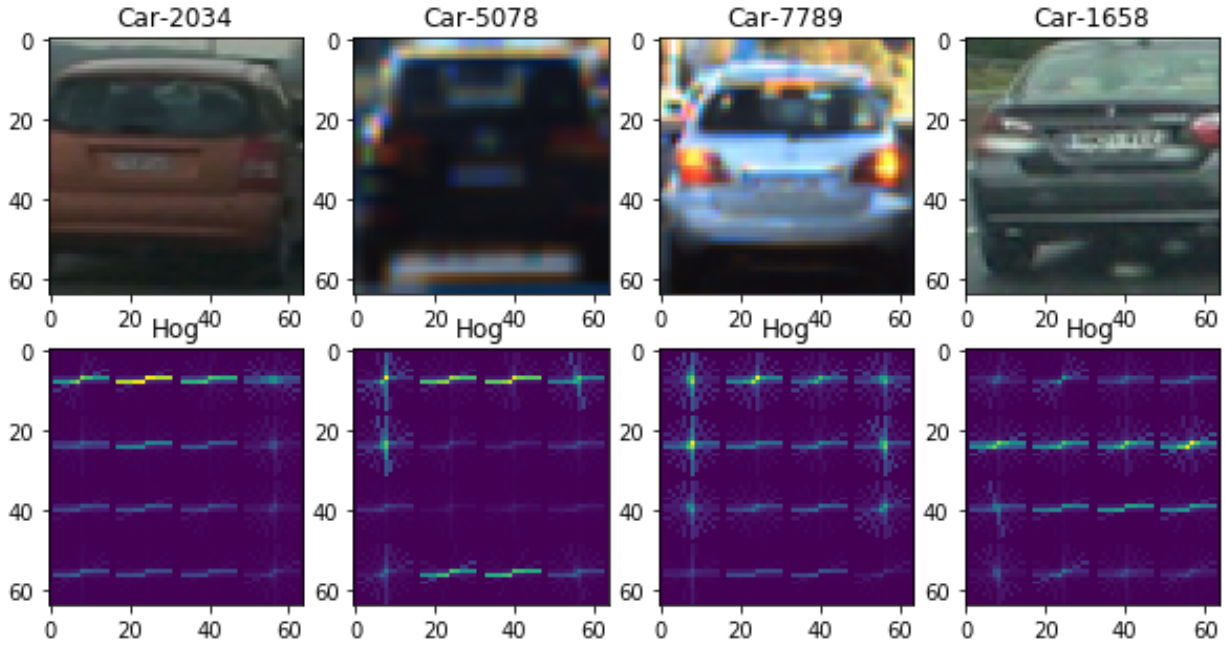
This project started by needing to have images that were both car and non-car. That way the program could learn to determine what was a car on the road. So the first thing I started with was to show examples of car and non-car.

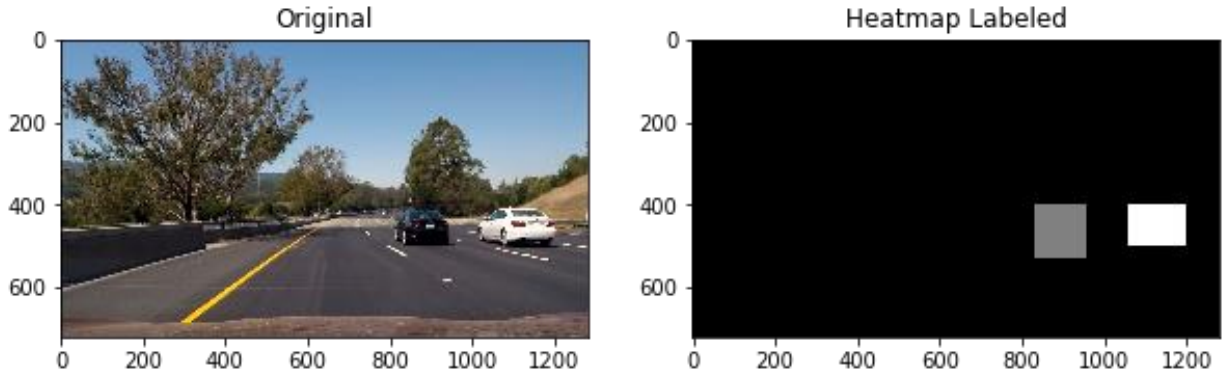
Pictures from the Jupyter notebook. Then the next thing that was needed was to create a histogram of the image for each of the three channels. Then the HOG features could be extracted. The HOG settings was to look at all channels in the YUC color space. This was further controlled by looking at only 2 cells per block and 16 pixels per cell for a total of 32 pixels per block. In the Jupyter notebook the HOG data is shown using the plt.imshow(img_hog).



