Part I. Implementation:

part1:

```python
    # Begin your code (Part 1)
    """first just find the path where the data is
    open the folder
    get the data
    put them into array
    which label 1 for face and 0 for non_face
    later we return the array
    """
    #raise NotImplementedError("To be implemented")
    dataset=[]
    facepath=dataPath+ "/face/"
    for image in os.listdir(facepath):
        dataset.append((cv2.imread(facepath+image,cv2.IMREAD_GRAYSCALE),1))
    nonfacepath=dataPath+ "/non-face/"
    for image in os.listdir(nonfacepath):
        dataset.append((cv2.imread(nonfacepath+image,cv2.IMREAD_GRAYSCALE),0))

    # End your code (Part 1)
```

part2:

```python
    # Begin your code (Part 2)
    """
    get how many row and column in the array
    then we calculate each faeture's error
    to calculate the error we must check whether the value in the featureVal is greater than 0 first
    after checking we just run the formalu to calculate the error that in the course ppt
    we just save each error in the array
    and assume that the first is the smallest
    later i just travel all the array the pick up the best(the smallest) classifier(error feature)
    and just return the error and the best claddifier
    """
    #raise NotImplementedError("To be implemented")
    featurenum=featureVals.shape[0]
    infeaturenum=featureVals.shape[1]
```

```python
    featurenum=featureVals.shape[0]
    infeaturenum=featureVals.shape[1]
    Index=[0]*featurenum
    for i in range(featurenum):
        for j in range(infeaturenum):
            if featureVals[i][j]<0:
                Index[i]+=weights[j]*abs(1-labels[j])
            else:
                Index[i]+=weights[j]*abs(0-labels[j])
    bestClf=WeakClassifier(features[0])
    bestError=Index[0]
    for i in range(1,featurenum):
        if Index[i]<bestError:
            bestError=Index[i]
            bestClf=WeakClassifier(features[i])
    # End your code (Part 2)
    return bestClf, bestError
```

part 4:

移至(G) 執行(R) ··· • detection.py - AI_HW1 - Visual Studio Code

dataset.py | detection.py | main.py | adaboost.py | classifier.py

detection.py > detect

```python
    # Begin your code (Part 4)
    """
    create a array
    open the file with mode txt
    put the data into the array with data spilt
    get the first line of each inage to know the name and how many face in it
    read the image for one colored and one gray
    get the other line of the data and put them to another array
    with the data in the new array just calculate the value to let them can be resize
    and use the interpolation to avoid the pixel loss
    check whether the face is face or not
    put rectangle on them on the colored image
    final place the result to another folder
    """
    #raise NotImplementedError("To be implemented")
    element=[]
    f=open(dataPath, 'r')
```

問題 輸出 偵錯主控台 終端機

項目(S) 檢視(V) 移至(G) 執行(R) ··· • detection.py - AI_HW1 - Visual Studio Code

dataset.py | detection.py | main.py | adaboost.py | classifier.py

detection.py > detect

```python
    f=open(dataPath, 'r')
    for line in f.readlines():
        element.append(line.split())
    temp=0
    while temp < len(element):
        image_name,number=map(str,element[temp])
        image=cv2.imread('data/detect/'+image_name)
        gray_image=cv2.imread('data/detect/'+image_name,cv2.IMREAD_GRAYSCALE)
        temp+=1
        final=[]
        for i in range(int(number)):
            final.append(list(map(int,element[temp])))
            temp+=1
        for face in final:
            left=face[0]
            top=face[1]
            right=face[0]+face[2]
            bottom=face[1]+face[3]
```

