CSE 291
Project 1

Chao Yu A99049546
Wen Liang A53214852

4/25/17

1. methods description

PCA: PCA aims to detect the correlation between variables. If a strong correlation between variables exists, it attempt to reduce the dimensionality to the direction (eigenvector) has greatest variance (eigenvalue). In a nutshell, it find find the directions of maximum variance in high-dimensional data and project it onto a smaller dimensional subspace which can preserve most of the information.

Steps:

- Get the covariance matrix
- Compute eigenvectors and eigenvalues of the covariance matrix
- Sort the pair by eigenvalues and Select K dominant eigenvectors
- Project original data to the subspace

LDA: LDA also aims to reduce the dimension but the fundamental difference between LDA and PCA is that LDA is supervised process and takes that advantage of the "label" information to find suitable directions that maximize the discrimination between classes.

- Get the within-class scatter matrix S_W and between-class scatter matrix S_B
- Compute eigenvectors and eigenvalues for $S_W^{-1}S_B$
- Sort the pair by eigenvalues and Select K dominant eigenvectors
- Project original data to the subspace

Naive Bayes:

Firstly using cross validation to test the training data. From the confusion matrix we will be able to tell if the training data is good or not. Then using training data to perform naive bayes. The posterior probability is equal to likelihood times prior. We assume the prior are the same for each class and the likelihood to be a Gaussian distribution. After calculating the posterior probability, we can get our predict result. The boundary is 0.5.

2. Eigenvectors

PCA

Wine dataset

The most important eigenvector: [4.9746983159876343, 1.8207111697712279, 1.5264854238606775, 1.2037605666770128, 0.87193645865543856, 0.73494357883134165, 0.61289148567954266, 0.43280006702356399, 0.37272730553772926, 0.29763729894947633, 0.20032975003884818, 0.19237692826604838, 0.078079972016062607, 0.031188166481191187]
The 20th eigenvector is a null space {0}

