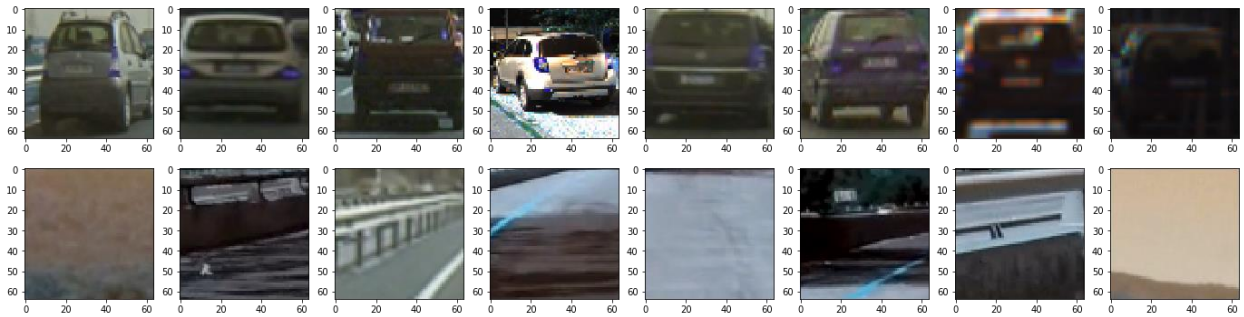
# Vehicle Detection and Tracking

Dear reviewer: I am getting a lot of false positives and I don't know how to reconcile this. Can you please look through my code and see what I am doing wrong? I went through forums for help, and tried many of the suggestions, but I still can't figure it out. I will continuously work to try to figure out what I am doing wrong. But given that I have very little time to submit this final project on an extension schedule, I thought a second pair of expert eyes would only be helpful. Thanks!

1: Data summary and visualization

As always, the first step is the data exploration to understand what dataset we are using. For this project data was provided by Udacity. It included 8,792 images of cars and 8.968 images of non-cars. Image shape was 64x64 RGB image. Figure 1 shows a few of the car and not cars images. Car images have picture of cars from the back or the side from various angles. The lighting seems to differ a lot, with some images look to be with a lot of sun lights while others are images from the darkness.

