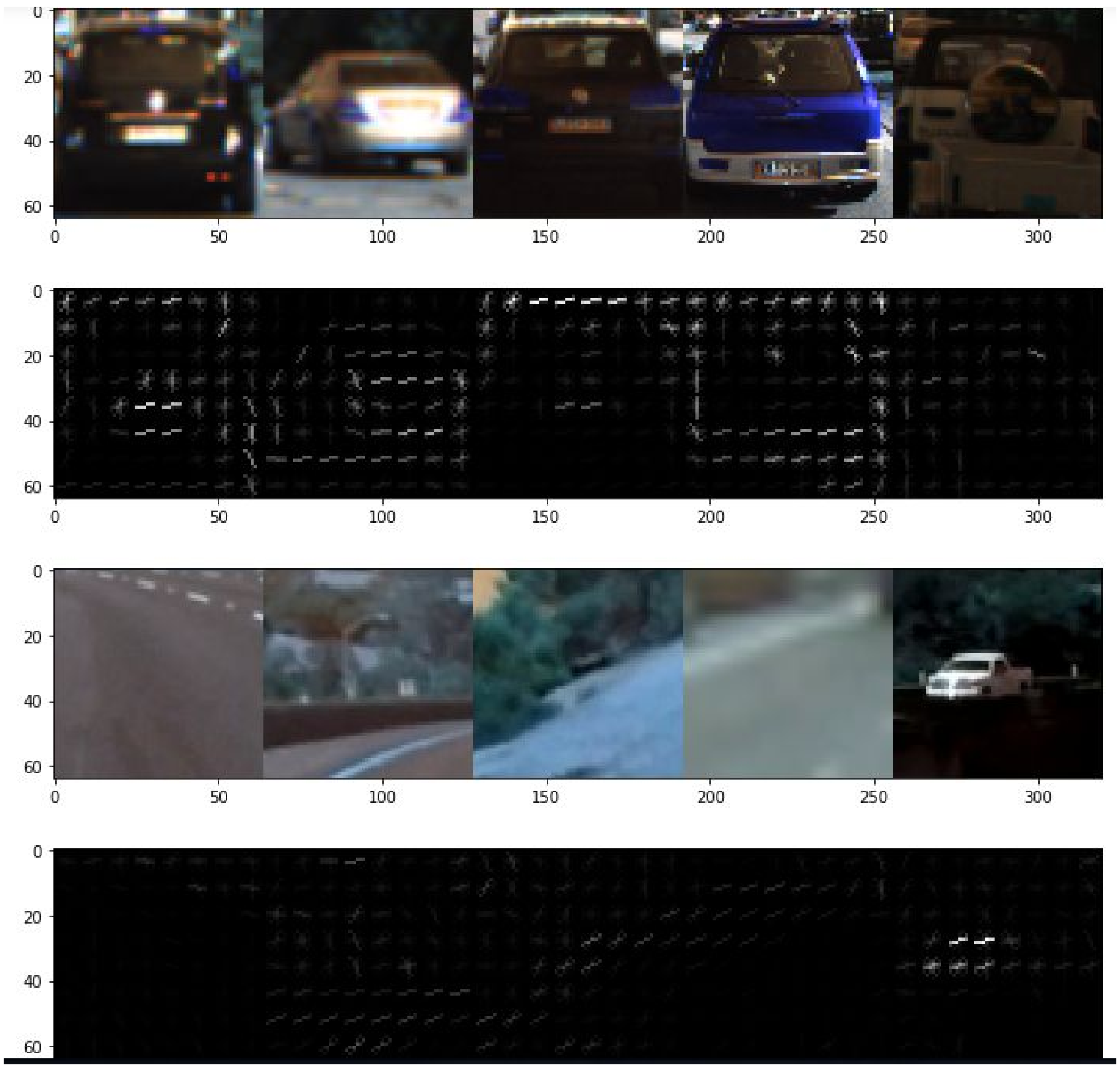
###Histogram of Oriented Gradients (HOG)

####1. Explain how (and identify where in your code) you extracted HOG features from the training images.

I extracted the vehicle and non-vehicle zip files provided by udacity. Here’s a random sample of a car and a non car image as shown in cell 4 of the ipython notebook

I did some random sampling of car and not car images and got the HOG features for each to test whether HOG features are actually representative of the images. Here’s a sample that I had drawn



How did I settle for the HOG parameters - Fundamentally educated guesses and some trial and error.

####3. Describe how (and identify where in your code) you trained a classifier using your selected HOG features (and color features if you used them).

I trained a classifier in the classify_car method. I used a combination of HOG features,  spatial bins and the color histogram features

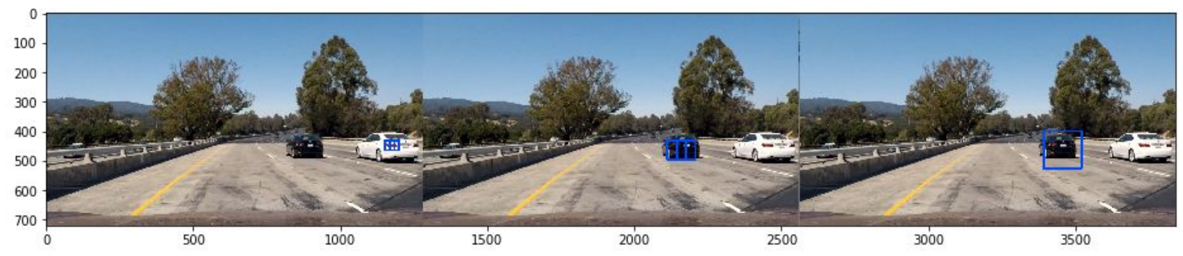
I attained an accuracy of 99% on the training set.

