Vehicle Detection

The goals / steps of this project are the following:

- Perform a Histogram of Oriented Gradients (HOG) feature extraction on a labeled training set of images and train a classifier Linear SVM classifier
- Optionally, you can also apply a color transform and append binned color features, as well as histograms of color, to your HOG feature vector.
- Note: for those first two steps don't forget to normalize your features and randomize a selection for training and testing.
- Implement a sliding-window technique and use your trained classifier to search for vehicles in images.
- Run your pipeline on a video stream (start with the test_video.mp4 and later implement on full project_video.mp4) and create a heat map of recurring detections frame by frame to reject outliers and follow detected vehicles.
- Estimate a bounding box for vehicles detected.

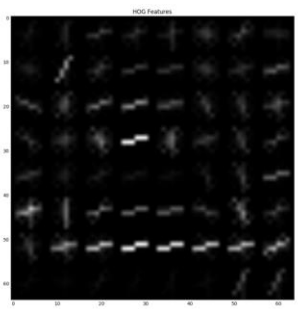
Reading images

If the images was ".jpg" then a scaler was used to convert the scale of image to (0-1), as the png test images used for training have range of 0-1 when read through mpimg.imread.

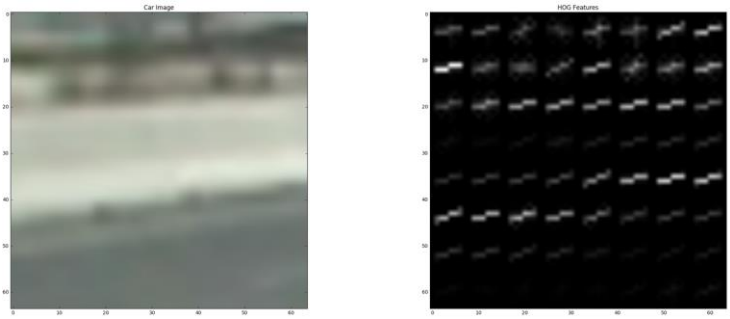
Histogram of Oriented Gradients (HOG) feature extraction

For HOG extraction, different combination of parameters was tested out. But finally the number of orientations of gradient was set as 9. The pix per cell was 8 and cell per block was set as 2.

The below image displays the result of HOG feature extraction for vehicle image:



The below image displays the result of HOG feature extraction for non-vehicle image:



HOG + Color Histogram +Spatial Features

A SVM classifier based on a combination of HOG features, color histogram and spatial features were used to determine features in the vehicle and non-vehicle images. The code for the same could be found in Cell 4.

The YCrCb and RGB color models were tested, but YCrCb color space was used as it gave better results.

Normalizing, Training and testing Accuracy

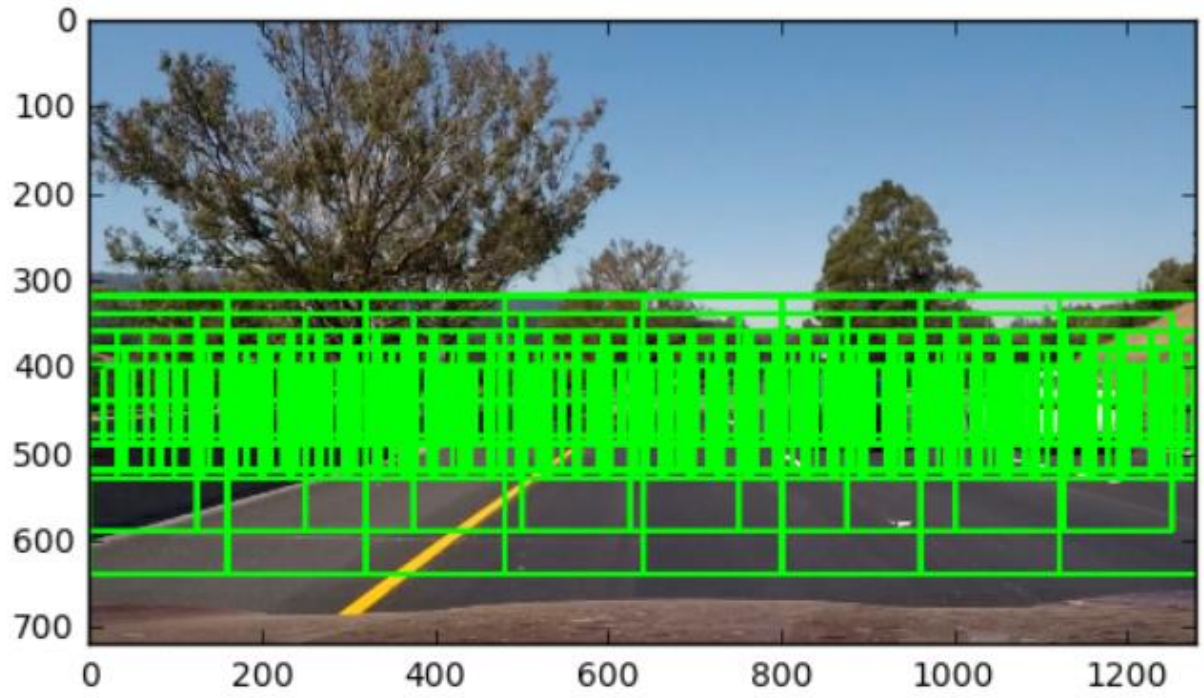
The data for testing and training Linear SVM was taken from the vehicle and non-vehicle dataset. The dataset was normalized using StandardScaler and split randomly so that 20% of the dataset was kept aside for testing. The code for the same could be found in the 10^{th} cell.