33.jpg

es.txt

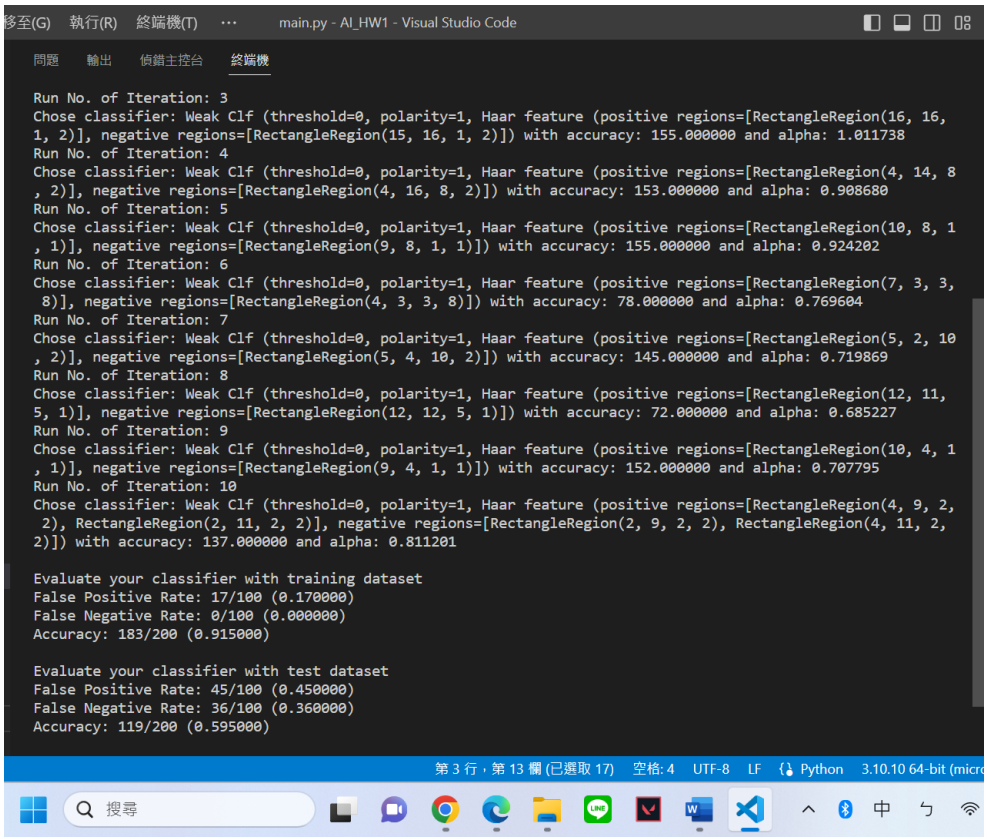
```python
            face_image=cv2.resize(gray_image[top:bottom,left:right],(19,19)
                                  ,interpolation=cv2.INTER_LANCZOS4)
            if clf.classify(face_image)==1:
                cv2.rectangle(image,(left,top),(right,bottom),(0,255,0),thickness=5)
            else:
                cv2.rectangle(image,(left,top),(right,bottom),(0,0,255),thickness=5)
        cv2.imwrite('ans/tt/'+image_name,image)


    # End your code (Part 4)
```

Part II. Results & Analysis

part 1:

part 2:when T=10

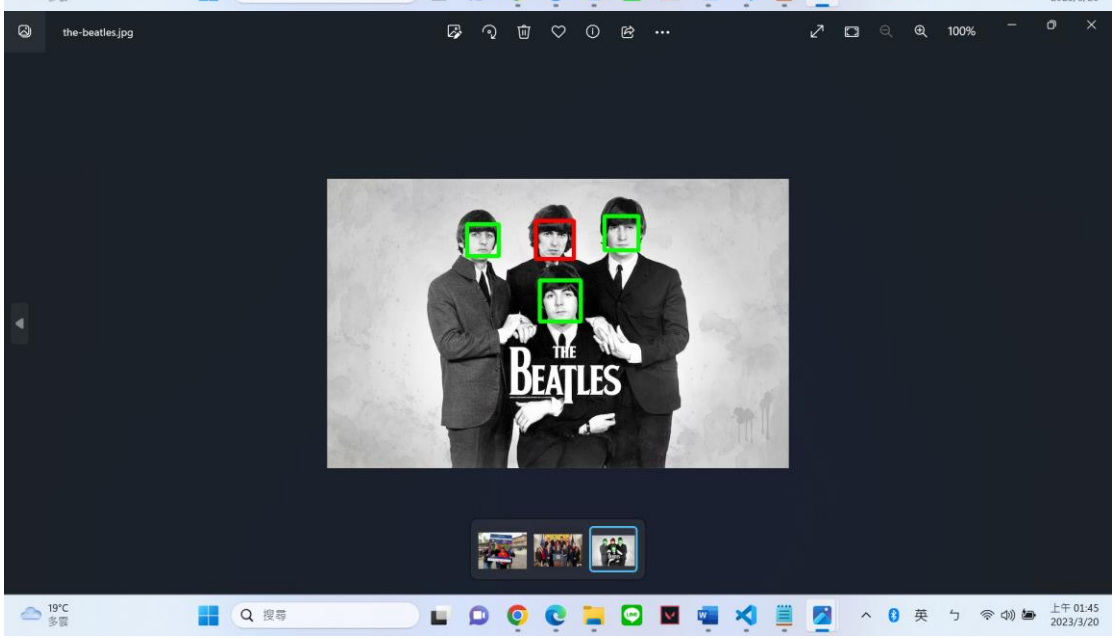


part 3:

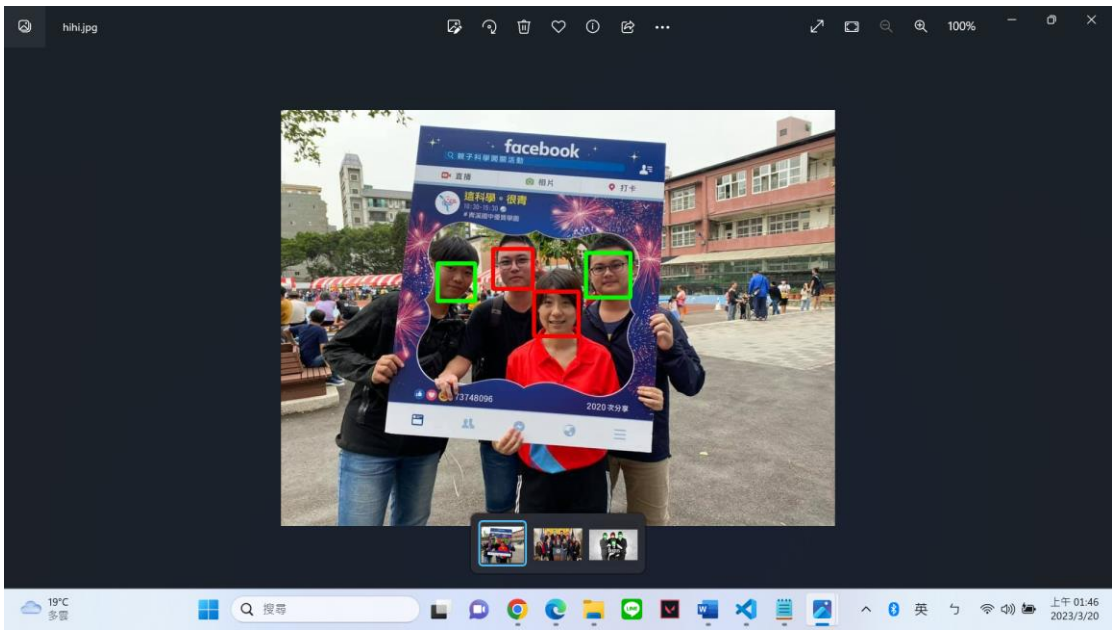
200 張	Train data accuracy	Test data accuracy
METHOD 1 T=1	81%	48%
METHOD 1 T=2	81%	48%
METHOD 1 T=3	88%	53%
METHOD 1 T=4	86%	47.5%
METHOD 1 T=5	88.5%	54%
METHOD 1 T=6	89%	51%
METHOD 1 T=7	90%	54.5%
METHOD 1 T=8	91%	55%
METHOD 1 T=9	90%	57.5%
METHOD 1 T=10	91.5%	59.5%

We can get the better result in train data. Because the classifier is trained from this data. The test data just is the face we want to know to get the classifier good or not. Also when the times go more, the accuarcy is going better for both.

part 4: when T=10



part 5: when T=10



This tell us that this machine is not good enough to detect the human face maybe if we change the size of the square, the result will be different or something else.

Part III. Answer the questions

1. Please describe a problem you encountered and how you solved it.

   in part 2 when I just return the bestclf=feature[i] it will go wrong

   the solution is that :I just go back to see what datatyoe of the clf is so I use Weakclassifier(feature[i]) to replace the original answer.

2. What are the limitations of the Viola-Jones' algorithm?

   restricted to binary classifications

   training time slow

   will be effected by the type of the image(front or high/low exposure)

3. Based on Viola-Jones' algorithm, how to improve the accuracy except changing the training dataset and parameter T?

   ANS: finding a more strictly process to get the best classifier or adjust the threshold.

4. Other than Viola-Jones' algorithm, please propose another possible face detection method (no matter how good or bad, please come up with an idea). Please discuss the pros and cons of the idea you proposed, compared to the Adaboost algorithm.

   ANS: I think is the random method, just generated a number if it is above 0 that it is face. If it is below 0 that it is not face

   advantage: quick.

   disdvantage: low accuracy rate