How about high values? Does this algorithm improve changing the split percentage p?

Yes, these predictions perform better than random selection, as accuracy varies from 35%-60%, which is much better than what the accuracy would be for random selection at 7%.
The accuracy isn’t much different at lower or higher k values, as most of the information would be captured within low values of k, because most of the variance is captured in lower eigenvectors.
The accuracy is low for lower p percentages (10%, 20%) but increases with more training data, but does not improve after a certain p, around 50%, at which point the model has sufficient training data to learn

2a: Average accuracy

Report the average accuracy over 5 iterations. In each iteration, load and split the dataset, instantiate a Boosting object and obtain its accuracy.

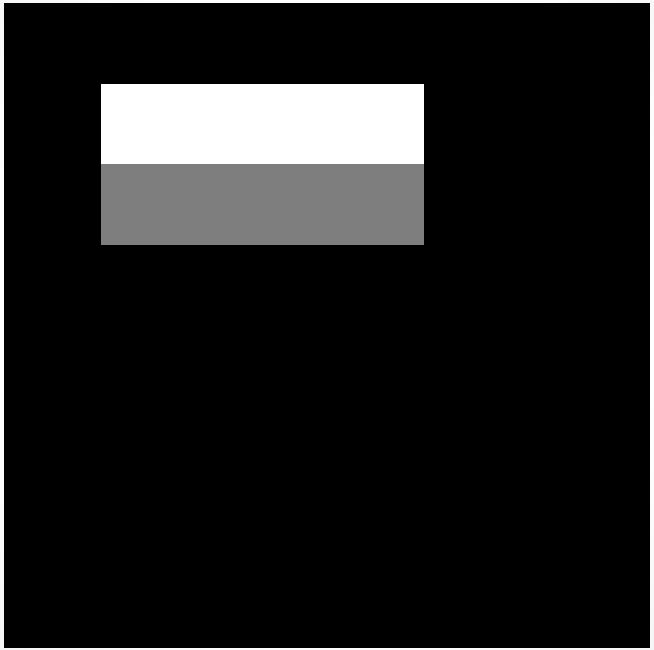
(Random) Training accuracy: 49.14%
(Weak) Training accuracy 87.17%
(Boosting) Training accuracy 93.90%
(Random) Testing accuracy: 55.00%
(Weak) Testing accuracy 86.88%
(Boosting) Testing accuracy 95.00%

2a: Analysis

Analyze your results. How do the Random, Weak Classifier, and Boosting perform? Is there any improvement when using Boosting? How do your results change when selecting different values for num_iterations? Does it matter the percentage of data you select for training and testing (explain your answers showing how each accuracy changes).

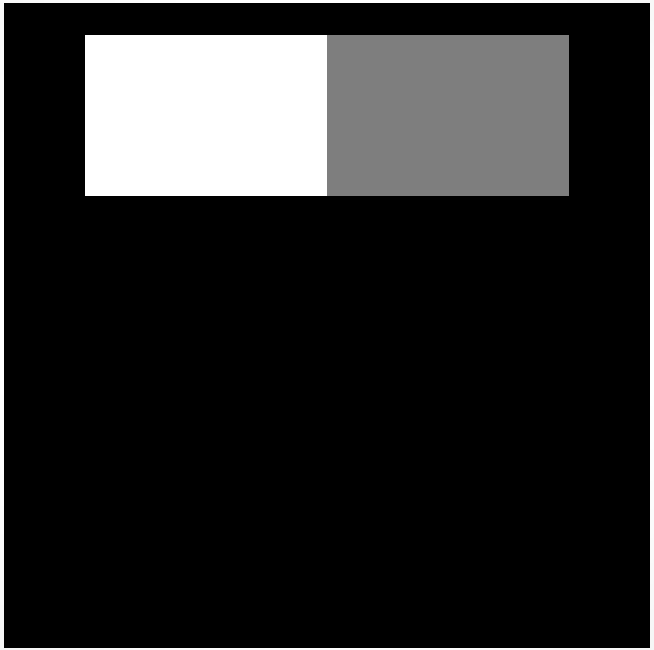
Random classifier is more or less about 50% accurate. Weak Classifier works significantly better at about 80-90% accuracy, on both train and test data. Boosting is even more accurate, at about 90-95% in most tests with 5 iterations. It increases up to about 95% for 20 iterations but then starts dropping sharply for 30-40 iterations, because Boosting model starts fitting to the noise after a certain number of iterations.
The model seems to perform fairly well, even with small training data i.e. at low percentages. Only for training date less than 10%, the model accuracy deteriorates, quite robust otherwise.

