Udacity – Self Driving car Engineer Program

Vehicle Detection and Tracking Project

Steps of Project:

1- Import modules
2- Load dataset
3- Compute color histogram features
4- Compute binned color features
5- HOG features and visualization
6- Explore characteristics of the dataset
7- Extract features from (spatial binned color , color histogram and Histogram of gradient) and combined them.
8- Classify (LinearSVC)
9- Sliding Window Search
10- search windows and Find car (extract features using hog sub-sampling and make predictions)
11- put all process in one (pipline) function
12- Process Video

1- Import modules

Import all modules that we will need through project to not miss some modules
Code cell : 1

2- Load dataset

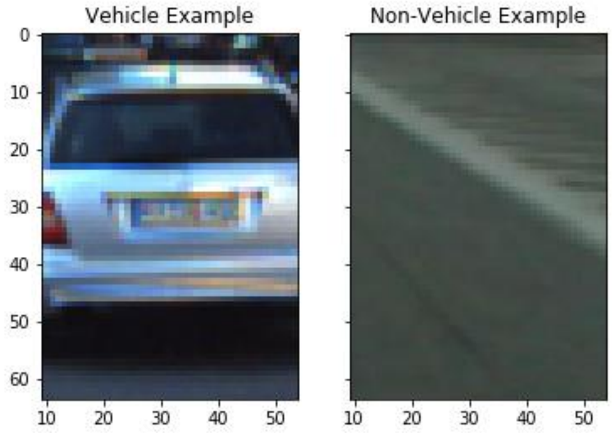
I changed all .png files to .jpg (using powershell on windows 10) then used ‘glob’ to load images from all folders:

```
cars = glob.glob('./data/vehicles/*/*.jpg')
notcars = glob.glob('./data/non-vehicles/*/*.jpg')
```

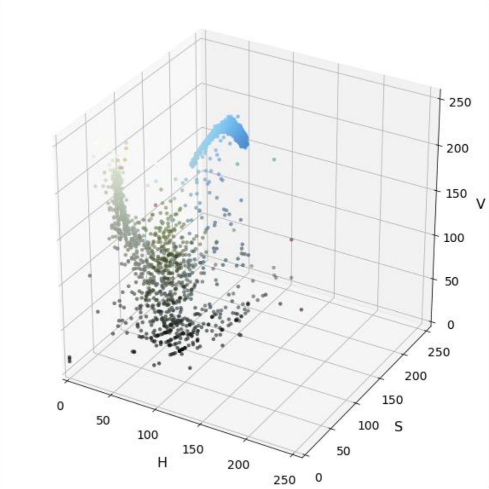
Total vehicles: 8792, Total non-vehicles: 8968

My pc can’t run with this number of feature so I reduced them and removed GTI folder to get:
Total vehicles: 5966, Total non-vehicles: 5068