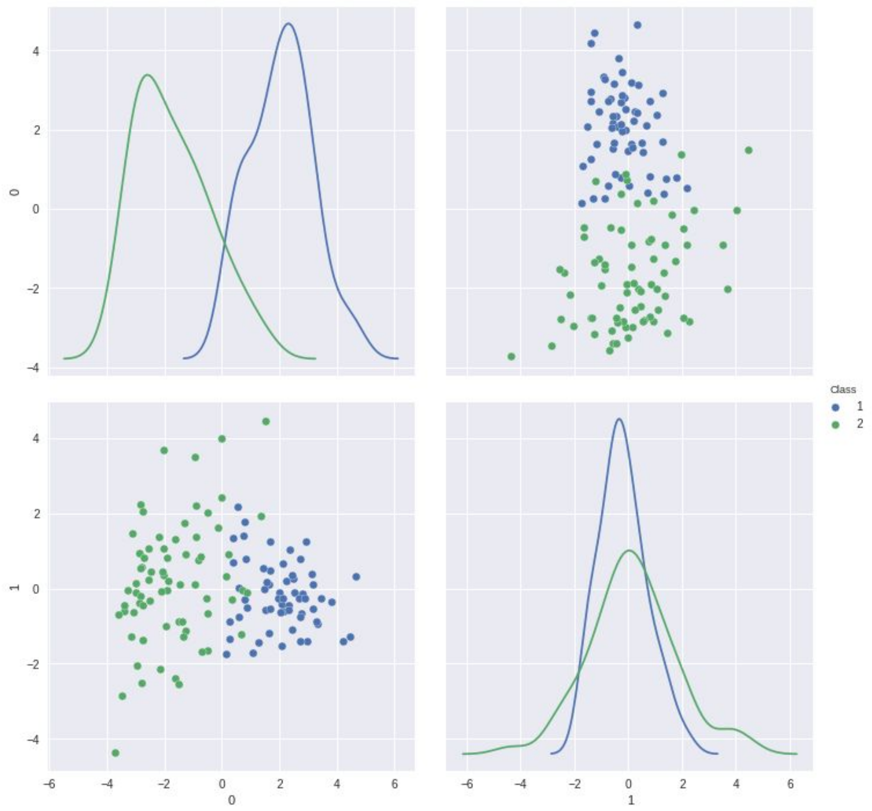

Figure1:. First 2 dimension of PCA projected data

0,1 dataset

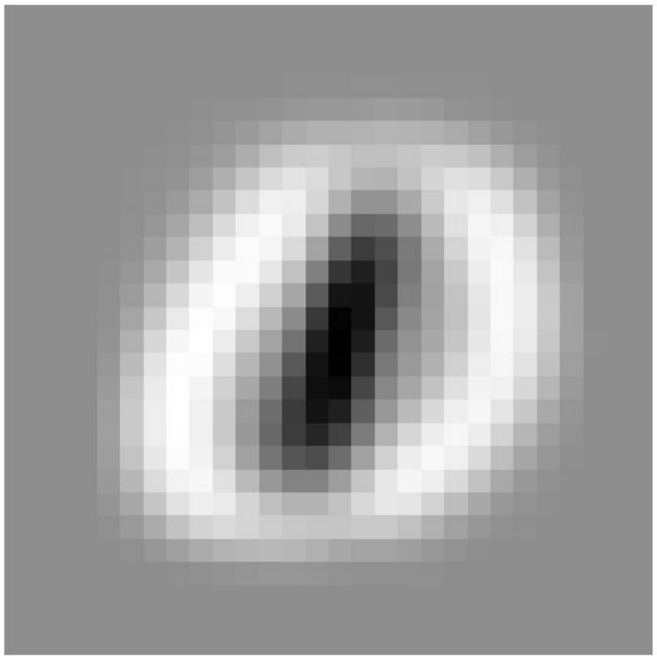

Figure2:The most important eigenvector

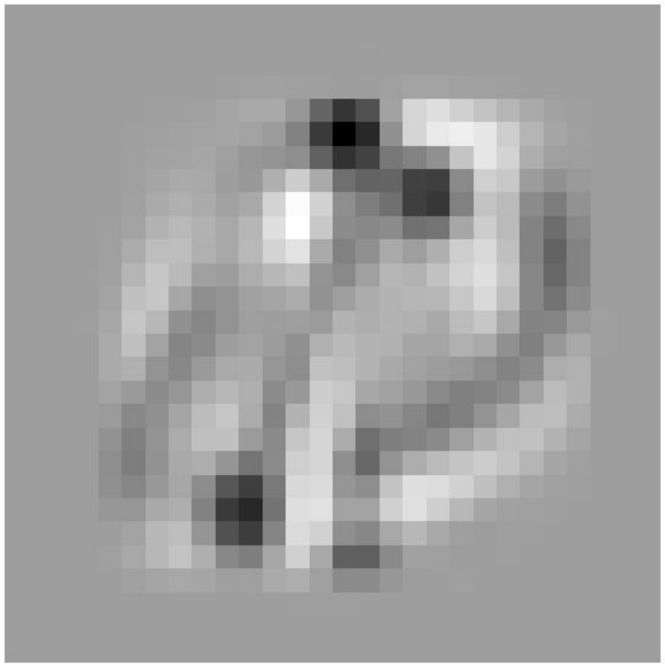

Figure3:The 20th eigenvector

3,5 dataset

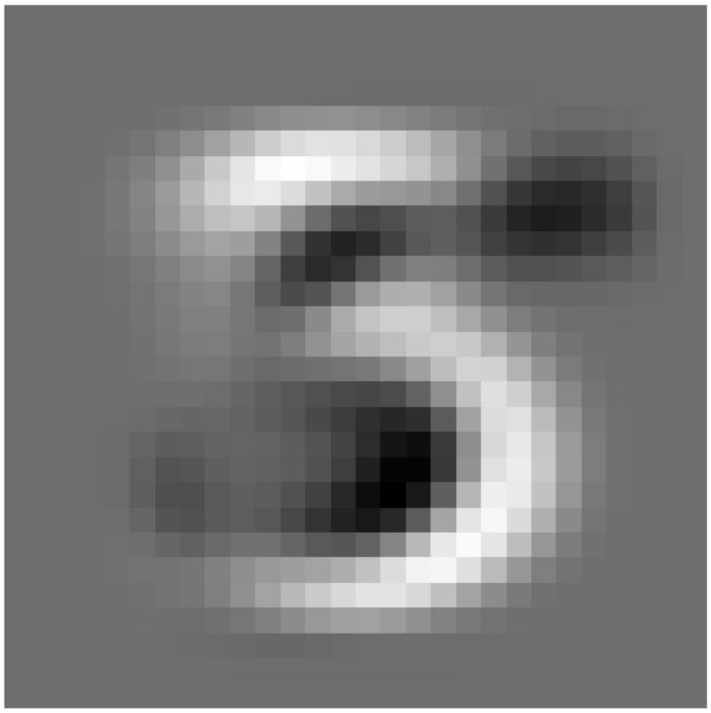