###Sliding Window Search

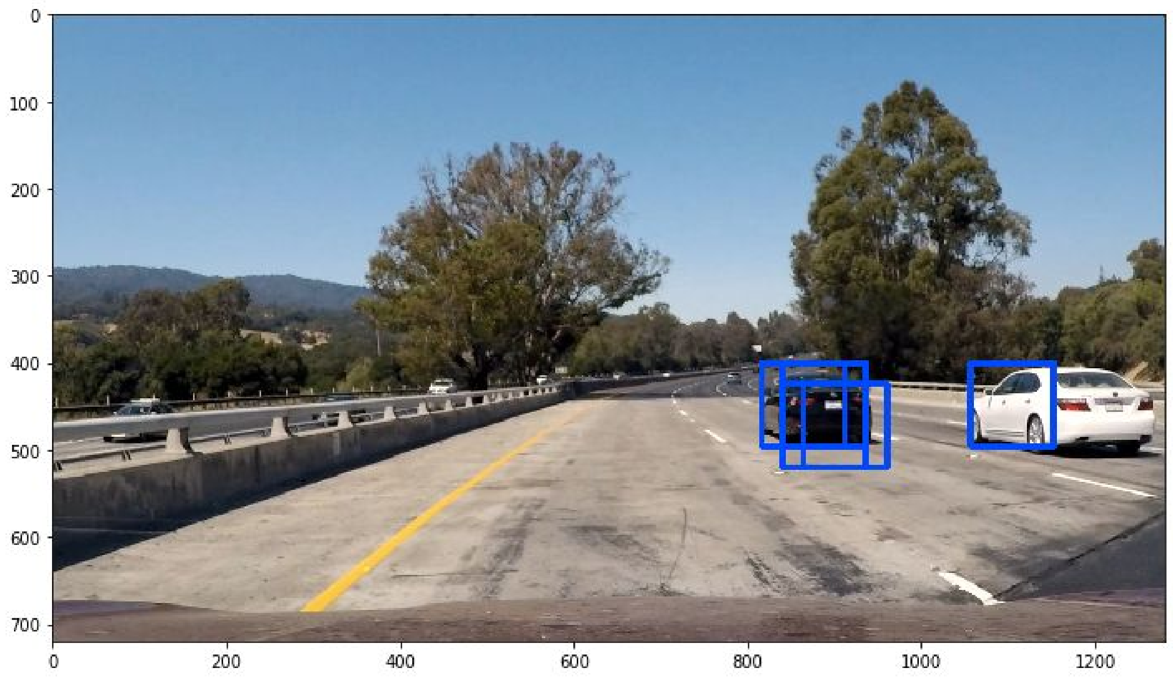
####1. Describe how (and identify where in your code) you implemented a sliding window search. How did you decide what scales to search and how much to overlap windows?

I have implemented the sliding window search in the slide_window method. I used a simple loop using 32, 64 and 128 size windows in the find_cars_no_subsample method. Here are the outputs.

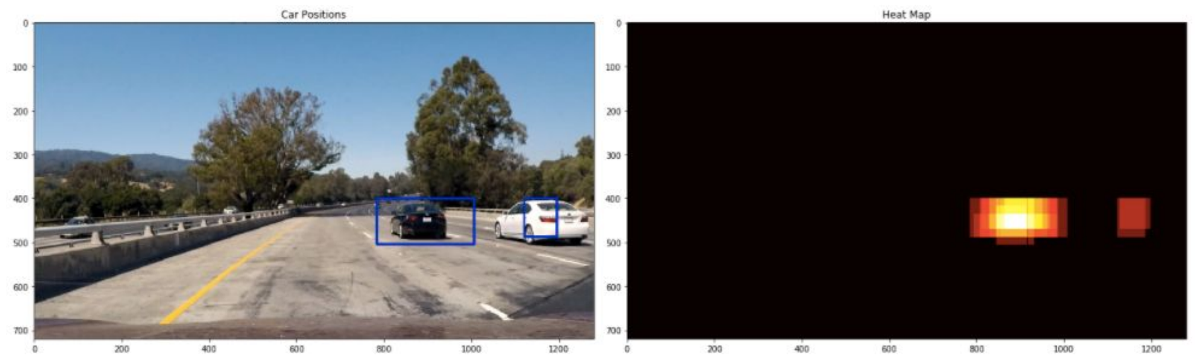
```
<matplotlib.image.AxesImage at 0x7f9a17489e48>
```



Ultimately I used Hog sub sampling and got the following result



I then used the heat map function to eliminate the false positives. The resulting heatmap and the bounding boxes are given below.



## Video Implementation

####1. Provide a link to your final video output. Your pipeline should perform reasonably well on the entire project video (somewhat wobbly or unstable bounding boxes are ok as long as you are identifying the vehicles most of the time with minimal false positives.)

####2. Describe how (and identify where in your code) you implemented some kind of filter for false positives and some method for combining overlapping bounding boxes.

This logic is added in the method process_image. I experimented with a smoothing threshold of 4-6. I have detected the false positives using the get_heat_map function. For the video I have used the output of the get_heat_map function and just averaged the bounding boxes.

I found a way to embed the heatmap image inside the main image so that they both can be seen side by side as the video plays.

The video output file is called out_smooth_4.mp4 and is in the src as well as the project output directories.

####3 Improvements:

I can clearly see the limitations of the SVM classifier for object detection. While it can detect cars, the precision and recall both are questionable. Precision is still high but recall is nowhere close. This definitely won’t fly on real roads.

The HOG and color features are also serious limitations. What is needed is a supervised classifier based on learning features rather than the engineered features which the SVM uses. Perhaps deep neural nets combined with some form of online learning might help.