ML HW2-2

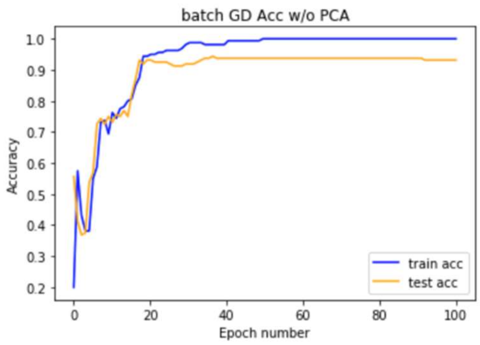
309513121 李偉齊

1.

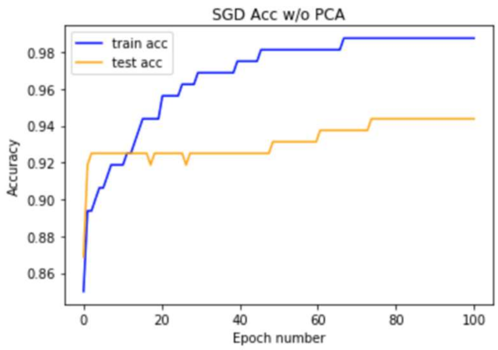
Set the initial weight vector $\mathbf{w}_k = [w_{k1}, \ldots, w_{kF}]$ to be a zero vector where F is the number of features and k is the number of classes. Implement batch GD, SGD, mini-batch SGD (batch size = 32) and Newton-Raphson algorithms to construct a multiclass logistic regression. (15%)

(a) **Plot** the learning curves of $E(\mathbf{w})$ and the accuracy of classification versus the number of epochs until convergence for training data as well as test data, e.g.

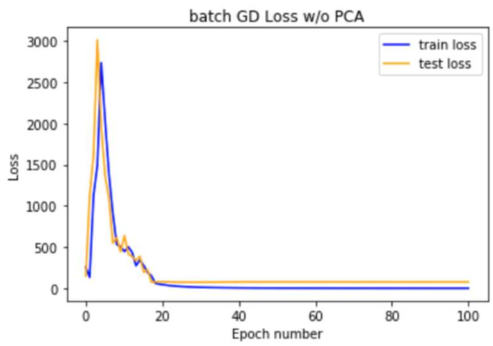
(b) **Show** the classification results of training and test data.

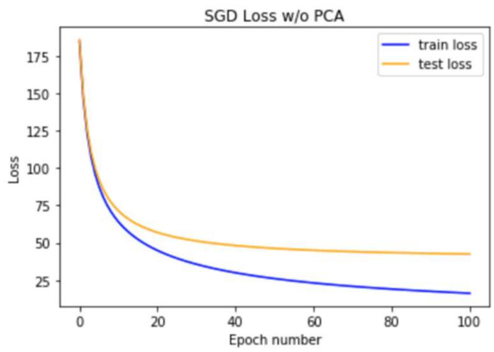


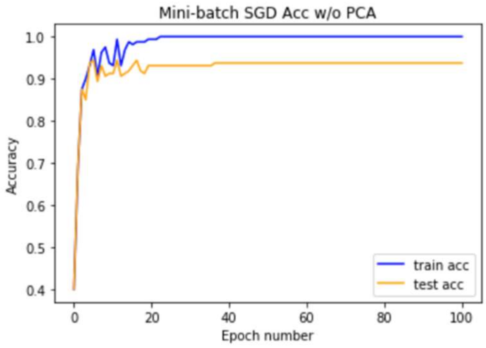

```
TYPE: batch GD Acc w/o PCA
Training ACC: 1.0
Test ACC: 0.93125
```

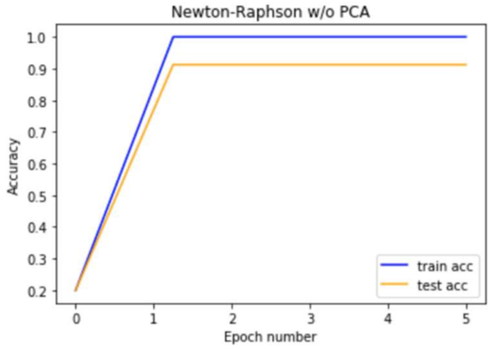



```
TYPE: SGD Acc w/o PCA
Training ACC: 0.9875
Test ACC: 0.94375
```