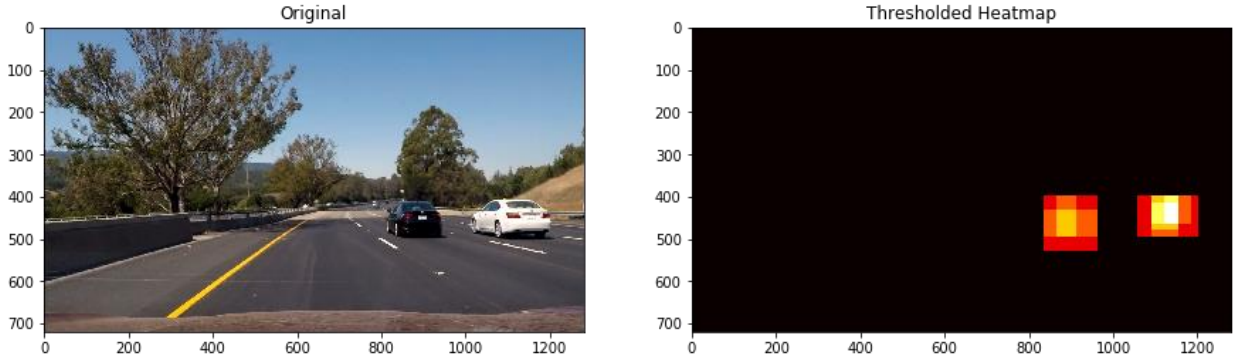
The HOG features of both the car and non-car are then extracted. The classifier chosen was that of the Linear SVC and after training the accuracy of this classifier was 98%. Based on this value there should be a good chance that the program would identify the car most of the time, with only a 2% potential error, next this had to be addressed since 2% when driving could still cause an accident. So the area of where the potential car would be was reduced since you would not be looking for a car in the sky. Here is a an image that has been cropped to prevent the program from looking for a car in the sky.

The next step was to determine where the car was on the road for this a sliding window was used to scan for a car, if one matched what was trained using the SVM classifier the program would identify it with a rectangle/box. Image showing a bounding box around the car.

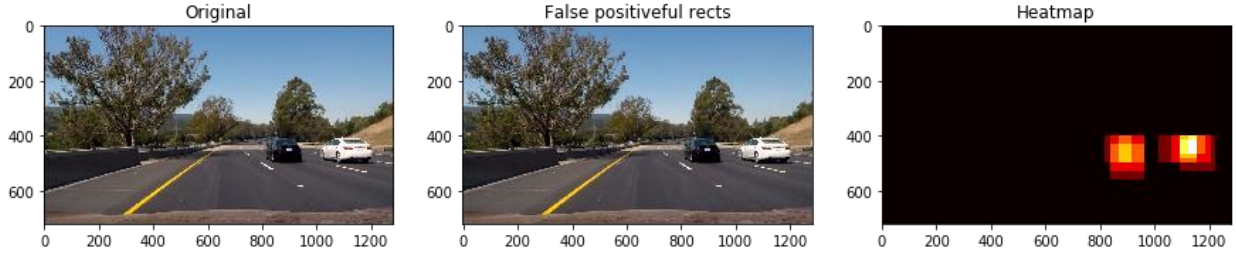
In order to help to eliminate false positives the technique of using a heat map was employed.



Along with the technique of using a threshold values to help to eliminate false positives.



When combined together we get a better representation of where a car maybe.



This technique would show changes as a car was moving so since this could be used to determine where a potential car might be when combined with the sliding window using the HOG features it would almost completely eliminate false positives. A problem that I saw is because area the program is scanning for a car is known to have them it can sometimes very briefly identify a car that is there for a brief second due to changes in the coloring of the road, so I could not eliminate all the false positives but by checking more than just that frame and using multiple windows we can have a high certainty that the car has been determined correctly and virtually eliminate the false positives determined on the road.