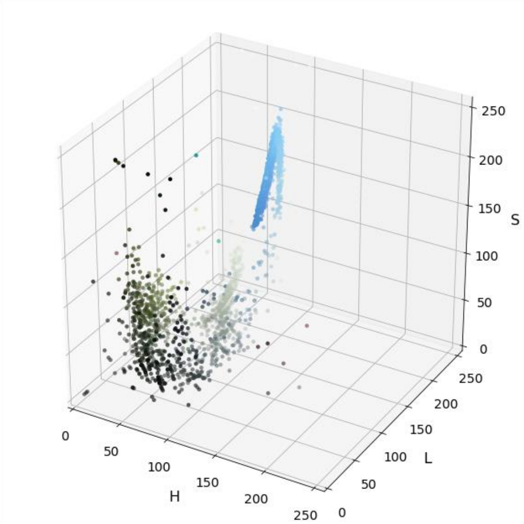
Code cell: 2

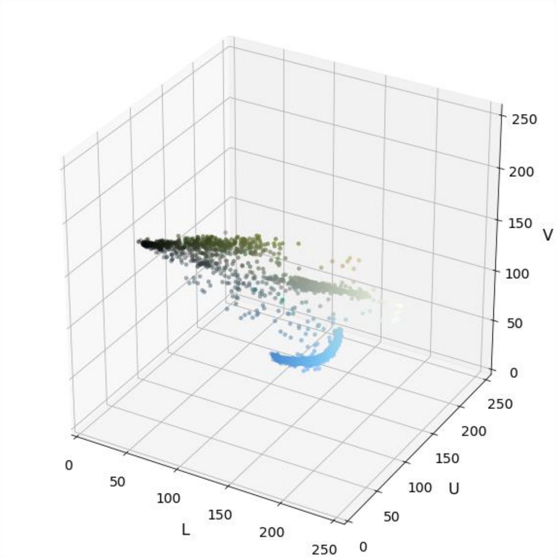


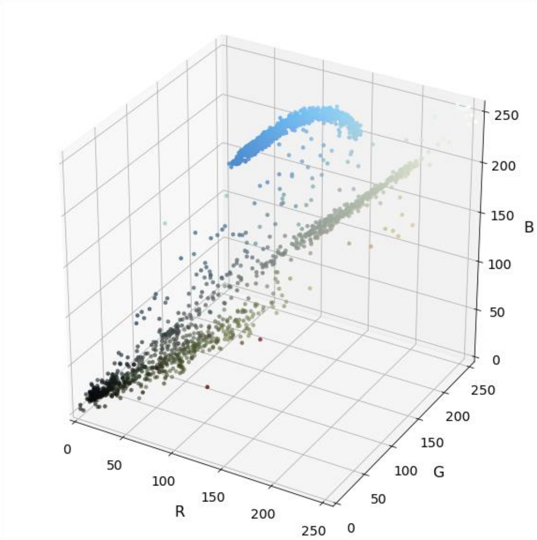
3- Compute color histogram features

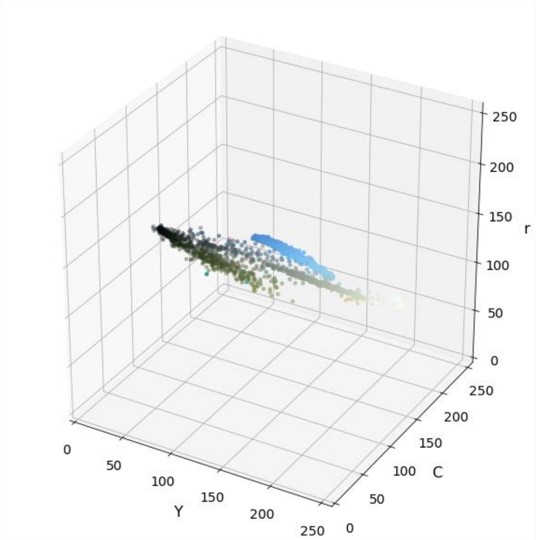


I plot some color spaces in 3d using ‘3dplot.py’