The training was done using Linear SVC and the accuracy achieved was 99.22%.

Sliding Window Selection

The sliding window selection was decided by the expected size of vehicles in a particular area. As the vehicles are expected only in the lower half of the image, the sliding windows were focused mainly in this region. The smallest windows aimed at detecting the smallest vehicles were at the top of the lower half. Different sizes of windows were used for vehicles with window size increasing to account for greater vehicles and taking more space in the image.

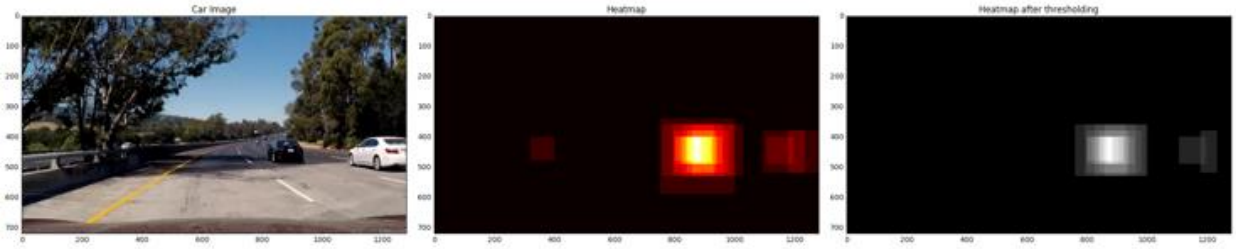
The image below shows the sliding window coverage:

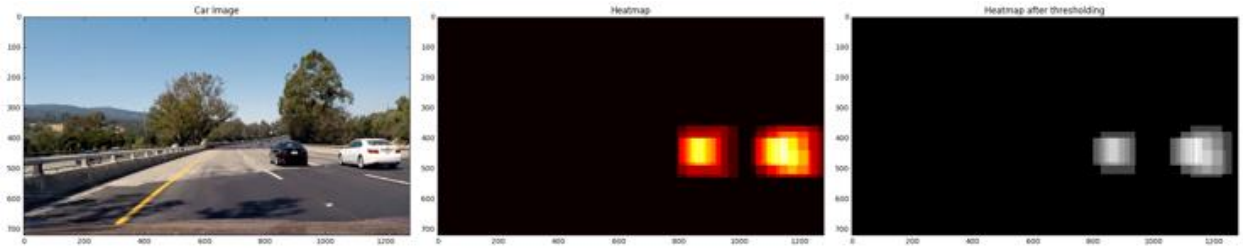


Removing False Positives

The heat methodology was used to remove misdetections. A threshold was applied to the heatmap for the same. The code for it can be found in cell 11.