Figure 4:The most important eigenvector

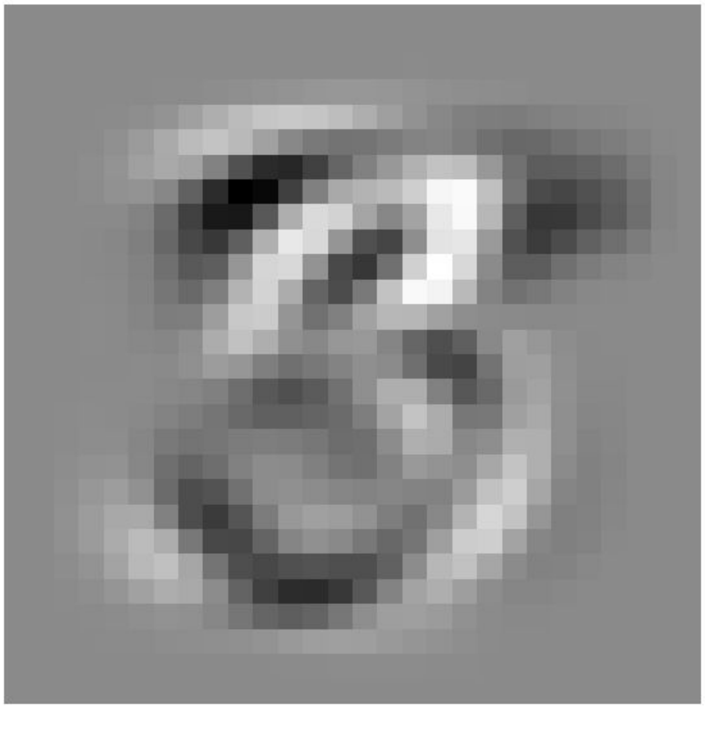

Figure 5: The 20th eigenvector

LDA

Wine dataset

The most important eigenvector: matrix([[0.4687, 0.1076, 0.3256, -0.3702, 0.0025, -0.1217, 0.1364, -0.0235, -0.0715, 0.0439, -0.0205, 0.2367, 0.6544]])
The 20th eigenvector is a null space {0}