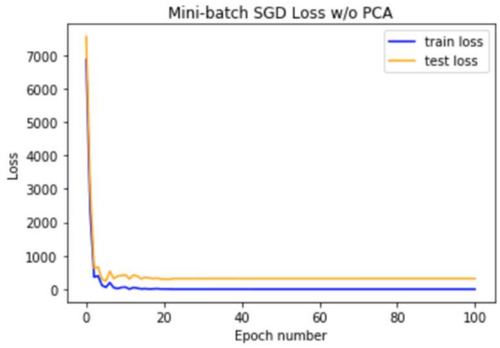


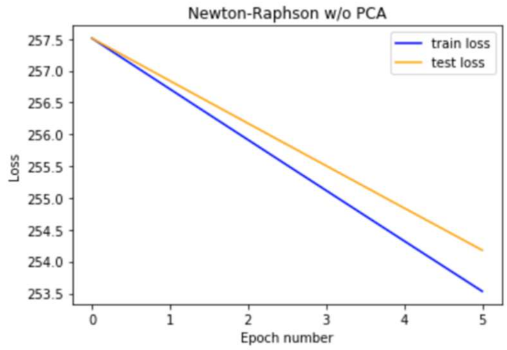





```
TYPE: Mini-batch SGD Acc w/o PCA
Training ACC: 1.0
Test ACC: 0.9375
```

```
TYPE: Newton-Raphson w/o PCA
Training ACC: 1.0
Test ACC: 0.9125
```





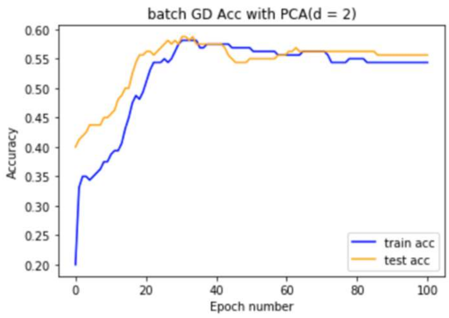
2.

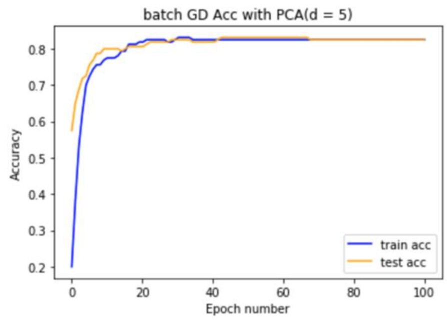
Use principal component analysis (PCA) to reduce the dimension of images to $d = 2, 5, 10$. (15%)

(a) Repeat 1 by using PCA to reduce the dimension of images to d.

(b) **Plot** d eigenvectors corresponding to top d eigenvalues, e.g.

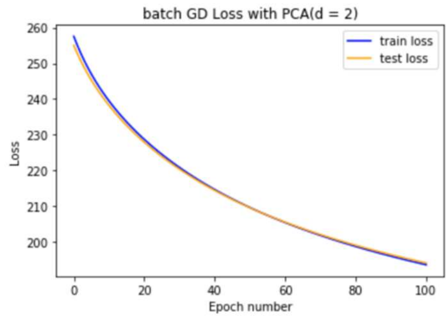
(1) batch GD with PCA (d = 2 ,5,10)

