Kenny McAvoy
Deliverable 2
Face Recognition

My project relies on python 2.7 and a number of packages that can all be installed through pip. The native ones are os, sys, and string. The ones needing to be installed are numpy, cv2, sklearn, and skimage. I also have a python file of function I will be using that is also imported. In terms of file structure for my program to work, I will upload it to the dropbox file exactly as it should be structured. All that is needed is really needed is for the parent folders to be in the same directory and the main program file and function file to be together. The only necessary command to step through the program is use of the keyboard. This only comes into play when moving through the validation part of the program which is evident through the command terminal. When “Complete“ is printed, the program has ended. Statistics will also be printed in the terminal.

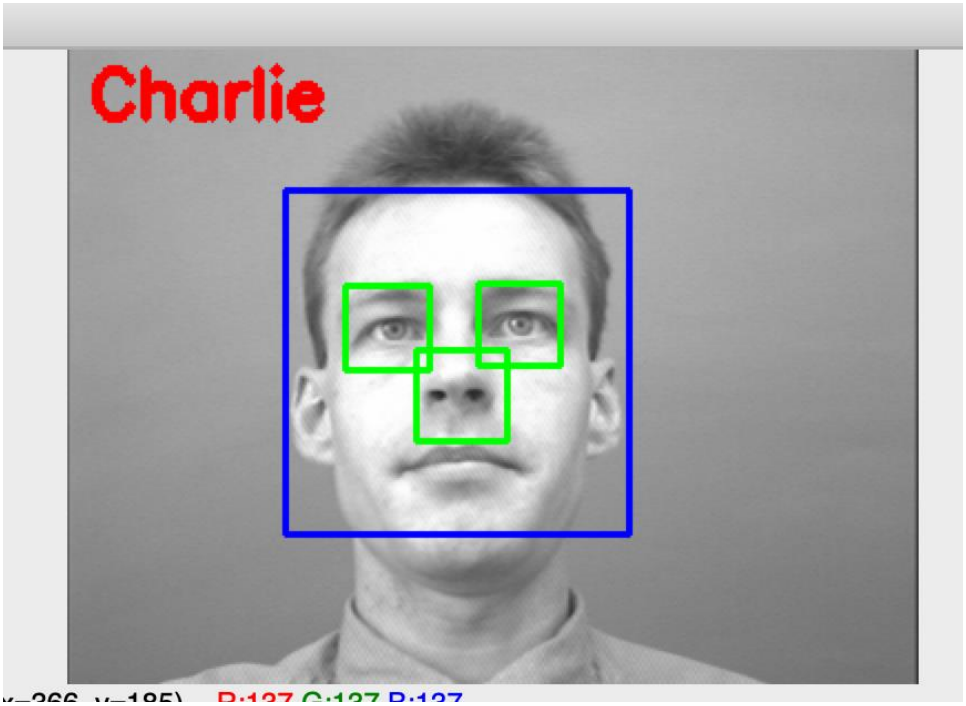
In my project I am currently using the OpenCV built in Haar cascade classifiers to detect both the front of the face and the eyes. I decided on this method to detect the faces of my subjects due to its prevalence and the Haar Violet method being so quick. I find this method works very well for detecting the faces of my subjects. On all of my training and validation subjects the face is well outlined and I have been using this data to then pass on to my feature extractor after first cropping the image to just the outline of the face.

I am using the LBP feature extraction because it works well in my testing and has been able to quickly identify key features and using the uniform bins, relay the important information succinctly without much noise. I then collect these features into two sets or data, one for the identifier of the training set and one for the feature data collected from the LBP histogram, which I have normalized to increase accuracy and reduce noise again. I am using 8 neighboring points and a radius of 1 in my LBP extractor.