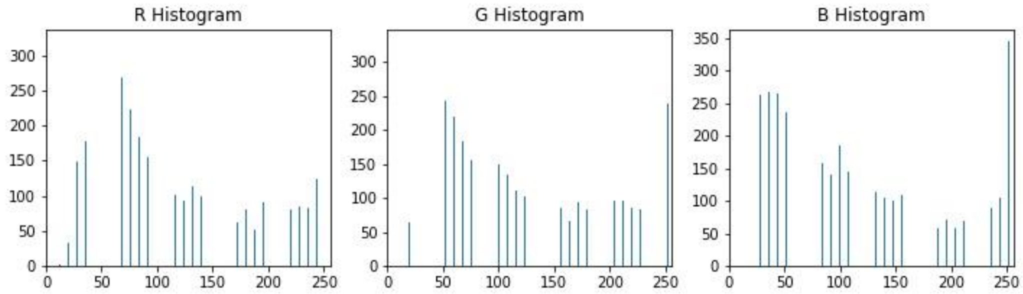






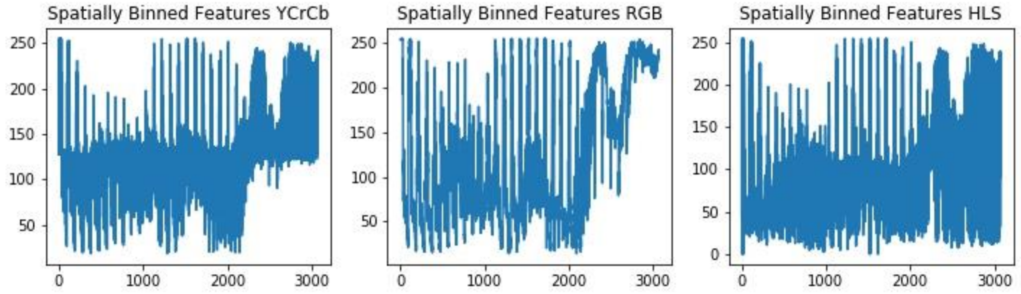


Then I compute color histogram feature and plot an example :
Code cell: 4



4- Compute binned color features

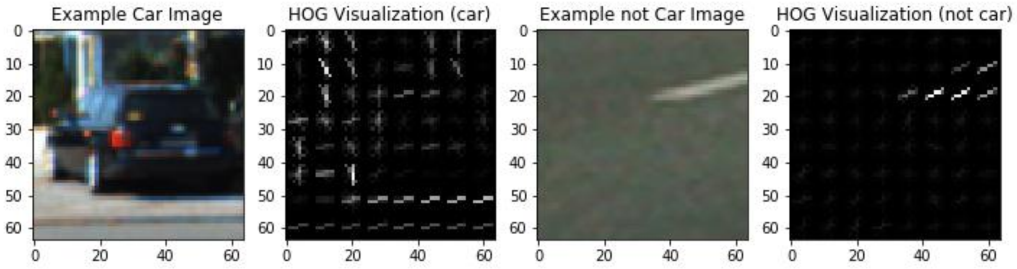
Define a function to compute binned color features and plot an example for YCrCb , RGB and HLS color space:
Code cell: 5



5- HOG features and visualization

Define a function to return HOG features and visualization, The HOG visualization is not actually the feature vector, but rather, a representation that shows the dominant gradient direction within each cell with brightness corresponding to the strength of gradients in that cell.

Code cell: 6



6- Explore characteristics of the dataset

Data set contatin: 5966 cars samples and 5068 non-cars samples
of size: (64, 64, 3) and data type: uint8
code cell: 7

Rubric Point: Explain how (and identify where in your code) you extracted HOG features from the training images. Explain how you settled on your final choice of HOG parameters.

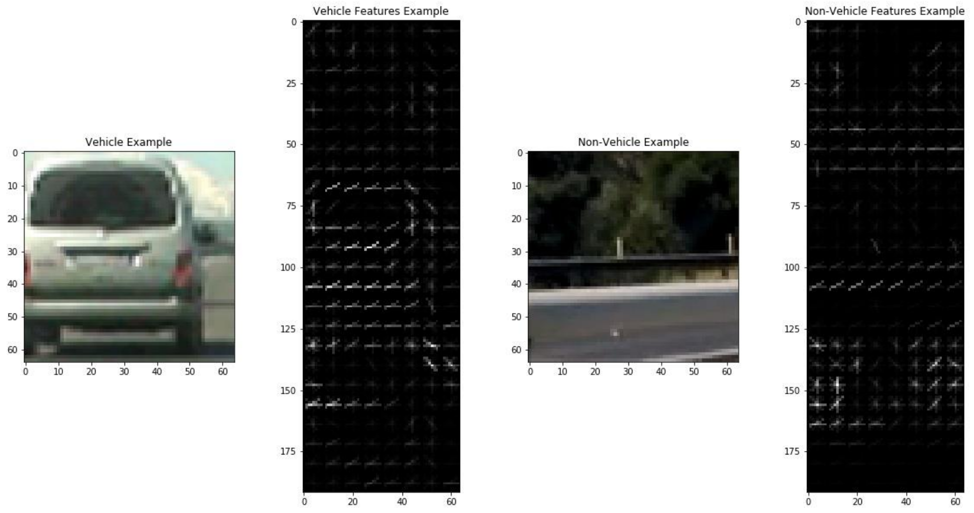
7- Extract features from (spatial binned color , color histogram and Histogram of gradient) and combined them

Extract HOG features, while also converting images to another color space (HLS used) ,and make bersion from previous function that can be included/called by extraction function.
Code cell: 8