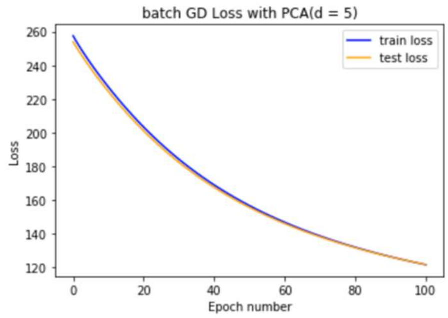


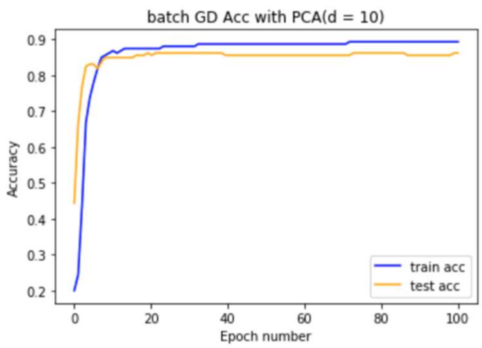



```
TYPE: batch GD Acc with PCA(d = 2)
Training ACC: 0.54375
Test ACC: 0.55625
```

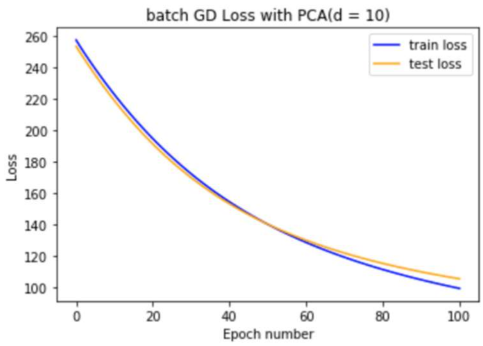
```
TYPE: batch GD Acc with PCA(d = 5)
Training ACC: 0.825
Test ACC: 0.825
```



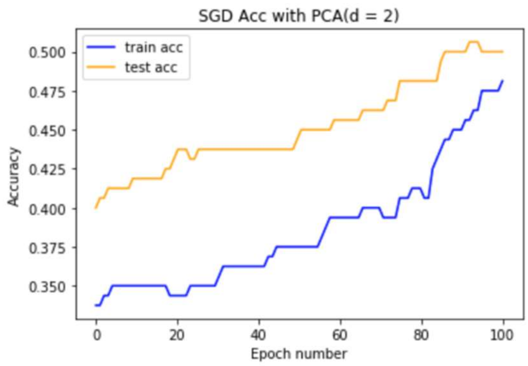




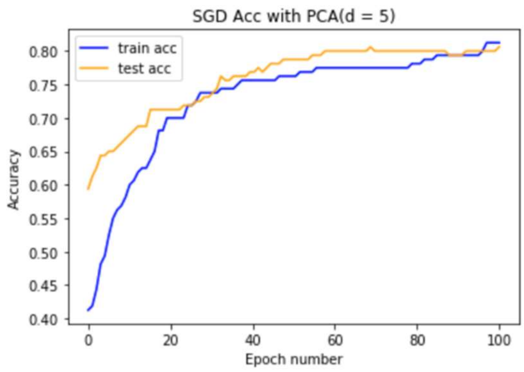

```
TYPE: batch GD Acc with PCA(d = 10)
Training ACC: 0.89375
Test ACC: 0.8625
```



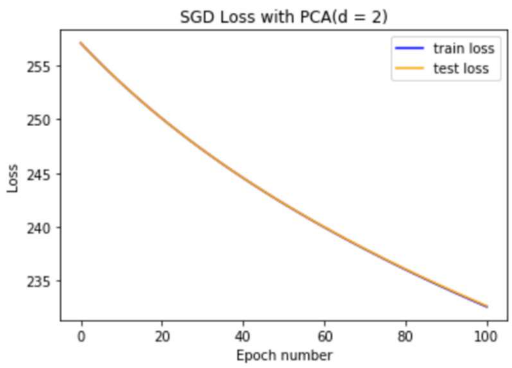
(2) SGD with PCA (d = 2 ,5,10)

