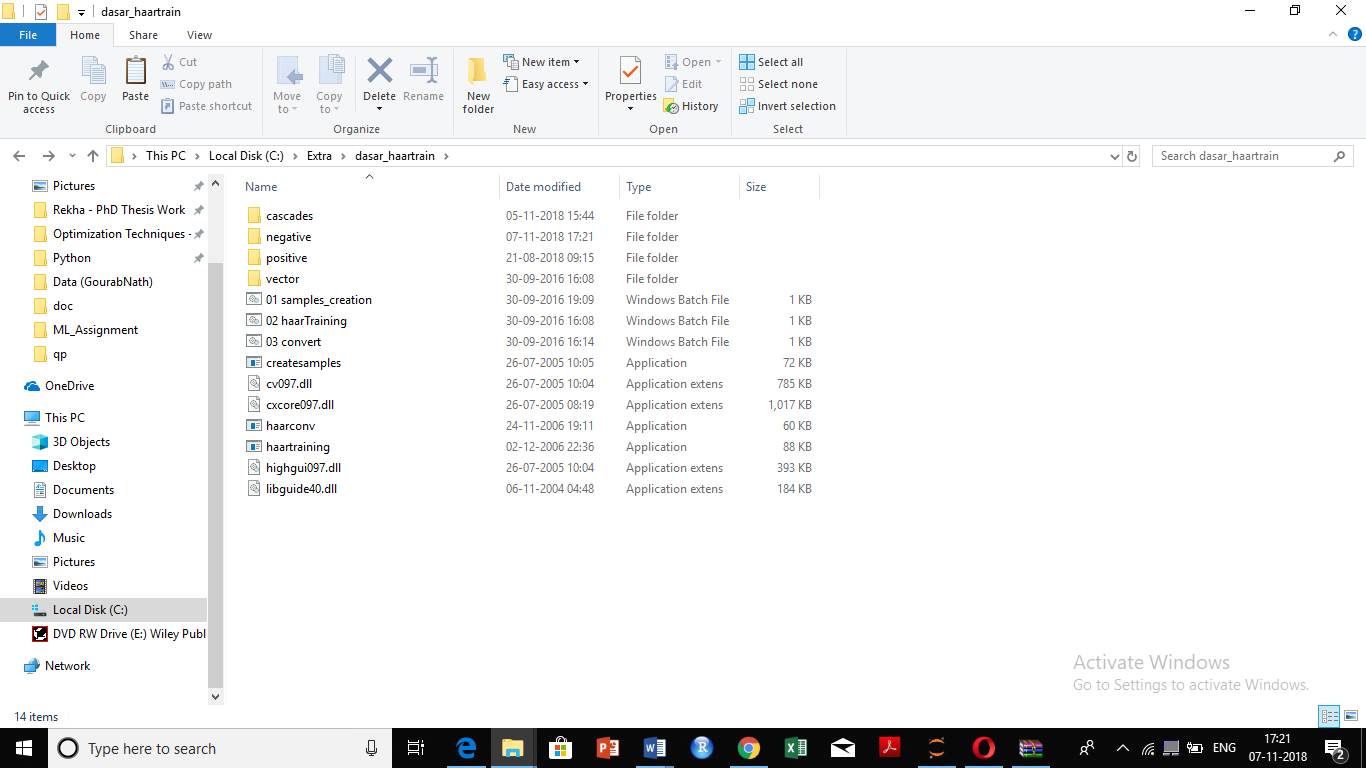
**Fitting the Viola Jones Object Detection Model**

In this stage we will create the haar cascade. We will get the cascade in the form of an XML (extensive mark-up Language) file.

We have got another folder with us, which contains the following files/folders:



* Inside the positive folder there is a folder called rawdata which contains the positive images.
* The negative folder contains the negative images and a file bg.txt which has the names of the negative images.
* The folders cascade and vectors are initially empty.