After few tuning I find best result on:

```
#cspace#
color_space = 'HLS'  # tried  RGB, HSV, YCrCb but get better results with HLS
orient = 8  # HOG orientations
pix_per_cell = 8  # HOG pixels per cell
cell_per_block = 2  # HOG cells per block
hog_channel = 'ALL'  # Can be 0, 1, 2, or "ALL"
spatial_size = (32, 32)  # Spatial binning dimensions
hist_bins = 32  # Number of histogram bins
```

And then make version from extract function can be able to visualize results on a single image (code cell: 9)



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

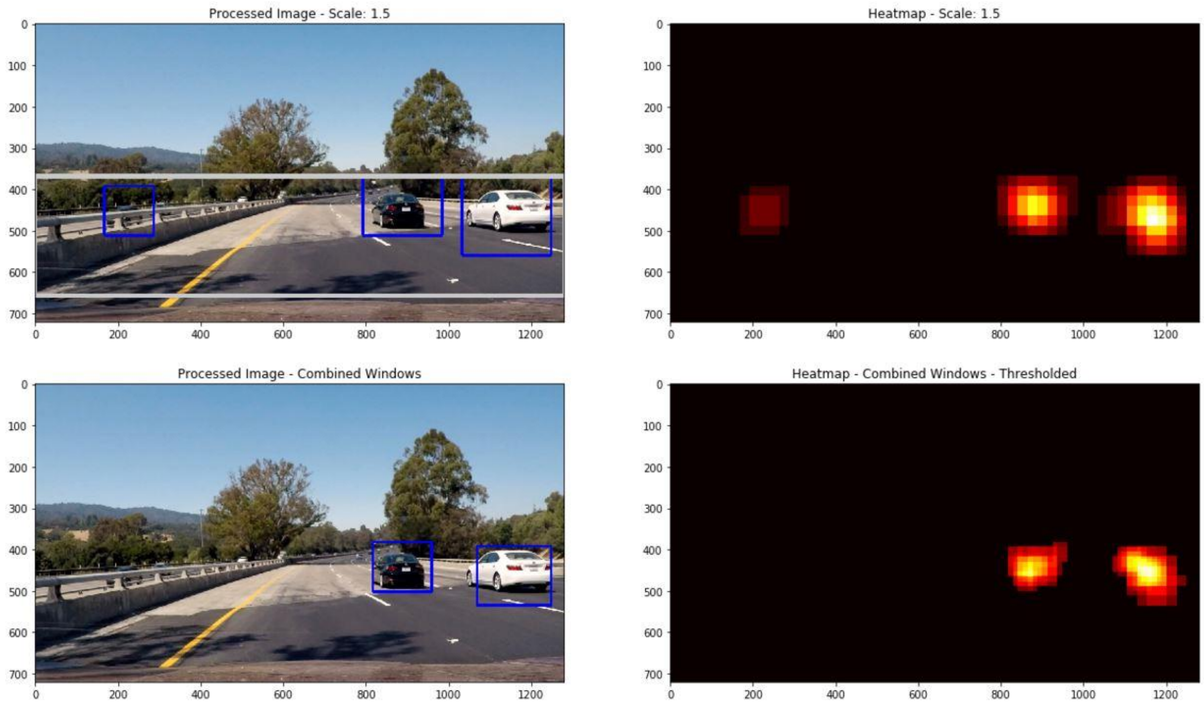
8- Classify (LinearSVC)

Define the labels vector ’y’ then Split up data into randomized training and test sets 20% for test set. Used lindear SVC

Results:

```
Labeling the feature set...
Feature set labeled.
The history saving thread hit an unexpected error (OperationalError('disk I/O error',)).History will not be written to the database.
Using: 8 orientations 8 pixels per cell and 2 cells per block, with features extracted using HLS color space
Feature vector length: 7872
Starting to train the SVC...
588.86 Seconds to train SVC...
Test Accuracy of SVC =  0.9995
```