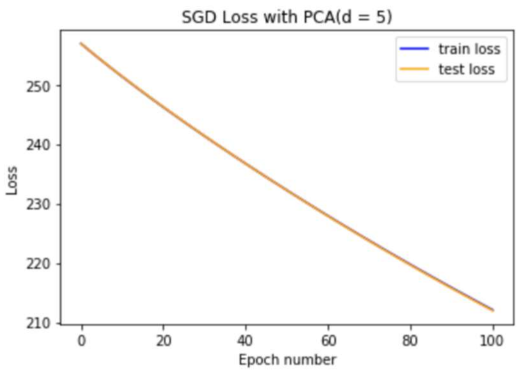


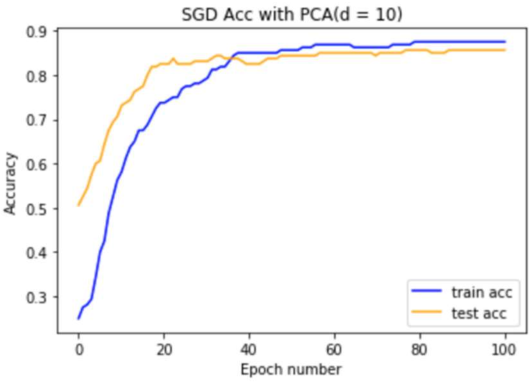

```
TYPE: SGD Acc with PCA(d = 2)
Training ACC: 0.48125
Test ACC: 0.5
```



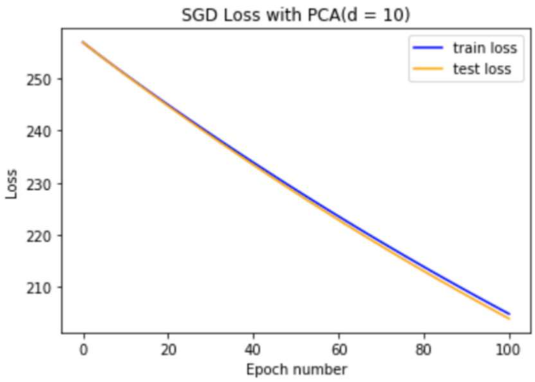

```
TYPE: SGD Acc with PCA(d = 5)
Training ACC: 0.8125
Test ACC: 0.80625
```



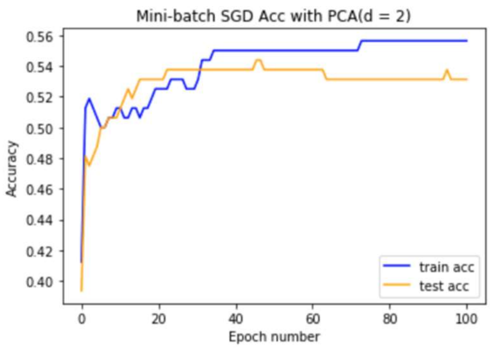




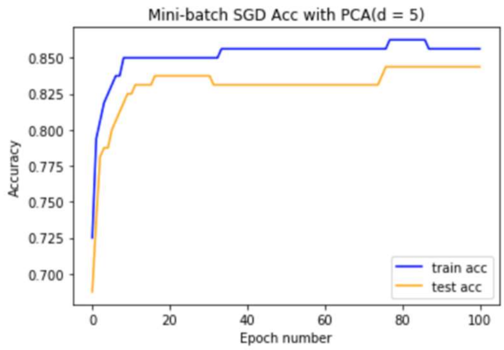
TYPE: SGD Acc with PCA(d = 10)
Training ACC: 0.875
Test ACC: 0.85625



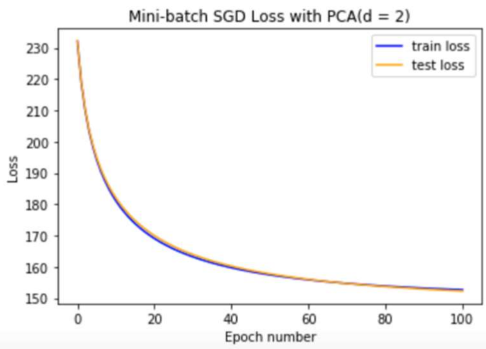
(3) Mini-Batch GD with PCA (d = 2 ,5,10)