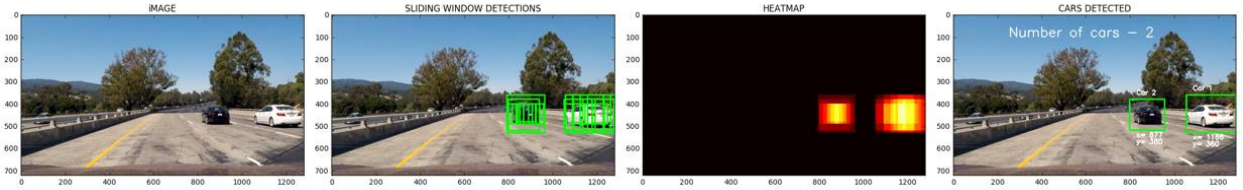
Result:

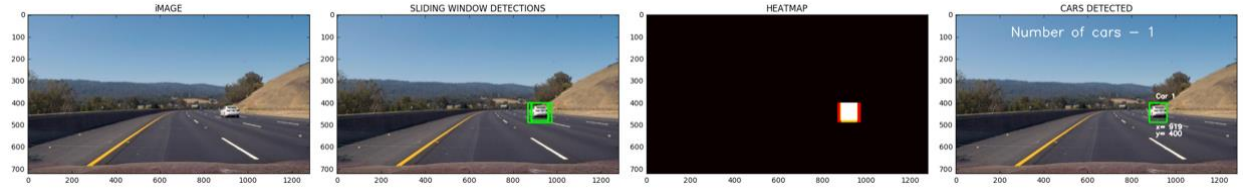


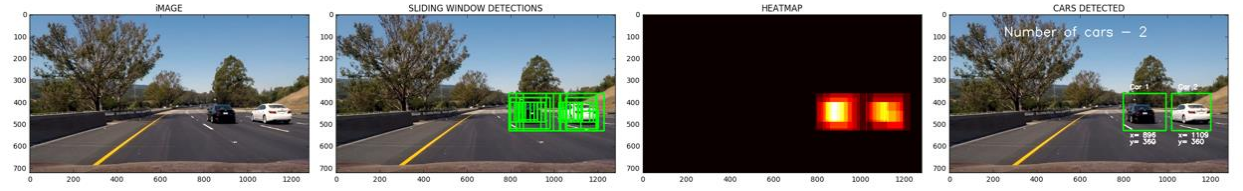


IMPLEMENTATION:

Below are the results of the above implementations on some sample test images. The images show the steps followed in order.

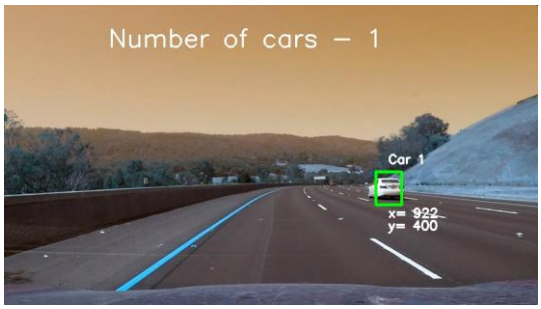


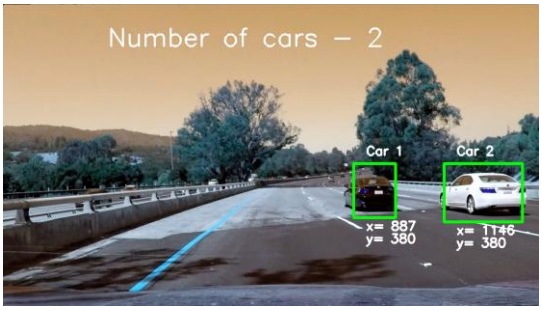




Final Result

The pipeline was run on the test images as well as the video. Here are the results of running the pipeline on test images:





The video displaying the results can be seen on the link below:

https://www.youtube.com/watch?v=oaQOj3ykdUs

Discussion

In my approach to this project, I trained a linear SVM classifier on dataset provided by Udacity. The data set consisted of Vehicle and non-vehicle png images. The images were read, normalized and the dataset was split into training and testing dataset. The features (HOG+ Color Histogram+Spatial) are extracted and fed into the classifier. The linear SVM classifier is trained and an accuracy 0f 99.22% is achieved.

Then a vehicle detection pipeline (code- cell 12) is created which takes in image and returns image displaying boxes around cars detected. The image is first scaled to range of (0-1). After this the image is searched in the sliding windows areas which were decided earlier for the vehicles using the trained classifier. The boxes

which detect vehicles are passed into a heatmap function, which increases the intensity of the intensity of pixels in the bounding box by 1 on a blank image. This heatmap is thresholded to remove false detections. The thresholded heatmap was labeled and it gave us the bounding areas of the vehicles detected.

This algorithm worked on test images and the test_video perfectly. The algorithm was also tested on the project_video. This algorithm falls in certain areas where the sliding window is not properly decided. Adding more search windows would improve the algorithm.