Code cell: 14

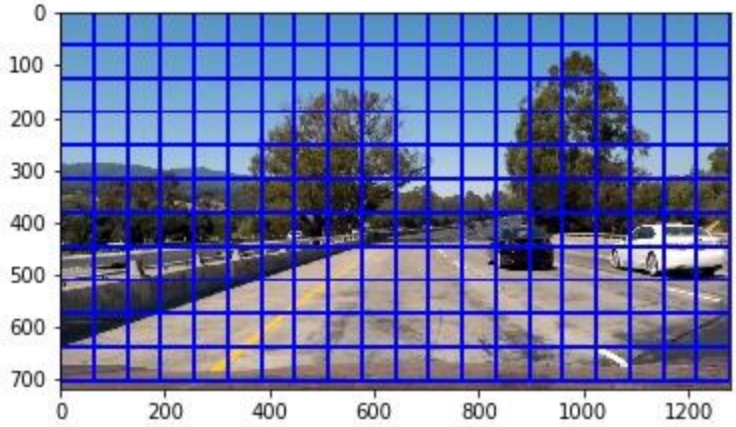


Rubric Point **Describe how (and identify where in your code) you implemented a sliding window search. How did you decide what scales to search and how much to overlap windows?**

9- Sliding Window Search

Define a function that takes an image, start and stop positions in both x and y, window size (x and y dimensions), and overlap fraction (for both x and y). And Loop through finding x and y window positions.

Then make draw box function just to plot an example of draw boxes all over image:
Code cell: 15

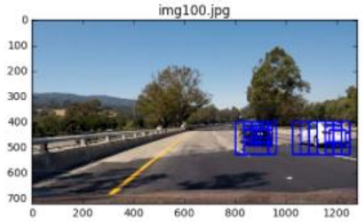


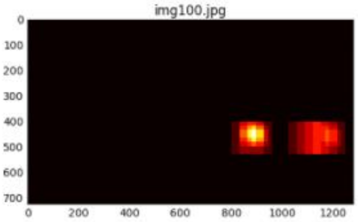
10- Search windows and Find car (extract features using hog sub-sampling and make predictions)

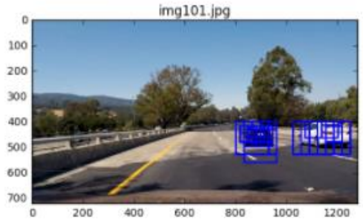
Define a function (find car) you will pass an image and the list of windows to be searched (by search windows function) are output of slide_windows()),then Return windows for positive detections
Code cell : 16

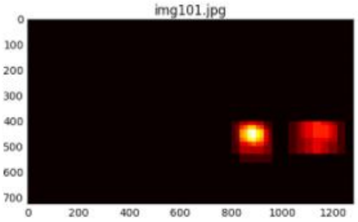
Few false positives examples before I solve that In pipeline function later:

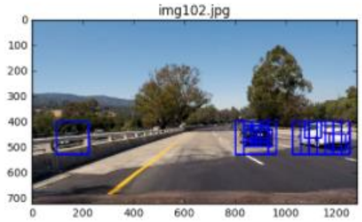
Rubric Point Show some examples of test images to demonstrate how your pipeline is working. How did you optimize the performance of your classifier?