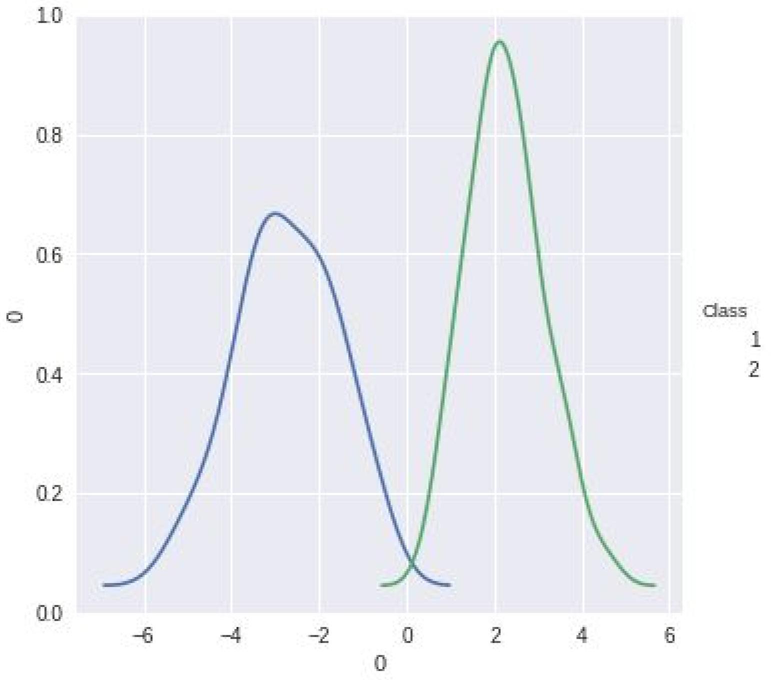

Figure 6: LDA projected data

0,1 dataset

I used all 784 dimensions for this LDA but I got the result below. I think that this process has been affected by the background which all samples contains zero values and has zero variance and covariance because I see there are only 1 or 2 dominant dimension and this is surprising. However, it works fine to classify in the next several section.