3a: Haar Features



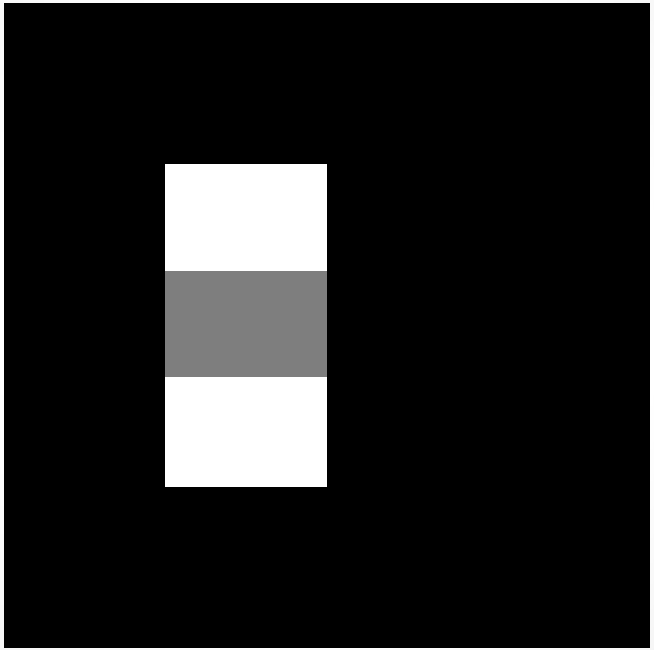
ps6-3-a-1

3a: Haar Features



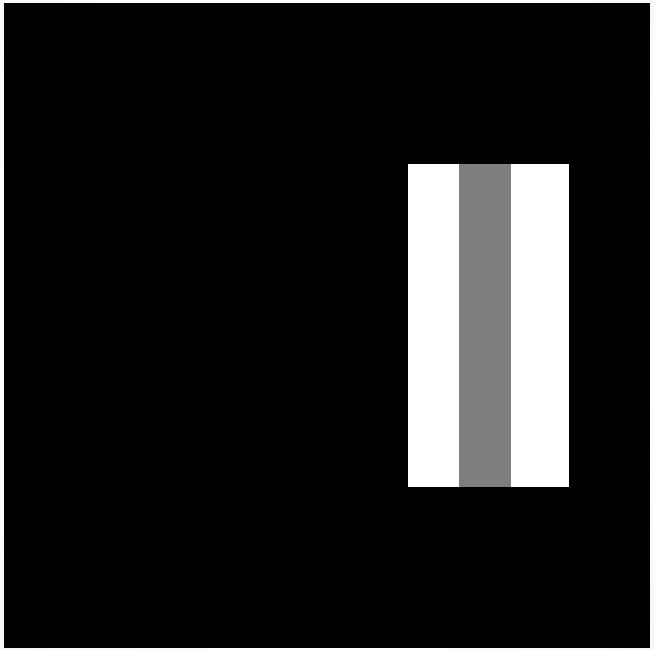
ps6-3-a-2

3a: Haar Features



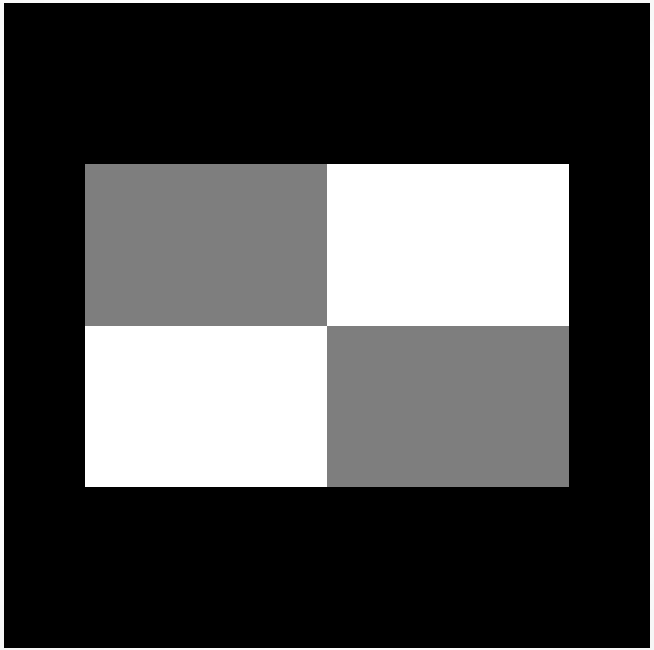
ps6-3-a-3

3a: Haar Features



ps6-3-a-4

3a: Haar Features



ps6-3-a-5

3c: Analysis

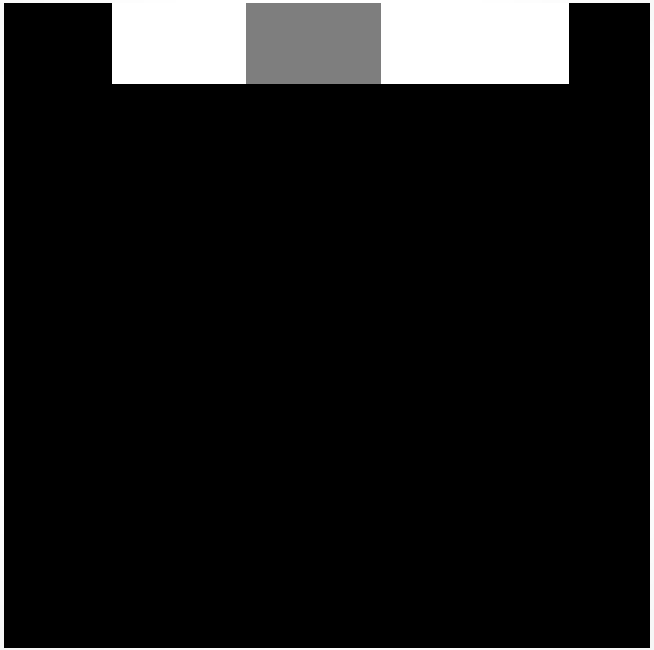
How does working with integral images help with computation time? Give some examples comparing this method and np.sum.

For a feature of size nxm, it would take knm sum operations for a single feature, where k is the number of blocks in the feature. Whlie with integral images, it only takes 4k operations for a feature. Therefore, from a complexity point of view, it is much faster. However, np.sum is much faster than normal sum operations. But with integral images, it is still 15-20% times faster for most of feature types. The operation to train a VJ model in 4A takes 40% lesser time with integral images.