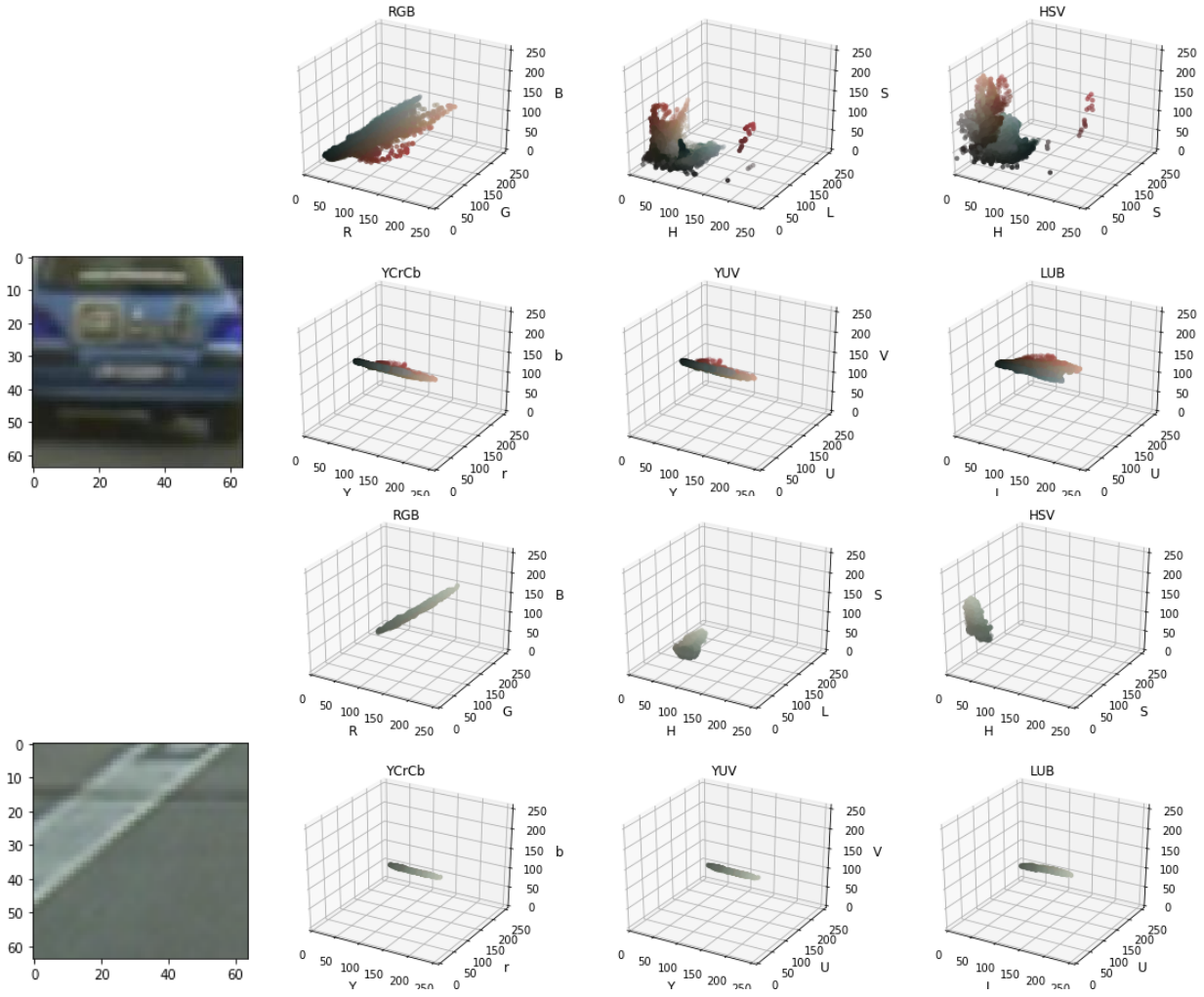
Figure 1: sample images



2: Visualize color spaces

I then visualized cars and non-cars images across different color spaces, including RGB, HLS, HSV, YCrCb, YUV, and LUB. Figure 2 has a car and non-car images along with their corresponding color spaces. One thing I noticed was that YCrCb and YUV were nearly identical in most images. Through iterations, they seem to produce most distinguishiable features. At the end, YCrCb was used to separate car images from non-car images.

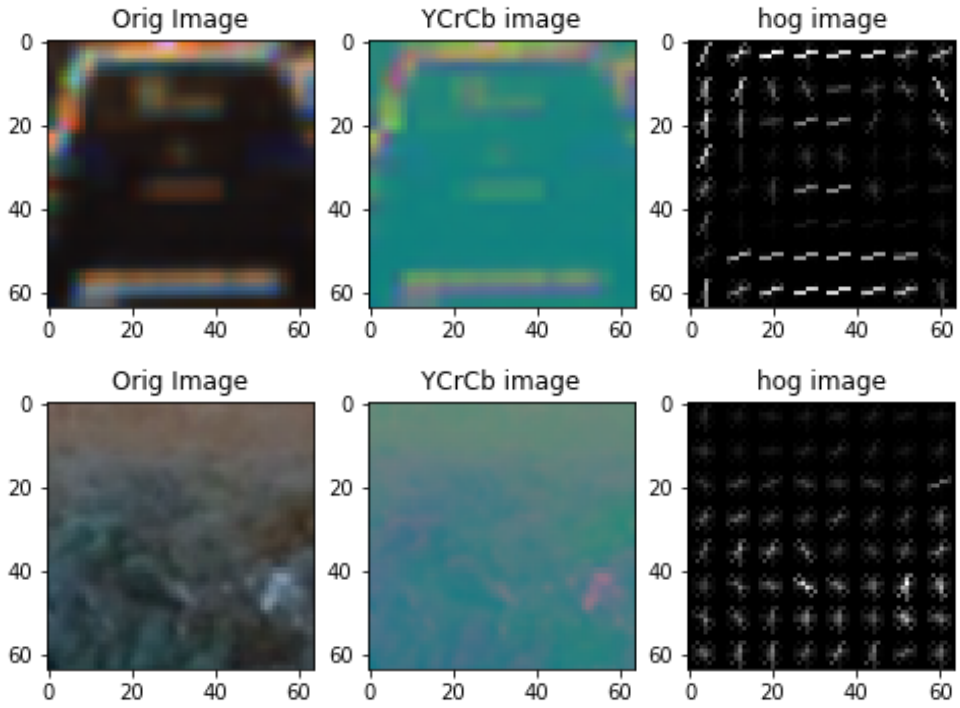
Figure 2: Colorspace for a car and non-car images




3: Visualize HOG features

I then compared the HOG features for car and non-car images as seen in figure 3.

Figure 3: HOG features for car and non-car images



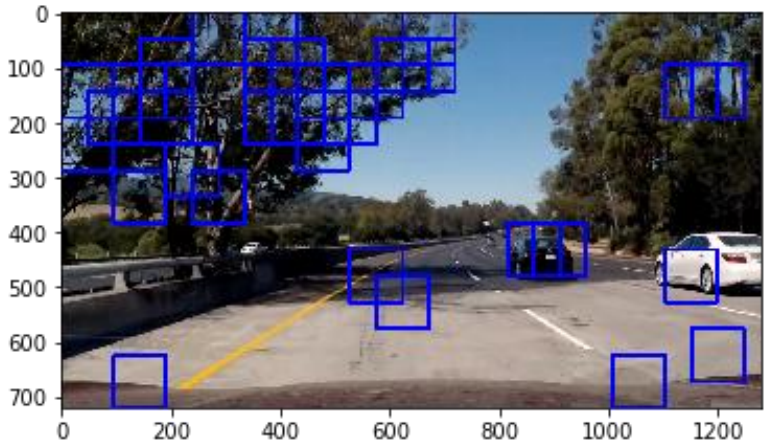

4: Train the data in SVM after combining and normalizing features

Before training the data, I combined the color spatial, color histogram and hog features of YCrCb converted images. This is because the features may have values that vary widely in magnitude, so normalizing try prevent larger value to get a dominant effect. I used the spatial size of 32 and histogram bins of 32. For HOG, I used orientations of 9, pixels per cells of 8 and 2 cells per block. In addition, I split the training and test data into 80/20 split. This resulted in the accuracy of 98.2%.