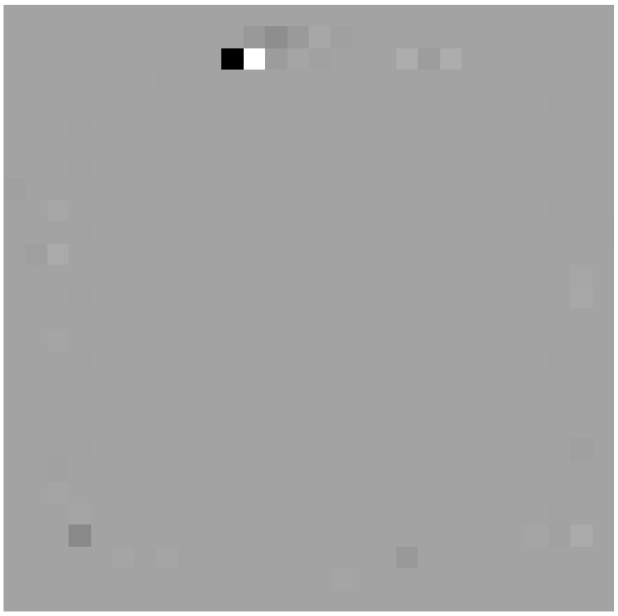

Figure 7: The most important eigenvector

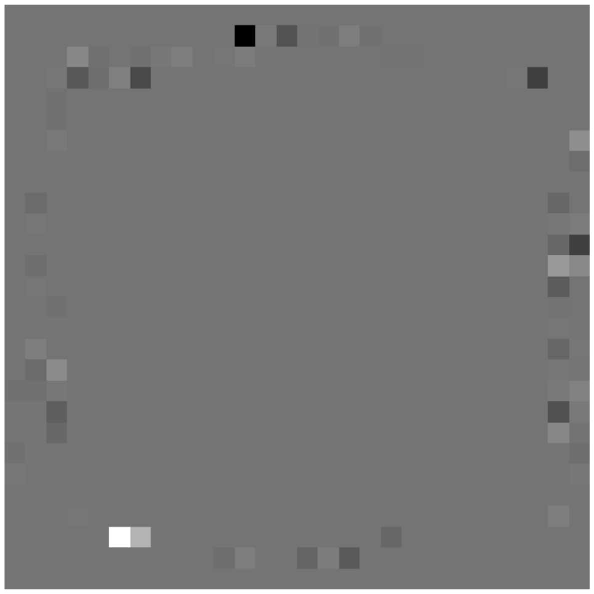

Figure 8:. The 20th eigenvector

3,5 dataset

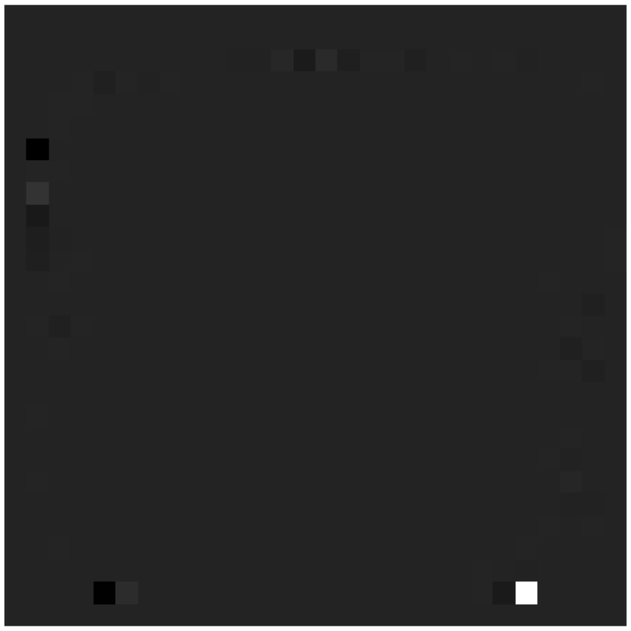

Figure 9. *The most important eigenvector*

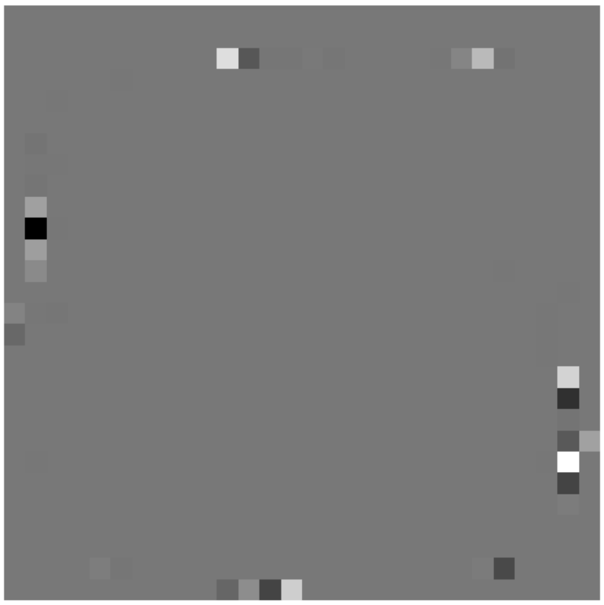

Figure 10: The 20th eigenvector

LDA Method 2

This time I drop those background dimensions to do the LDA and I got a more interpretable result. I reconstructed them to the full dimension at last.

