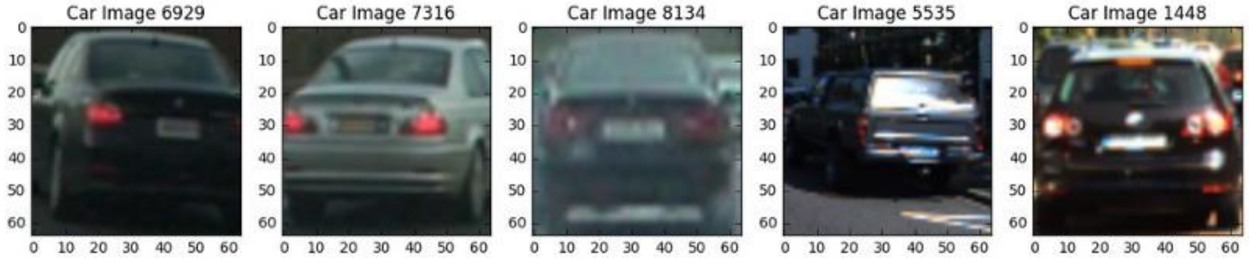
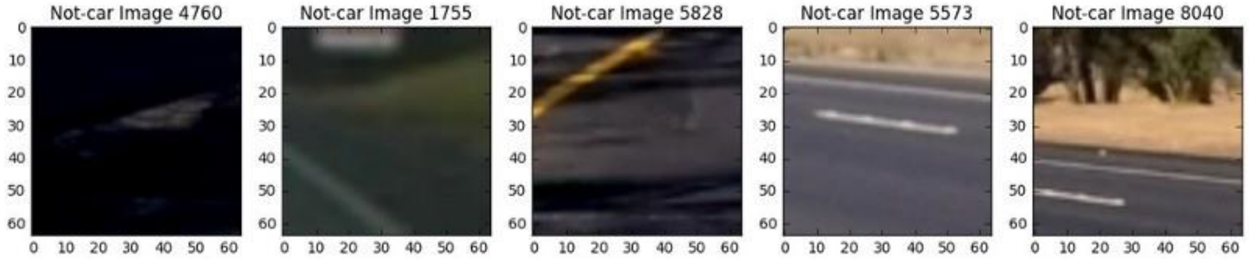
Vehicle Detection

Data Analysis

These images have to be extracted from real world videos and images, and correctly labeled.





Feature extraction

Color Binning:

The image is converted to YCrCb. Spatial binning was done to resize the images to size 32*32 and the 3 channel values are concatenated.

Color Features:

The image is converted to YCrCb and the corresponding channels values are appended together.

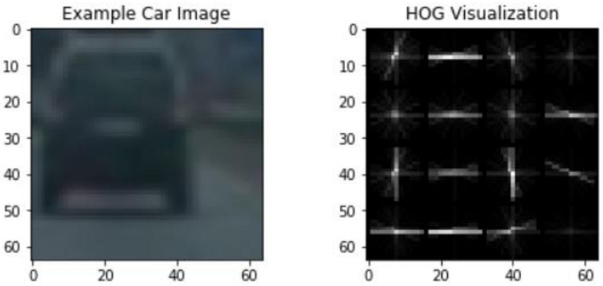
HOG Features:

By tuning the parameters, the orientations of 9, and 8 pixels per cell are enough to identify a car in the YCrCb color space. The HOG was distinct for a car in the YCrCb space.

The following are the final parameters chosen for HOG, Spatial Bins and Color Histograms:-

- Color Channel - YCrCb
- Orientation bins - 9
- Pixels per cell - 8
- Cells per block - 2
- Histogram Bins - 32
- Spatial Image Size - (32, 32)

hog feature is shown below:



Example Car Image

HOG Visualization

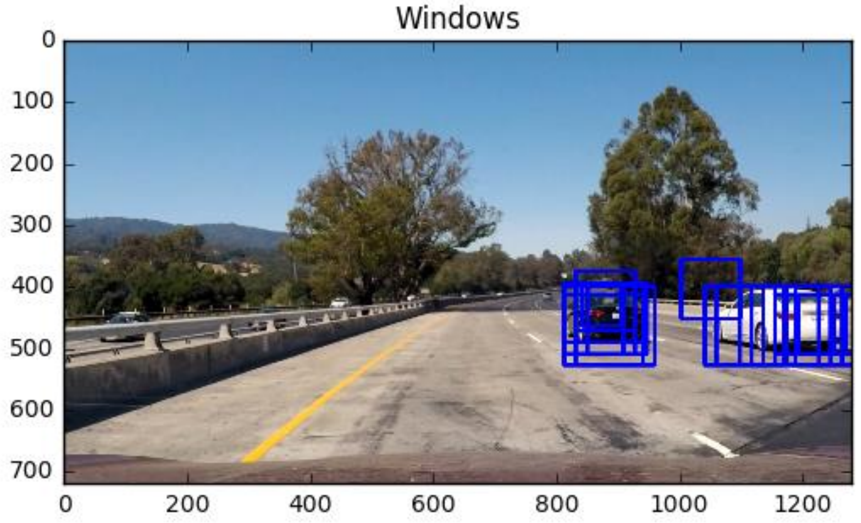
Classification

The next step is to train a classifier. It receives the cars / non-cars data features, and returns if the data sample is or not a Car. I have trained my classifier using Linear SVC with 10000 samples in each of car and non-car. In order to have a robust classifier some simple but efficient step were taken:

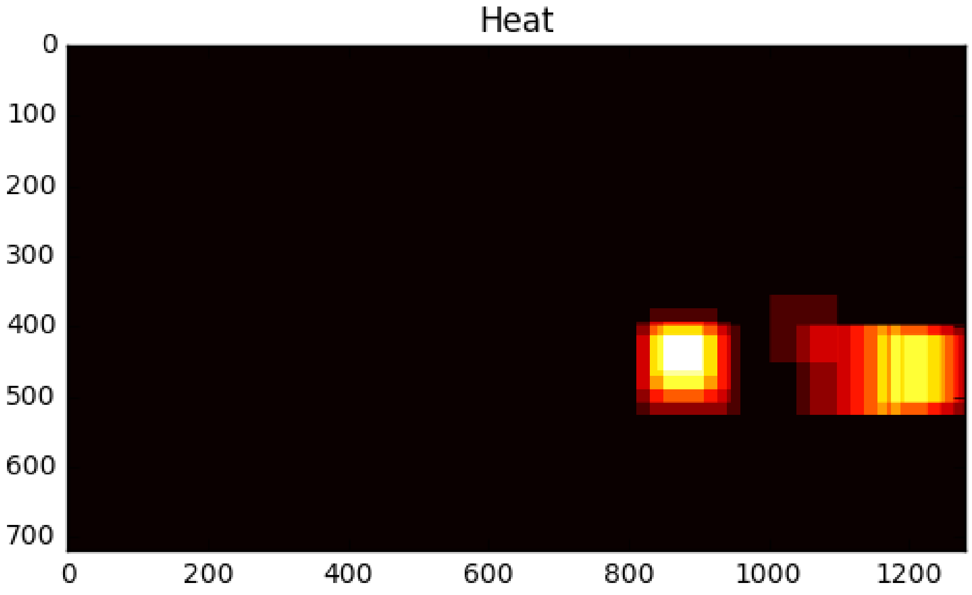
- The training data was randomized so as to prevent any biasing over time space, as some of the data is collected from the video on which it is being tested on. Randomization helped better the classifier.
- The features on which it the classifier was trained consisted of 3 unique features, namely the spatial binning to get the raw color information, the color histogram and the HOG features to get the shape information. Combining all these 3 unique features into one feature and normalizing the feature set to 0 mean and unit variance.

Applying the classifier in an image frame

The car can appear in different sizes so different window sizes are required to detect the car. I have used scales of 1.3, 1.5, 1.8 which corresponds to window of 83, 96 and 115 respectively



Heatmaps are used to find the actual car in the image as shown below.



Smoothing

One problem of the method described so far is that it detects a lot of false positives. To avoid false positives, we do an average over 10 frames of images. A real car is probable to keep appearing in the image. A false positive will disappear.

Discussion

-There are still some false positives in the resultant video. The pipeline needs to be improved further.

-Wouldn’t be used in real world application as-is, since it is likely to fail in too many scenarios