```python
def fix(i1,i2,max):
    if(i1 < 10):
        return 0, i2-i1
    if(i2>max):
        return i1-i2+max, max
    return i1,i2
```

```python
def crop(image):
    width,height,n = image.shape
    loc = cv2.minMaxLoc(image[:,:,1])
    x = loc[3][0]
    y = loc[3][1]
    size = width/3

    y1, y2, x1, x2 = y-int(size/2), y+int(size/2), x-int(size/2), x+int(size/2)


    y1,y2 = fix(y1,y2,width)
    x1,x2 = fix(x1,x2,height)




    cropPic = image[y1:y2,x1:x2,]

    return cropPic
```

ในส่วนนี้จะเป็นโค้ดที่ใช้สำหรับ crop ภาพ และฟังก์ชั่น fix มีไว้เพื่อเลื่อนรูปภาพหากจุดศูนย์กลางที่คำนวณได้จาก crop เกิน length ของภาพไป

```
def deletePic(path,num = -1):
    i = 0
    for filename in os.listdir(path)[:num]:
        image = read(path+filename)
        image = crop(image)
        image2 = image[:,:,1]
        countZero = np.count_nonzero(image2 < 4)
        size = image2.shape
        if countZero*1.0/(size[0]*size[1]) < 0.18:
            if num != -1: print("%.2f" %(countZero*1.0/(size[0]*size[1])),"percents.")
            im = Image.fromarray(image)
            im.save(path[:-1]+"new\\"+filename)
        if i% 250 == 0: print(i,"pics done")
        i+=1
```

โค้ดที่เอาไว้คัดรูปภาพที่ตัดภาพไม่ดีออก โดยจะคำนวนจากพื้นที่สีดำของรูปภาพนั้นๆ

```python
def vessel_remove(image):
    qq = 4
    image2 = image.copy()
    gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
    blur = cv2.medianBlur(gray, 5)
    adapt_type = cv2.ADAPTIVE_THRESH_GAUSSIAN_C
    thresh_type = cv2.THRESH_BINARY_INV
    bin_img = cv2.adaptiveThreshold(blur, 255, adapt_type, thresh_type, 31, 5)
    bini = cv2.cvtColor(bin_img, cv2.COLOR_BGR2RGB)
    gg = cv2.cvtColor(bini,cv2.COLOR_BGR2GRAY)

    rr,cc = image2.shape[:2]

    r,g,b = cv2.split(image2)

    for i in range(1,rr):
            for j in range(1,cc):
                if gg[i,j] == 255:
                    i1,i2 = fix(i-qq,i+qq,rr)
                    j1,j2 = fix(j-qq,j+qq,cc)
                    image2[:,:,0][i,j] = np.amax(r[i1:i2,j1:j2])
                    image2[:,:,1][i,j] = np.amax(g[i1:i2,j1:j2])
                    image2[:,:,2][i,j] = np.amax(b[i1:i2,j1:j2])
    for i in range(1,rr):
            for j in range(1,cc):
                    i1,i2 = fix(i-4,i+4,rr)
                    j1,j2 = fix(j-4,j+4,cc)
                    image2[:,:,0][i,j] = np.amax(r[i1:i2,j1:j2])
                    image2[:,:,1][i,j] = np.amax(g[i1:i2,j1:j2])
                    image2[:,:,2][i,j] = np.amax(b[i1:i2,j1:j2])

    return image2
```

# โค้ดสำหรับลบเส้นเลือดออกจากรูปภาพ

```python
def vessel_count(image):
    image2 = image.copy()
    gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
    blur = cv2.medianBlur(gray, 5)
    adapt_type = cv2.ADAPTIVE_THRESH_GAUSSIAN_C
    thresh_type = cv2.THRESH_BINARY_INV
    bin_img = cv2.adaptiveThreshold(blur, 255, adapt_type, thresh_type, 31, 5)
    bini = cv2.cvtColor(bin_img, cv2.COLOR_BGR2RGB)
    gg = cv2.cvtColor(bini,cv2.COLOR_BGR2GRAY)
    return np.count_nonzero(gg == 255)
```

