Preprocessing:

* We have start working on preprocessing part of the drive image
* In case of preprocessing of images What we have done is:
* First, we applied masking on the image and stored the images in masking folder.
* After masking of image, we applied clahe on each image for contrast enhancement of vessel structure and stored the clahe enabled images in folder.
* After ‘clahe’, we applied Non-Local Mean to the image to remove the noise generated during clahe and stored these images in a folder.
* There were some problems generated during the entire process of preprocessing.

Like:

* In images like in number ‘34’ image, after application of Non-Local Mean some vessel structures were erased from image.

Training:

* LOADING IMAGES:
* First, we have to load the images, that is we have to make a numpy array of shape (20,584, 565,3)

**Use of Glob.glob:**

* We did this using Glob.glob() function which is taking random images and concatenating the images to from above array, which is major **problem.**
* Size of single image of drive folder is (584, 565, 3)

PATCH CREATION:

* **Things Discovered**
  + - * For purpose of training After preprocessing of image, we have three option to discover:
        + Rule1: We can create 25\*25 patch with overlapping type structure.

**Problem: Memory limit exceeds**

* + - * + Rule2: We can create 25\*25 patches but with overlapping gap of 9\*9 or 11\*11

Here we reduced the number of patches to avoid memory exceed error.

Affect on Accuracy due to less training data.

* + - * + Rule3: We can reduce the patch size to 17\*17 or 19\*19 (haven’t tried it).

**ACC. to Rule 1:**

* Step 1: Separate channels: Input array shape (20, 584, 565, 3)
  + We first have to separate the input array into 3 different channels RED, GREEN, BLUE.
  + RED, GREEN, BLUE SHAPE: (20, 584, 565)
* Step2: Create Patches (Rule 1): input shape (20, 584, 565, 3)
  + We can create patch of 3 different channels separately with each channel shape of flattened array (6599200, 625, 1), there are total 6599200 patches acc. to rule1.
* Step3: Combining Patches:
  + For Each and every patch of every single image All separated channels will now be combined to form a patch of shape (625, 1, 3).
  + As there are 6599200 patches in total, therefore output shape we get is (6599200,625,1,3).
* Step 4: Reshape patches:
  + Input Shape: (6599200,625,1,3)
  + Output Shape: (6599200,25,25,3).

Problems faced during rule1:

* While combining patches we initially were using np.append( ) function for appending and combining the three channels,
  + Due to this in resulting shape the channel was at the first like (3, 625, 1).
  + But we want shape to be (625, 1, 3).
* To solve this problem, we used cv2.merge() function for appending and combining the three channels resulting shape was (625, 1, 3).