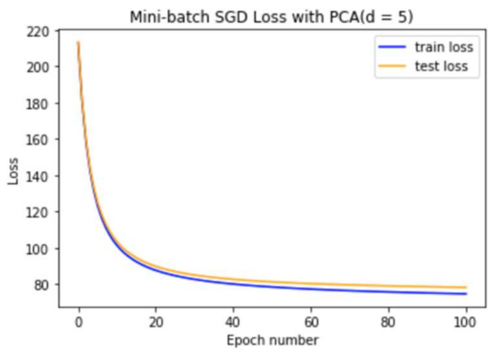


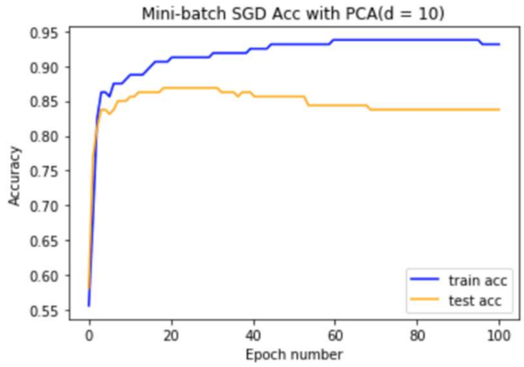
TYPE: Mini-batch SGD Acc with PCA(d = 2)
Training ACC: 0.55625
Test ACC: 0.53125



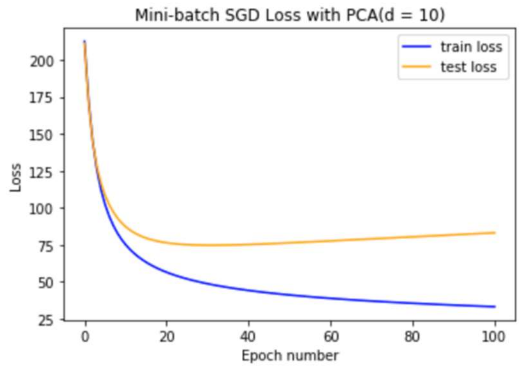
TYPE: Mini-batch SGD Acc with PCA(d = 5)
Training ACC: 0.85625
Test ACC: 0.84375



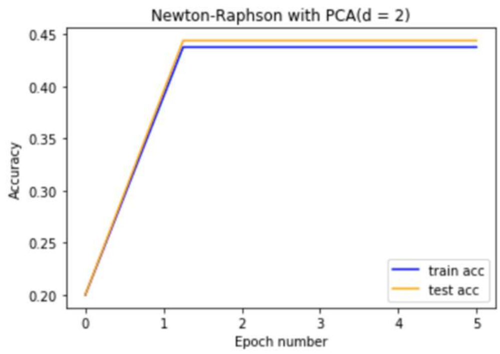


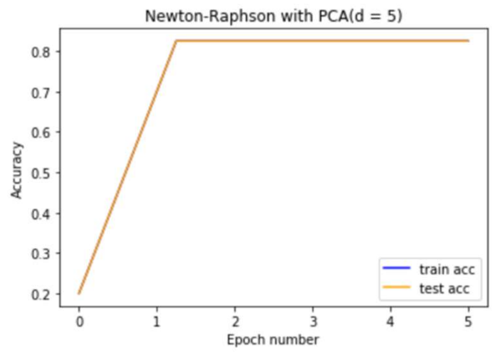



```
TYPE: Mini-batch SGD Acc with PCA(d = 10)
Training ACC: 0.93125
Test ACC: 0.8375
```



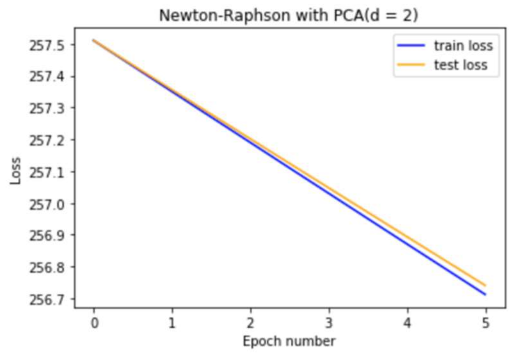
(4) Newton Raphson with PCA (d = 2 ,5,10)