โค้ดสำหรับนับจำนวน pixel ของเส้นเสือด

```python
def segment(image):
    Aro,Ago,Abo = cv2.split(image)
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(9,9))
    Aro = clahe.apply(Aro)
    Ago = clahe.apply(Ago)
    M = 60
    sd = Ago.std()
    filter = signal.gaussian(M, std=6)
    filter=filter/sum(filter)
    STDf = filter.std()

    Ar = Aro - Aro.mean() - Aro.std()

    Mr = Ar.mean()
    SDr = Ar.std()
    Thr = 0.5*M - STDf - Ar.std()

    hist,bins = np.histogram(Ag.ravel(),256,[0,256])

    smooth_hist_g=np.convolve(filter,hist)
    smooth_hist_r=np.convolve(filter,histr)

    r,c = Ag.shape
    Dd = np.zeros(shape=(r,c))
    Dc = np.zeros(shape=(r,c))

    Thg = np.amax(Ago)-sd*1.3+50-sd*1.61

    for i in range(1,r):
        for j in range(1,c):
            if Ar[i,j]>Thr:
                Dd[i,j]=255
            else:
                Dd[i,j]=0

    for i in range(1,r):
        for j in range(1,c):

            if Ago[i,j]>Thg:
                Dc[i,j]=1
            else:
                Dc[i,j]=0

    kernel = np.ones((2,2),dtype=np.float32)
    Dd = resize(Dd,20,20)
    Dd = resize(Dd,r,c)

    Dc = cv2.dilate(Dc,kernel,iterations = 2)
    Dc = cv2.erode(Dc,kernel,iterations = 3)
    Dc = resize(Dc,r,c)

    cv2.imwrite('disk.png',Dd)
    plt.imsave('cup.png',Dc)
```

โค้ดสำหรับสร้างรูป threshold ของ cup และ disc

```
def cdr(image,label):
    try:
        cup = read("cup.png")
        disc = read("disk.png")
        image1 = image.copy()
        image2 = image.copy()
        clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(9,9))

        R1 = cv2.morphologyEx(cup, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(2,2)), iterations = 1)
        r1 = cv2.morphologyEx(R1, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(7,7)), iterations = 1)
        R2 = cv2.morphologyEx(r1, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(1,21)), iterations = 1)
        r2 = cv2.morphologyEx(R2, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(21,1)), iterations = 1)
        R3 = cv2.morphologyEx(r2, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(33,33)), iterations = 1)
        r3 = cv2.morphologyEx(R3, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(43,43)), iterations = 1)
        r3 = cv.cvtColor(r3, cv.COLOR_BGR2GRAY)

        img = clahe.apply(r3)

        cup = cv.cvtColor(cup, cv.COLOR_BGR2GRAY)
        ret,thresh = cv2.threshold(cup,127,255,0)
        contours,hierarchy = cv2.findContours(thresh, cv.RETR_EXTERNAL,cv.CHAIN_APPROX_SIMPLE)


        cup_diameter = 0
        largest_area = 0
        el_cup = contours[0]
        if len(contours) != 0:
            for i in range(len(contours)):
                if len(contours[i]) >= 5:
                    area = cv.contourArea(contours[i])
                    if (area>largest_area):
                        largest_area=area
                        index = i
                        el_cup = cv.fitEllipse(contours[i])

        cv.ellipse(image1,el_cup,(140,60,150),3)
        x,y,w,h = cv2.boundingRect(contours[index])
        cup_diameter = max(w,h)
        ac,bc = max(w,h),min(w,h)
```

โค้ดสำหรับการคำนวณ cup to disc ratio จากภาพ Threshold ที่คำนวนไว้

```
        R1 = cv2.morphologyEx(disc, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(2,2)), iterations = 1)
        r1 = cv2.morphologyEx(R1, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(7,7)), iterations = 1)
        R2 = cv2.morphologyEx(r1, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(1,21)), iterations = 1)
        r2 = cv2.morphologyEx(R2, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(21,1)), iterations = 1)
        R3 = cv2.morphologyEx(r2, cv2.MORPH_CLOSE, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(33,33)), iterations = 1)
        r3 = cv2.morphologyEx(R3, cv2.MORPH_OPEN, cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(43,43)), iterations = 1)
        #f4 = cv2.subtract(R3,img)
        r3 = cv.cvtColor(r3, cv.COLOR_BGR2GRAY)
        img2 = clahe.apply(r3)
        disc = cv.cvtColor(disc, cv.COLOR_BGR2GRAY)
        ret,thresh = cv.threshold(disc,127,255,0)
        contours,hierarchy = cv.findContours(thresh, cv.RETR_EXTERNAL,cv.CHAIN_APPROX_SIMPLE)

        disk_diameter = 0
        largest_area = 0
        el_disc = el_cup
        if len(contours) != 0:
              for i in range(len(contours)):
                if len(contours[i]) >= 5:
                    area = cv.contourArea(contours[i])
                    if (area>largest_area):
                        largest_area=area
                        index = i
                        el_disc = cv.fitEllipse(contours[i])
        image2 = image1.copy()
        cv.ellipse(image2,el_disc,(140,60,150),3)
        x,y,w,h = cv2.boundingRect(contours[index])
        ad,bd = max(w,h), min(w,h)
        disk_diameter = max(w,h)

        if(disk_diameter == 0): return 1
        cdr = (math.pi*ac*bc)/(math.pi*ad*bd)
        return cdr,bc/ac

    except:
        print("error")
        return -1,-1
```

```python
def readPickle(fname):
    features = []
    label = []
    with open(fname,'rb') as file:
        pkl = pickle.load(file)
    for i in range(len(pkl)):
        features.append(pkl[i][0])
        label.append(pkl[i][1])
    features_array = np.asarray(features)
    label_array = np.asarray(label)
    fname = ["area","diameter","green_ratio","blood vessel","label"]
    return sklearn.utils.Bunch(features = features_array, label = label_array,features_name = fname)
```

