Writeup

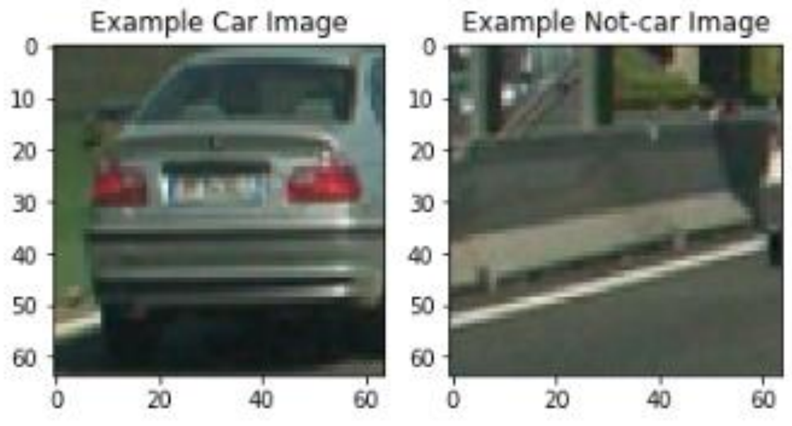
Working code is in working_final.ipynb

Working video is result_c_10_h_2_final.mp4 **(SVM c parameter was 10 and heatmap threshold is 2)**

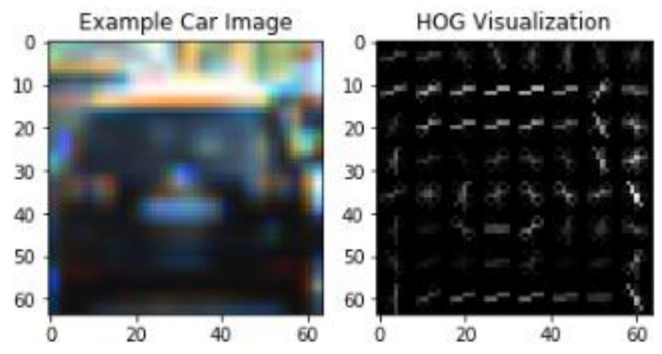
Histogram of Oriented Gradients (HOG)

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

I started by reading in all the vehicle and non-vehicle images. Here is an example of one of each of the vehicle and non-vehicle classes:



I then explored different color spaces and different skimage.hog() parameters (orientations, pixels_per_cell, and cells_per_block). I grabbed random images from each of the two classes and displayed them to get a feel for what the skimage.hog() output looks like.
Here is an example using the YCrCb color space and HOG parameters of orientations=13, pixels_per_cell=(16, 16) and cells_per_block=(2, 2):



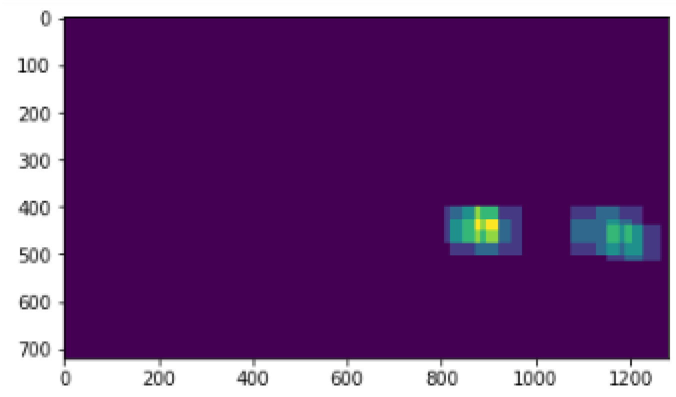
I tried various combinations with orientations as 9 and 11 and 13. 13 worked the best for me when cells_per_pixel parameter in my code was made 16. I had initially tried using only 'Y' channel too but results were not good so I used 'ALL' ultimately.

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 Linear SVM in code cell 21 with C=10. It took about 30 seconds to train and was giving a test accuracy of 99.13%

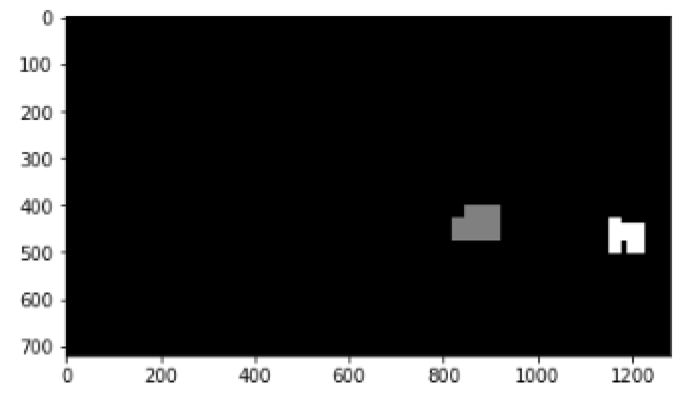
Sliding Window Search

I used Hog subsampling plus spatially binned color and histograms of color in the feature vector, which provided a nice result as given in Udacity course. I used 4 scales which were 0.8,1.2,1.6 and 2. The boxes were then collected and passed through heat map and deque averaging implementation over the last 6 frames. I recorded the positions of positive detections in each frame of the video. From the positive detections I created a heatmap and then thresholded that map to identify vehicle positions. I then used scipy.ndimage.measurements.label() to identify individual blobs in the heatmap. I then assumed each blob corresponded to a vehicle. I constructed bounding boxes to cover the area of each blob detected.

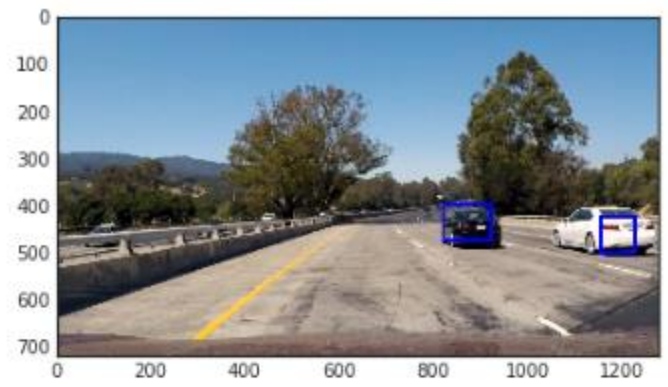


An example of heat map for the first test_image frame.

Here is the output of `scipy.ndimage.measurements.label()` on the integrated heatmap



Here the resulting bounding boxes are drawn onto the first frame in the series:



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.)

https://github.com/gireek/Carnd-object-detection-P5/blob/master/result_c_10_h_2_final.mp4

### Discussion

#### 1. Briefly discuss any problems / issues you faced in your implementation of this project. Where will your pipeline likely fail? What could you do to make it more robust?

Getting the right combination of parameters to make the code work on the video pipeline was a bit challenging .I observed the parameters of HOG and spatial binning are very important for this project and I must have tried more than 20 set of parameters as I was training 3 different sets of parameters at 3 different machines at the same time.

The cell_per_pixel parameter made to 16 from 8 speeds up the video making process a lot as the number of features are reduced greatly. Two more things are the sliding window technique and removing false positives for which I tweaked different scales and heat threshold at the final step. My future pipeline would be to implement the same with deep learning object detection CNNs mostly which is more robust.

It's has quite a cool result at the end and this was by far the most rewarding and exciting project till now. Excited for term 2!! ☺