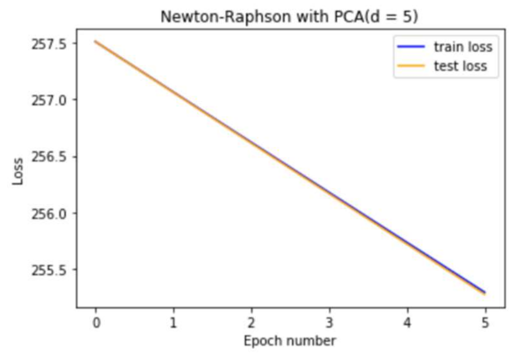


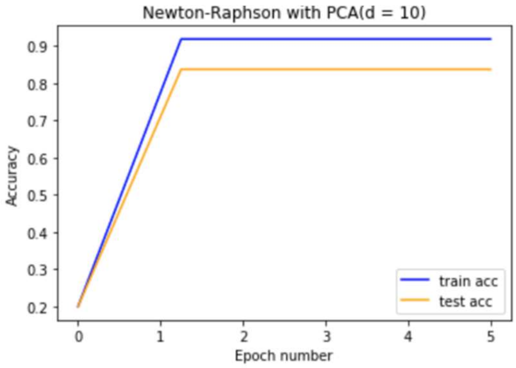



```
TYPE: Newton-Raphson with PCA(d = 2)
Training ACC: 0.4375
Test ACC: 0.44375
```

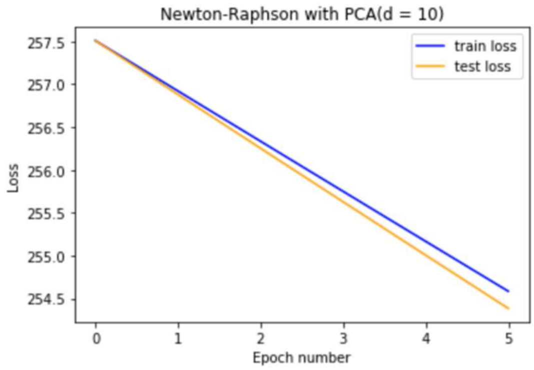
```
TYPE: Newton-Raphson with PCA(d = 5)
Training ACC: 0.825
Test ACC: 0.825
```



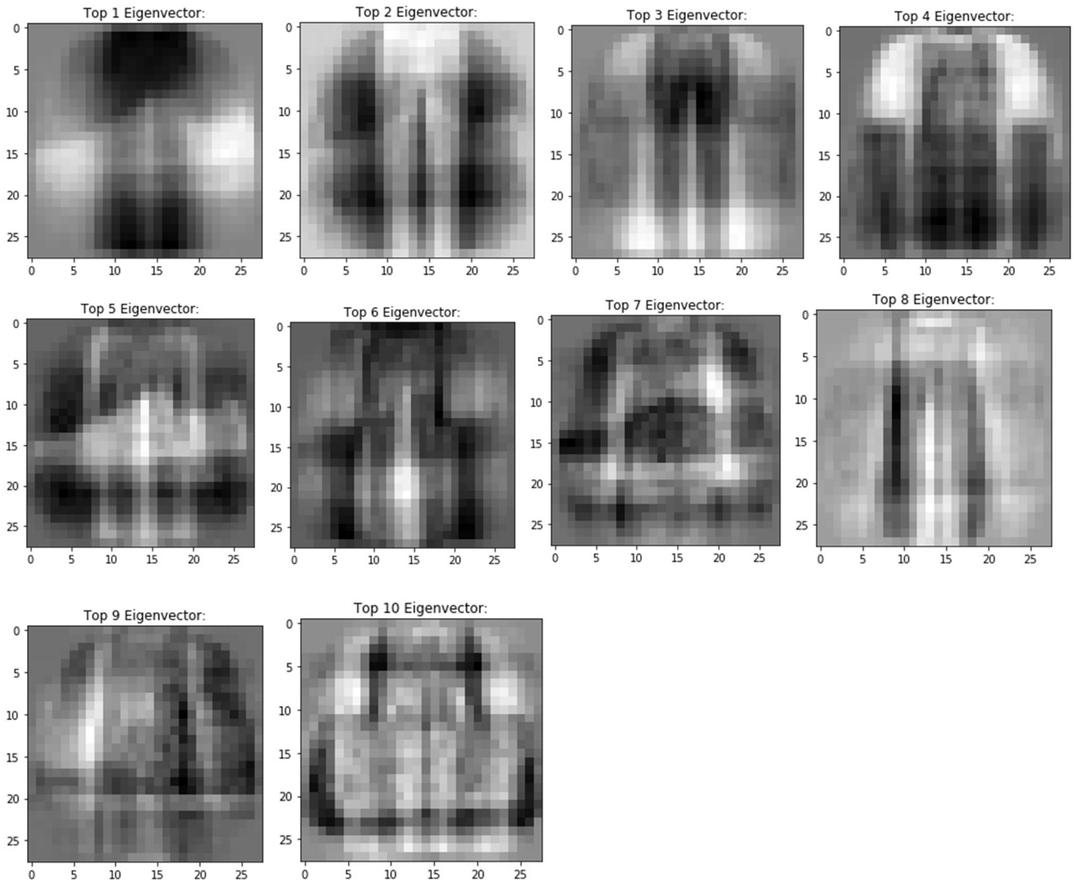





```
TYPE: Newton-Raphson with PCA(d = 10)
Training ACC: 0.91875
Test ACC: 0.8375
```