โค้ดสำหรับอ่านไฟล์ features ที่บันทึกไว้

```python
def trainModel2(X_train, X_test, y_train, y_test,isplot):

    knn = KNeighborsClassifier(n_neighbors=5)
    knn.fit(X_train, y_train)


    svm = SVC(kernel='linear',probability=True)
    svm.fit(X_train, y_train)


    y_pred_svm = svm.predict(X_test)
    y_pred_knn = knn.predict(X_test)

    y_pred_proba_knn = knn.predict_proba(X=X_test)
    y_pred_proba_svm = svm.predict_proba(X=X_test)

    from sklearn.metrics import confusion_matrix
    y_pred = y_pred_svm

    a = confusion_matrix(y_test,y_pred)

    tp,tn,fp,fn = checkk(y_test,y_pred)

    print("tn =",tn)
    print("fp =",fp)
    print("fn =",fn)
    print("tp =",tp)
    sen=tp/(tp+fn)
    print ('\nSensitivity =', sen)
    spec= tn/(tn+fp)
    print ('\nSpecificity =', spec)
    accuracy = (tp+tn)/(tp+tn+fp+fn)
    print ('\nAccuracy =', accuracy,'\n')
    print( "Model = ensemble")
    from sklearn.metrics import classification_report, confusion_matrix
    print('\nClassification Report')
    target_names = ['Glaucomatous', 'Normal']
```

```python
y_pred = y_pred_knn


tp,tn,fp,fn = checkk(y_test,y_pred)

print("tn =",tn)
print("fp =",fp)
print("fn =",fn)
print("tp =",tp)
sen=tp/(tp+fn)
print ('\nSensitivity =', sen)
spec= tn/(tn+fp)
print ('\nSpecificity =', spec)
accuracy = (tp+tn)/(tp+tn+fp+fn)
print ('\nAccuracy =', accuracy,'\n')
print()
from sklearn.metrics import classification_report, confusion_matrix

return svm,knn,accuracy,[y_test,y_pred_proba_svm],[y_test,y_pred_proba_knn]
```

โค้ดสำหรับ train model (SVM,K-nearest neighbors

```python
import seaborn as sn
        import pandas as pd
        import matplotlib.pyplot as plt
def plotcurve(list1,list2,list3,list4,list5,model):
        plt.figure()

        from sklearn.metrics import roc_curve, auc

        rocf1, roct1, roc_auc_1 = roc_curve(list1[0],list1[1][:,1])
        rocf2, roct2, roc_auc_2 = roc_curve(list2[0],list2[1][:,1])
        rocf3, roct3, roc_auc_3 = roc_curve(list3[0],list3[1][:,1])
        rocf4, roct4, roc_auc_4 = roc_curve(list4[0],list4[1][:,1])
        rocf5, roct5, roc_auc_5 = roc_curve(list5[0],list5[1][:,1])

        roc_score1 = roc_auc_score(list1[0],list1[1][:,1])
        roc_score2 = roc_auc_score(list2[0],list2[1][:,1])
        roc_score3 = roc_auc_score(list3[0],list3[1][:,1])
        roc_score4 = roc_auc_score(list4[0],list4[1][:,1])
        roc_score5 = roc_auc_score(list5[0],list5[1][:,1])


        plt.figure(figsize=(8,8))
        plt.title('model %s' %model, fontsize=18)
        plt.plot(rocf1, roct1, color='green', label = 'ROC rool 1 : AUC_SCORE = %0.2f' % roc_score1)
        plt.plot(rocf2, roct2, color='blue',  label = 'ROC rool 2 : AUC_SCORE = %0.2f' % roc_score2)
        plt.plot(rocf3, roct3, color='red', label = 'ROC rool 3 : AUC_SCORE = %0.2f' % roc_score3)
        plt.plot(rocf4, roct4, color='brown',  label = 'ROC rool 4 : AUC_SCORE = %0.2f' % roc_score4)
        plt.plot(rocf5, roct5, color='yellow', label = 'ROC rool 5 : AUC_SCORE = %0.2f' % roc_score5)

        plt.legend(loc = 'lower right')
        plt.plot([0,1], [0, 1],linestyle='--')
        plt.axis('tight')
        plt.ylabel('True Positive Rate', fontsize=18)
        plt.xlabel('False Positive Rate', fontsize=18)
        plt.show()
```

โค้ดสำหรับ plot roc curve graph