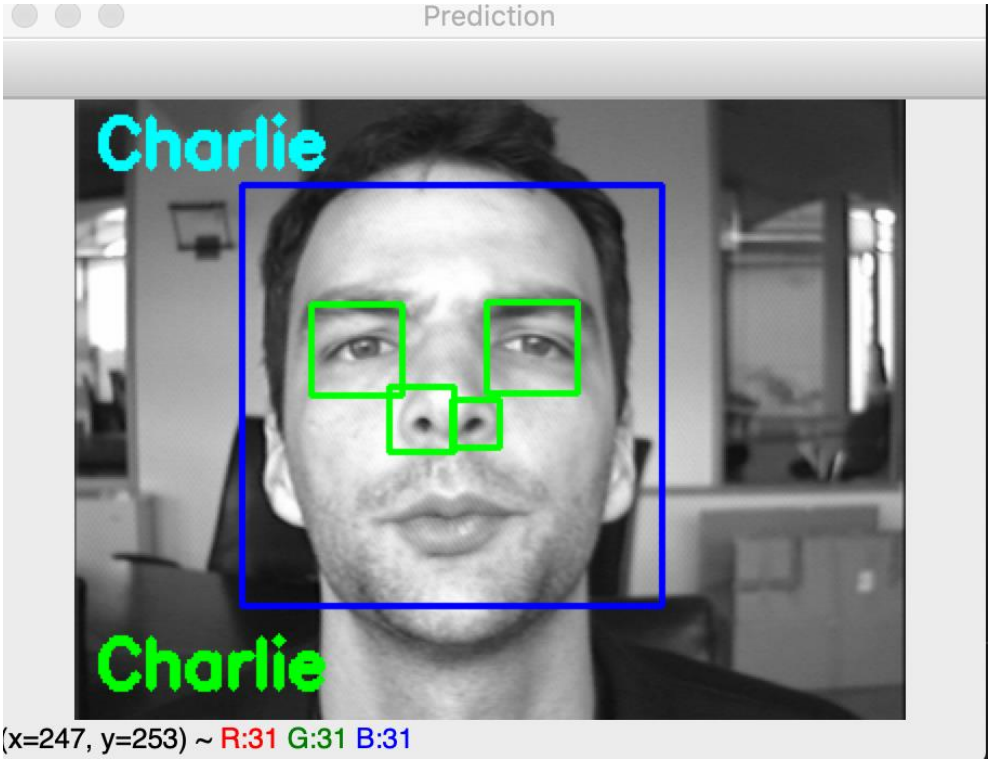
For my classifier, I had trouble deciding, I began with LinearSVC because I am familiar with it and it can quickly distinguish between feature images, while also being scalable to very large datasets. I am migrating over to KNeighbors, however because it can learn from a very small data set ,which I have, and does not make any assumptions other than those it bases on the dataset itself. I have tested a number of different values for the parameters on each classifier and I have implemented the ones that work best. On my training set, the LinearSVC accuracy is about 96% and the KNeighbors accuracy is about 94%. The LinearSVC model is predicting my validation set with an accuracy of about 100%. The KNeighbors model also has an accuracy of about 100%. I believe after I implement more subjects and a larger training set my model will improve, but I am trying to work out which model best suits my project and in the end I added the functionality of both, however I do see better results on the testing set from KNeighbors and I believe it is due to its ability to work well with small datasets. The program will display 4 windows in the stillimage mode. They are all labelled, but for clarity, the Haar Face window shows the Haar face detection for the validation images, the Face window shows the cropped face, the LBP window shows the LBP Features, and the prediction window shows the face and predicted name. The teal text is that of the LinearSVC and the green text is that of the KNeighbors prediction. I have another 20 subjects ready to be implemented and validated, followed by 6 for full testing. I will added a live camera element that will recognize people from

the video stream and allow the user to save these images to their respective training folders if correct.