**STEPS FOR MODEL BUILDING:**

**Step 1: Creating a vector of positive images**

Right Click on 01 samples\_creation.bat 🡪 Open 🡪 Enter the no, of positives (This number should be same as in the info.txt file)

The content of the 01 samples\_creation.bat file is:

createsamples.exe -info positive/info.txt -vec vector/facevector.vec -num 200 -w 24 -h 24

Main Parameters:

-info positive/info.txt Path for positive info file

-vec vector/facevector.vec Path for the output vector file

-num 200 Number of positive files to be packed in a vector file

-w 24 Width of objects

-h 24 Height of objects

The batch file loads info.txt and packs the object images into a vector file with the name of e.g. facevector.vec

After running the batch file, you will have the file facevector.vec in the folder..\training\vector

**Note:** To run creatsample.exe you also needs the files cv097.dll,cxcore097.dll, highgui097.dll, and libguide40.dll in the folder ..\training.

**Step 2: Haar-Training**

Right click on the haartraining.exe file 🡪 Open (for modification)

The content of the file is:

haartraining.exe -data cascades -vec vector/vector.vec -bg negative/bg.txt -npos 200 -nneg 200 -nstages 15 -mem 1024 -mode ALL -w 24 -h 24 –nonsym

-data cascades Path and for storing the cascade of classifiers

-vec data/vector.vec Path which points the location of vector file

-bg negative/bg.txt Path which points to background file

-npos 200 Number of positive samples ≤ no. positive bmp files

-nneg 200 Number of negative samples (patches) ≥ npos

-nstages 15 Number of intended stages for training

-mem 1024 Quantity of memory assigned in MB

-mode ALL Look literatures for more info about this parameter

-w 24 -h 24 Sample size

-nonsym Use this if your subject is not horizontally symmetrical

The size of –W and –H in harrtraining.bat should be same as what you defined on sample-creation.bat

Harrtraining.exe collects a new set of negative samples for each stage, and –nneg sets the limit for the size of the set. It uses the previous stages’ information to determine which of the "candidate samples" are misclassified. Training ends when the ratio of misclassified samples to candidate samples is lower than threshold. So:

Regardless of the number of stages (nstages) that you define in haartraining.exe, the program may terminate early if we reach above condition. Although this is normally a good sign of accuracy in our training process, however this also may happen when the number of positive images is not enough (e.g. less than 500).

Note: To run haartaining.exe you also needs the files cv097.dll, cxcore097.dll, and highgui097.dll in the folder ..\training.

Make the necessary changes and run the application.

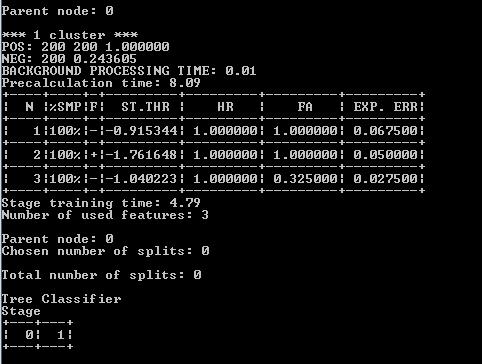
While running the haartraning.exe you will see some information similar to screen below.

Figure: Haartraining output window

Data provided in Image above is related for the first stage of training

Parent node: Defines the current stage under training process

N: Number of used features in this stage

%SMP: Sample Percentage (Percentage of sample used for this feature)