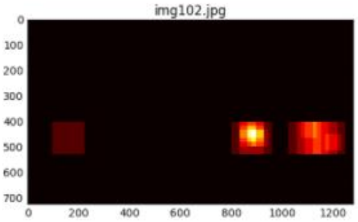

img100.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

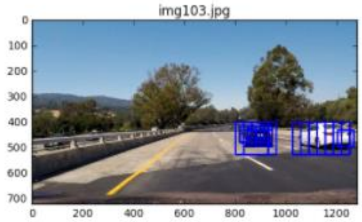

img100.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

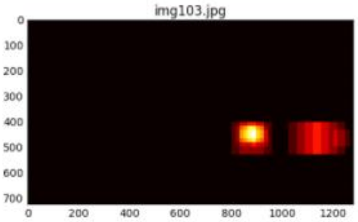

img101.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

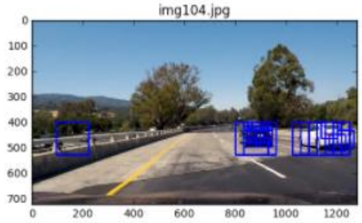

img101.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

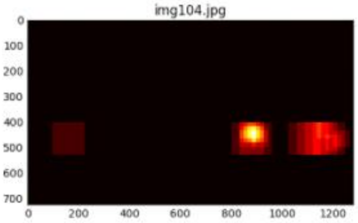

img102.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

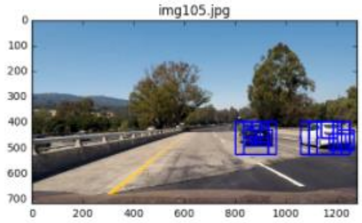

img102.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

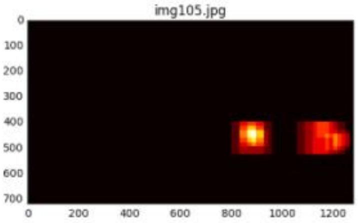

img103.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


img103.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


img104.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


img104.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


img105.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200


img105.jpg
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200

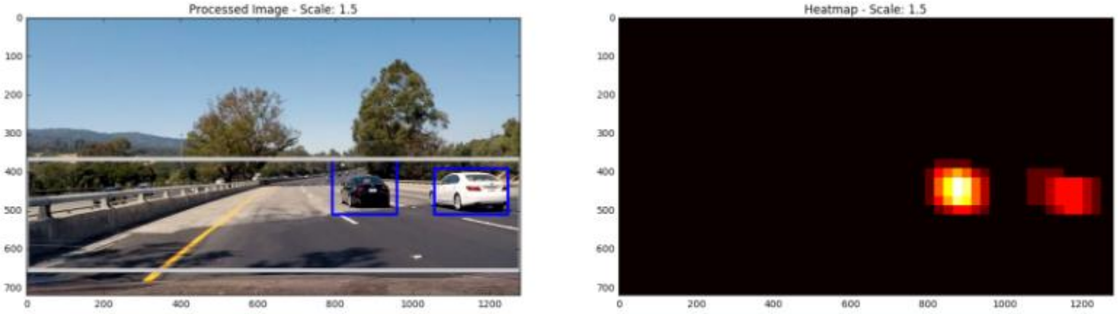
Rubric point: Describe how (and identify where in your code) you implemented some kind of filter for false positives and some method for combining overlapping bounding boxes.

11- Put all process in one (pipeline) function

Get the mean of the previous heatmap, Cool down the heatmap and reset pixel values less than zero to zero, From positive detections I created a heatmap , then thresholded that map to find vehicle positions. I assumed each remaining block to correspond to a vehicle. I make bounding boxes to cover the area of each detected block.
And for multiple window sizes, I create a heatmap for each window size, then use thresholding of the added heatmap to reduce false positives like multiple boxes for same car into a single bounding box.

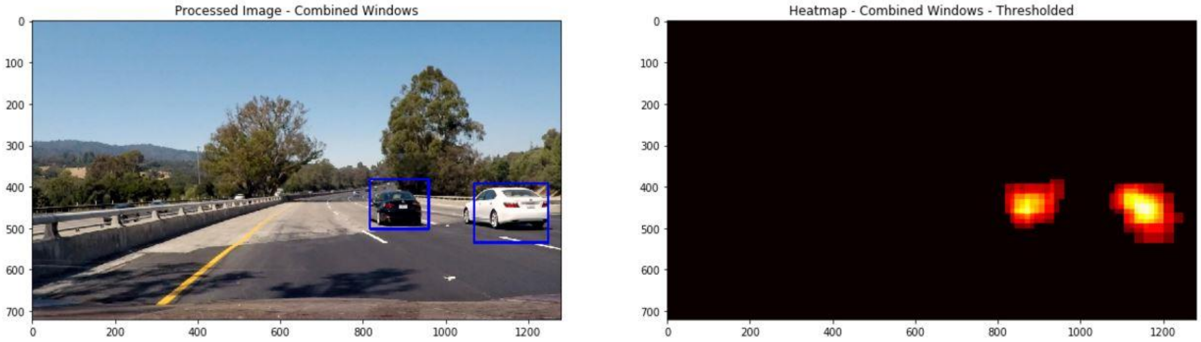
Code cell: 18

Some example of improvement that:



Processed Image - Scale: 1.5

Heatmap - Scale: 1.5



Processed Image - Combined Windows

Heatmap - Combined Windows - Thresholded

12- Process Video

Create process function to make image processing pipeline for the video https://youtu.be/Qrp3o8eG59w

Rubric Point: 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?

My pc is very slow so I just used part from data. And for this reason also I can’t tune many times just few times so maybe after I but new pc I can reach better accuracy. Because no one or library in my village have pc or laptop higher than mine.