Training time for np.sum with 5 iterations = 371s

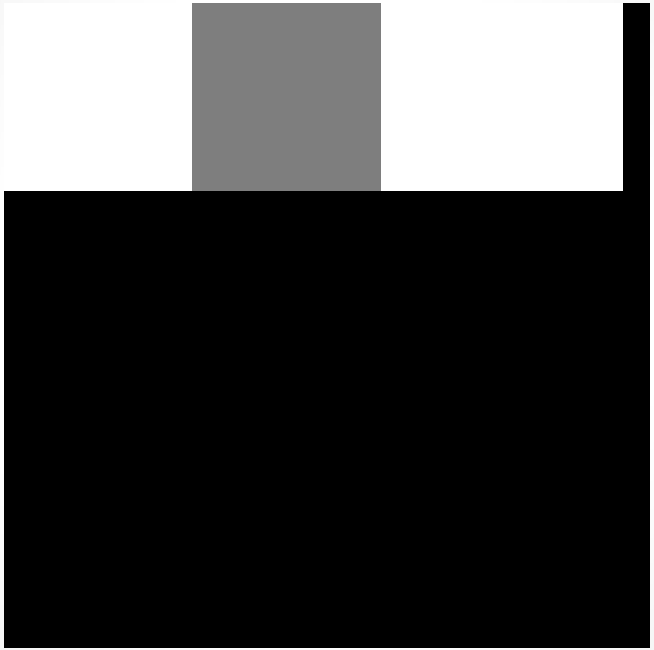
Training time for integral images with 5 iterations = 228s

4b: Viola Jones Features



ps6-4-b-1

4b: Viola Jones Features



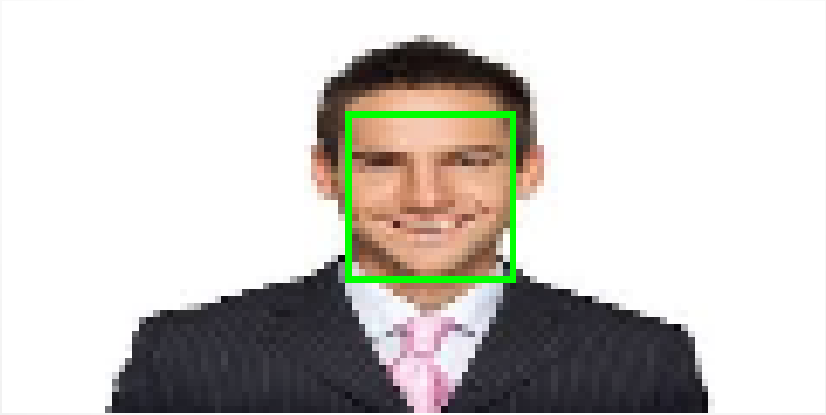
ps6-4-b-2

4b: Analysis

Report the classifier accuracy both the training and test sets with a number of classifiers set to 5. What do the selected Haar features mean? How do they contribute in identifying faces in an image?

Train accuracy 100%
Test accuracy 80%
The selected Haar features correspond to the detection of eye area, with two white blocks summing up the pixel values around two eyes and subtracting the space in between eyes covered by the grey area.

4c: Viola Jones Face Recognition



ps6-4-c-1