(b) Top EigenVectors

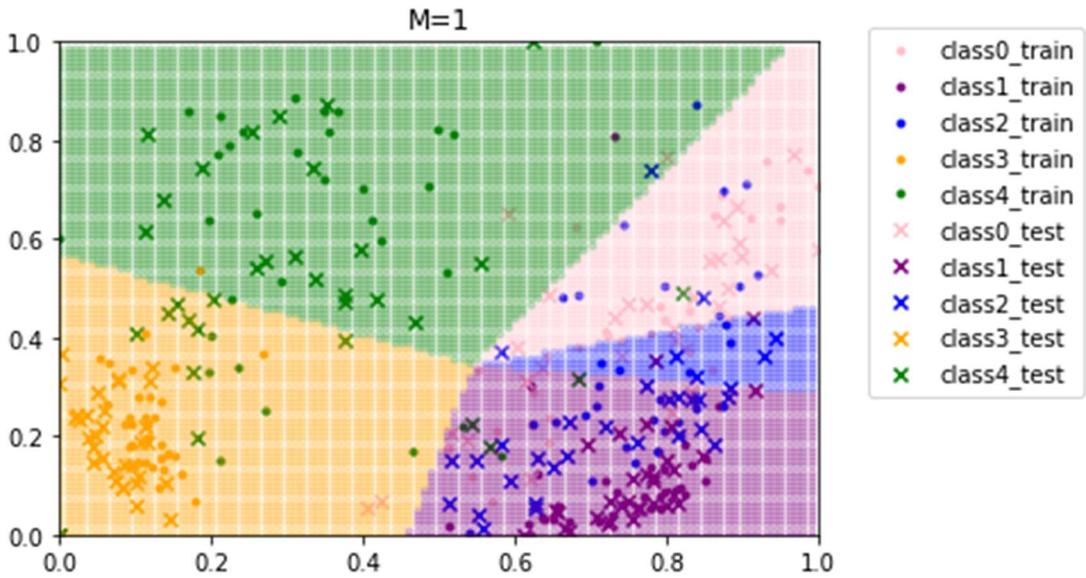


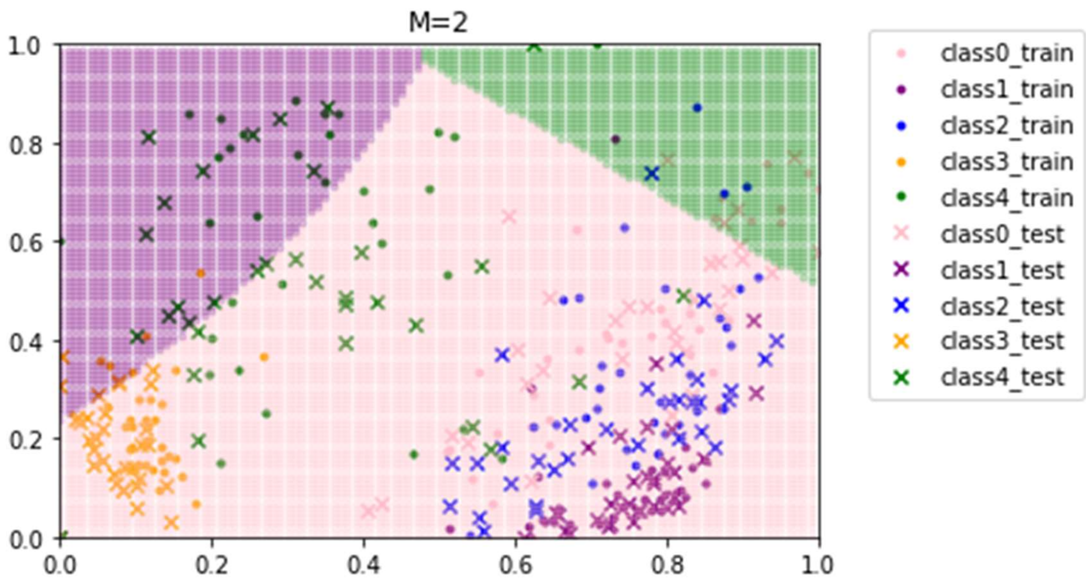
3.

What do the decision regions and data points look like on the vector space? (15%)

(a) **Plot** the decision regions and data points of the images on the span of top 2 eigen-vectors by using PCA to reduce the dimension of images to 2.

(b) Repeat 3(a) by changing the order from $M = 1$ to $M = 2$, e.g.





4. Make some discussion on the results of 1, 2 and 3

(1)

With 4 different methods of logistic regression, expect Newton Raphson method, the others 3 methods use gradient descent to update weight.

The differences between these three methods are batch-sizes and iterations. In batch-GD, we can decide batch-size (not 1) that after input this size of training data, updating the weight. In this question, we choose batch-size = 160, so the total iterations in an epoch is N=training data size/b = batch-size => 160/160 = 1. It means that we update weights 1 times in an epoch.

In SGD, batch-size defined to 1, so the total iterations in an epoch is N=training data size/b = batch-size => 160/1 = 160. It means that we update weights 160 times in an epoch.

In Mini-Batch GD, we can decide batch-size (not 1) that after input this size of training data, updating the weight. In this question, we choose batch-size = 32, so the total iterations in an epoch is N=training data size/b = batch-size => 160/32 = 5. It means that we update weights 5 times in an epoch.