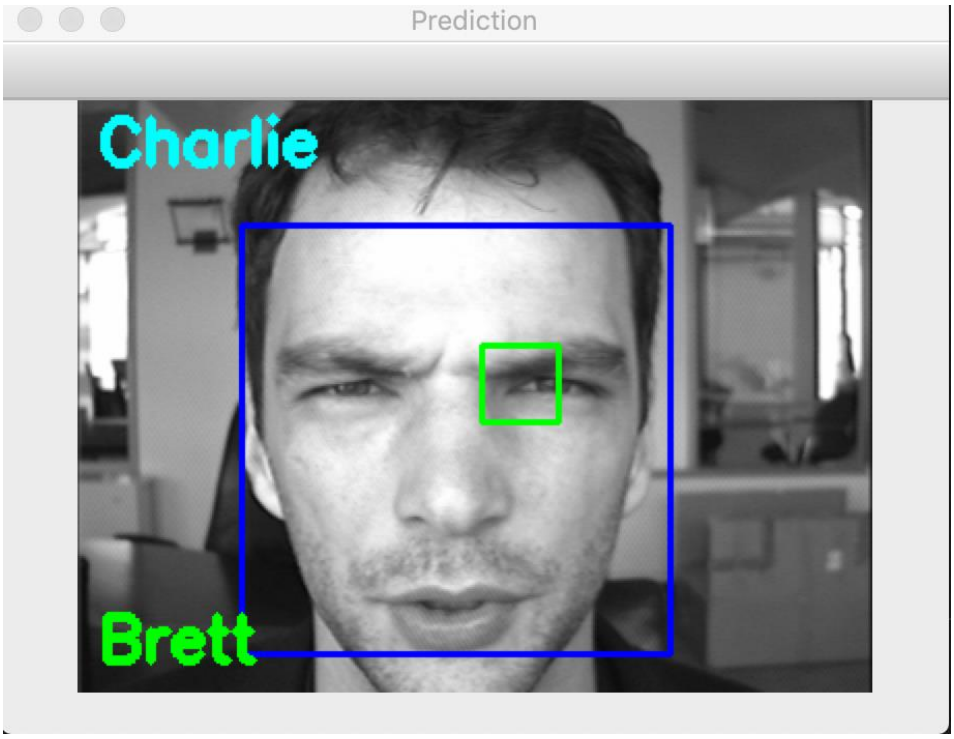
For the final steps of my project I plan on refining the models for better accuracy, deciding on the model that works best, and adding one my feature extraction method to provide the model with more data.



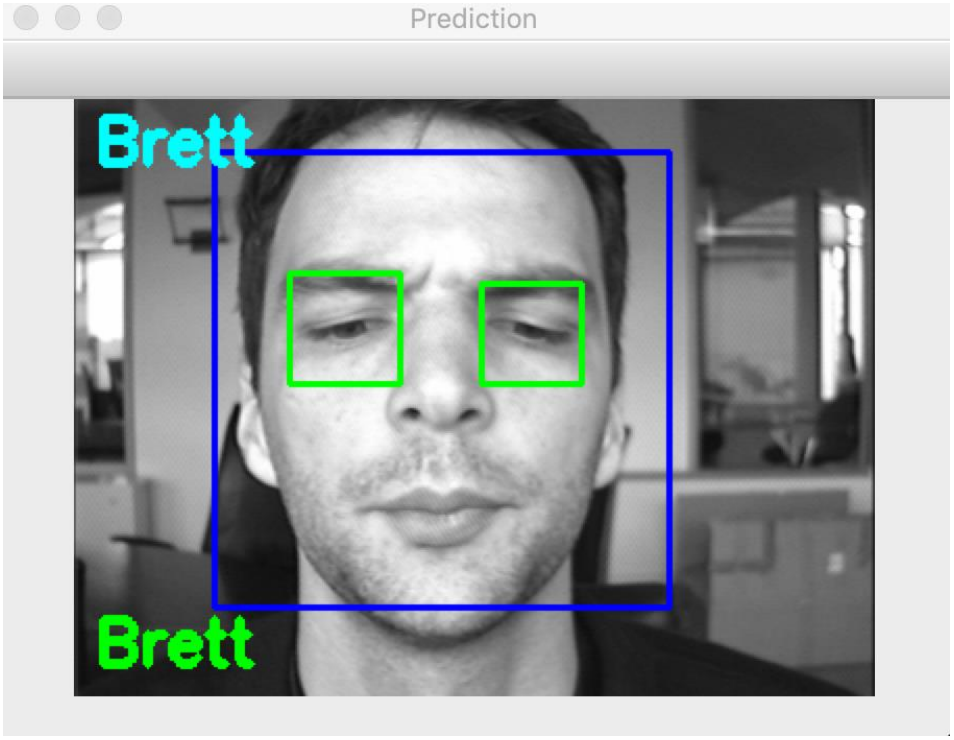
In this image you can see how the face is outlined in a blue box, the “eyes” in green, and the name displayed at the top. I am working to adjust the eye aspect to be more accurate.



This image shows an error in both prediction models. Along with another case of the nostrils being mistaken as eyes.



This picture shows an example of the Haar cascade correctly finding the face, incoorectly finding the eyes, and the LinearSVC models making an incorrect prediction and the KNeighbor model making an accurate prediction.



This picture shows the correct face outline, eye outline, and model predictions. This is ideally what each subject should look like after refinement of the model parameters and addition feature implementations.