5: Sliding Window

I then used the sliding windows in the test images and tried to identify if the car is in that window. The result seemed to show a lot of false positives as seen in figure 4.

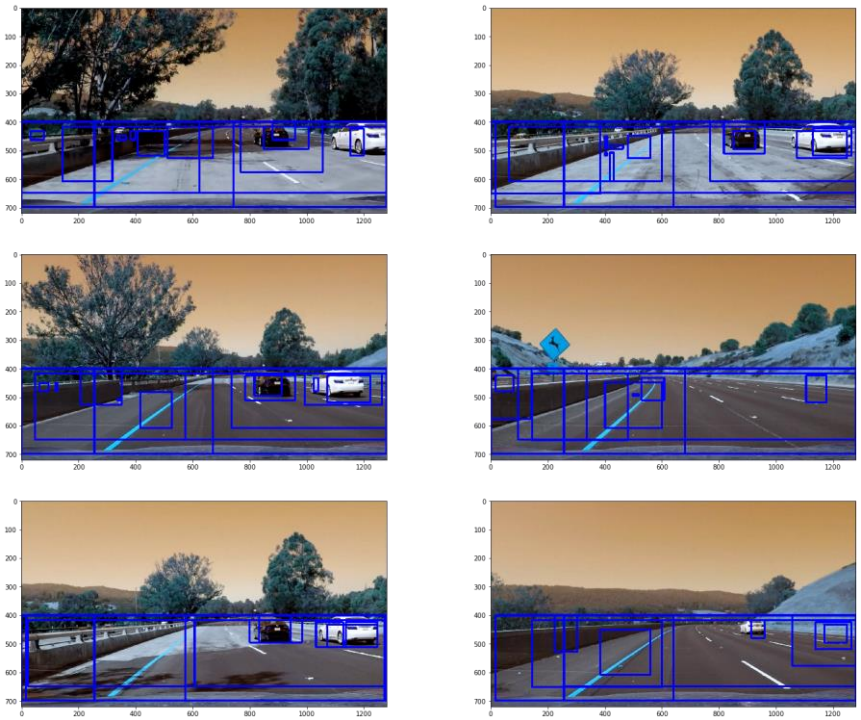
Figure 4:  Identifying cars in test images using sliding windows




5: Use various scale, limit region search and heatmap

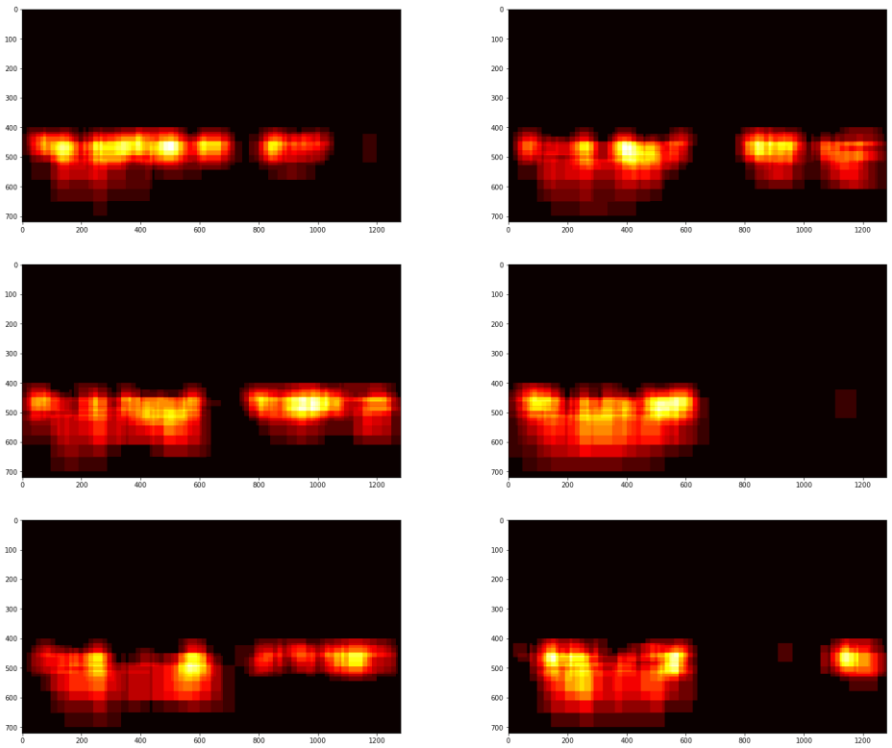
I then used various scaling on the sliding windows.  I also limited the search region because car images would only appear on the bottom half.  The result still looked bad as seen in Figure 5.

Figure 5: Using various scaling factor on sliding window and limiting search region



As a result, heatmap did not return good result.

Figure 6: heatmap



6: Use pipeline to run on a video

Similarly, the video did not produce a satisfactory output.