In the conclusion, SGD train too many times in an epoch, batch-GD train a few times in an epoch, and Mini-Batch GD combine both of above methods' benefits, train just fit times in an epoch. I train 100 epochs in each method, Mini-Batch GD present the best accuracy in test data.

Newton Raphson method can find the best solution of weight in one epoch by calculate Hessian Matrix. As the result, I train 5 epochs to show the feature of this method.

(2)

PCA method means that choosing the top "d" eigenvalues and select their eigenvectors to descending the dimension of data.

In this question, we select d = 2, 5, 10 to compare the result in question1.

When d = 2, all of the accuracies are very low, because there are too less features to input the models. According to this assume, d= 10 show some high accuracies in all of the models.

In question (b), I choose the top 10 eigenvalues and reconstruct their images by their eigenvectors. It shows that the max eigenvalue's result is the clearest.

(3)

In this question, I visualize the result multiclass logistic regression with Newton Raphson method and pre-process the data with PCA, d = 2.

As the result above, we can easily distinguish class 0, class 3 and class 4 with only two features and one constant. Nonetheless, we have a bad result on the boundary between class 1 and class 2. Maybe the reason is that the features of class1 and class2 are similar.

This problem can't be solved even in model M = 2, so maybe we should take more features into consideration to do a better classification, like PCA d = 10 or more.