F: “+” if flipped (when symmetry applied) and “–“ if not

ST.THR: Stage Threshold

HR: Hit Rate based on the stage threshold

FA: False Alarm based on the stage threshold

EXP. ERR: Exponential Error of strong classifier

Next figure is the data for the 10th stage of classifier

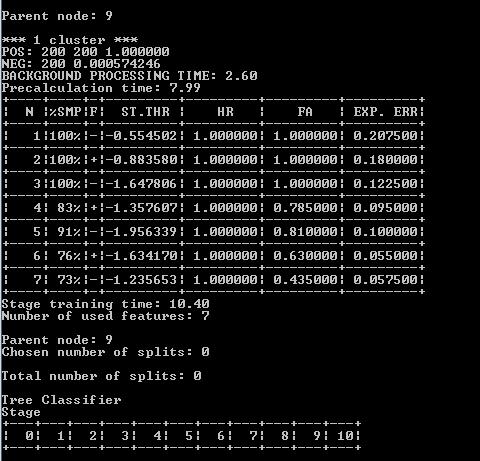
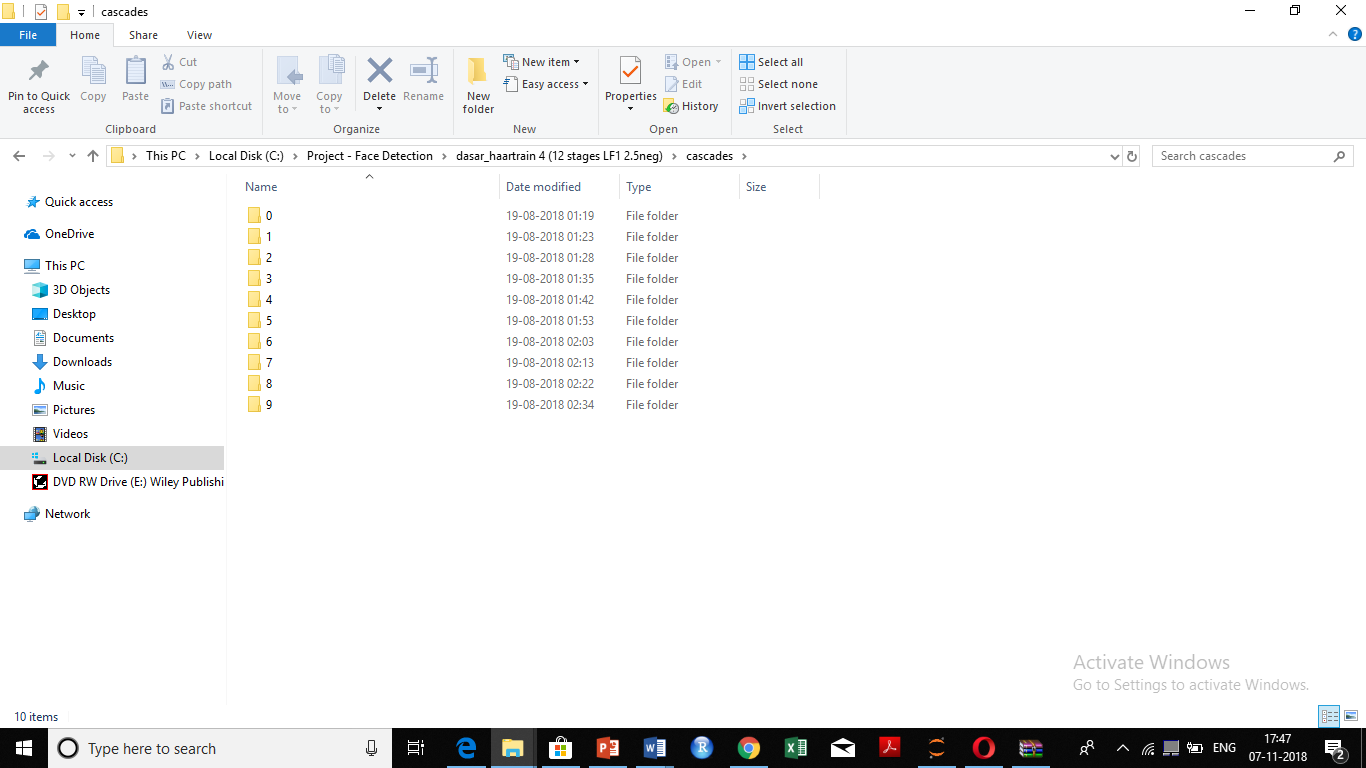


Figure 11: Output of 10 stage classifier

* It can be seen the number of features used in higher nodes are more than the earlier nodes
* The overall false detection (false alarm) has decreased
* and, the computational time for training has increased

**Step 3: Creating the XML File**

After finishing Haar-training step, in folder ../training/cascades/ you should have catalogues named from 0, 1, ..., N-1 in which N is the number of stages you already defined in haartraining.exe. (Note that it need not always be up to N-1. The algorithm may stop at an early stage as well).



In each of those catalogues there should be AdaBoostCARTHaarClassifier.txt file. Copy all the folders 0, 1, ..., N-1 into the folder ../cascade2xml/data/

Now we should combine all created stages (classifiers) into a single XML file which will be our final file a “cascade of Haar-like classifiers”.

Run the batch file **convert.bat** at ../cascade2xml/

Which is:

haarconv.exe data myfacedetector.xml 24 24

myfacedetecor.xml is the output file name and 24 24 are *W* and *H* respectively.

Now we have our own XML file. Point to this classifier from project source code, and run face detection program.