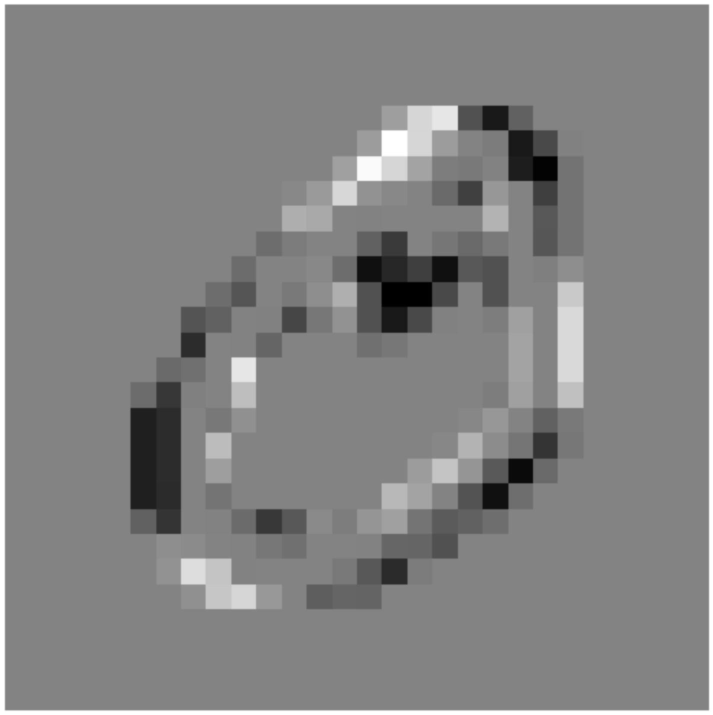

Figure 11: The 1st eigenvector

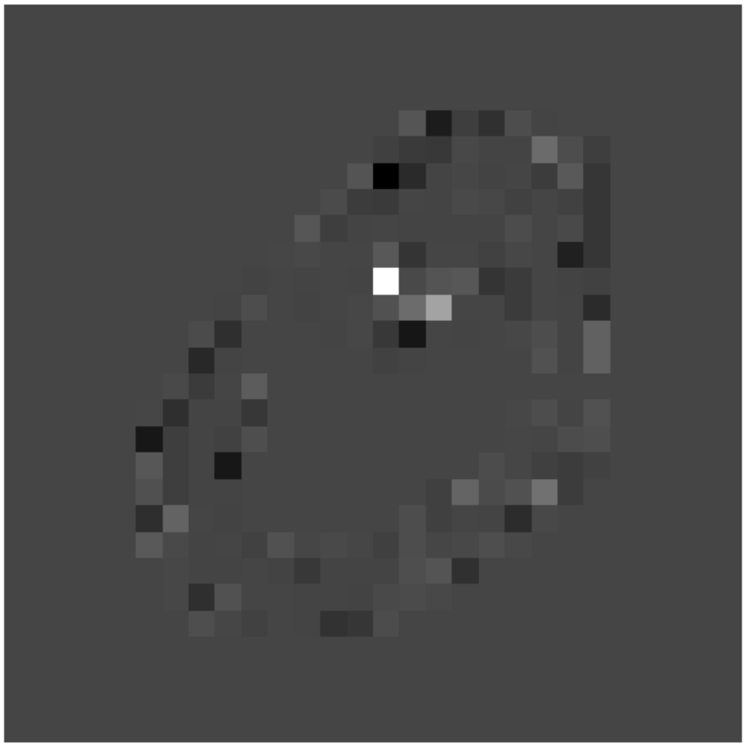

Figure 12: The 20th eigenvector

Results and explanation:

PCA: We can see the shape of "0" from the PCA most important eigenvector. This eigenvector remains the key features of number “0” and ”1”. However, the 20th PCA eigenvectors do not contain as much information as the first one.

For first LDA method, it is a little big hard to interpret the result because some dominant values make the trace of 0 or 1 hard to stand out. Also, the singular Sw may also make negative effect.

For second LDA method excludes background, we can see the trace of 0 and 1 which makes sense.

3. LDA cumulative sum of eigenvalues

For PCA wine data:

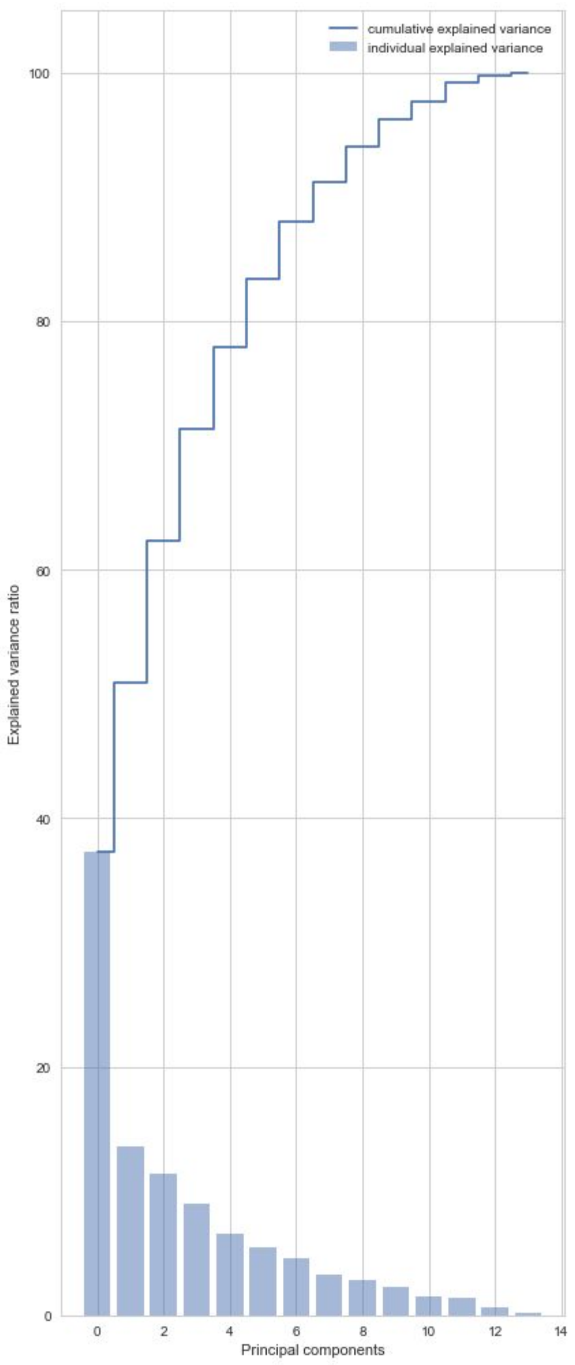

Figure 13:cumulative sum of eigenvalues for wine data(PCA)

We pick the top 4 eigenvectors as our new W. From the Figure, we see the influence of each eigenvalue. The top 4 eigenvalues weights about 75 percent of all eigenvalue.

For LDA

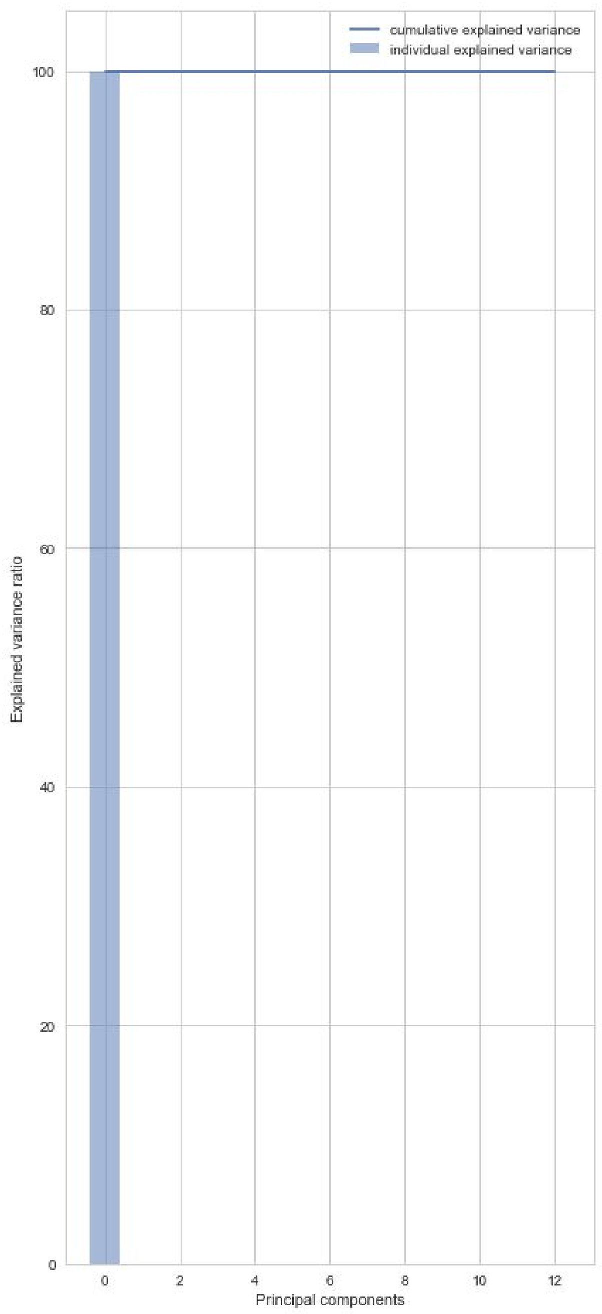

Figure 14 :cumulative sum of eigenvalues for wine data(LDA)

The first eigenvalue is very big, which weighs more than 99 percent of all eigenvalues. However, the LDA also lost a great amount of information and we cannot make decision how many vectors to use only by eigenvalues so we use all set of eigenvectors.

4. reconstruction error

Wine Dataset
The PCA reconstruction error (using 4 eigenvectors) is 6.73e-4.
The PCA reconstruction error (using all eigenvectors) is 4.51e-27.
the LDA reconstruction error is 1177.33156035

MNIST Dataset

I operate PCA on each 0, 1, 3, 5 four datasets separately and reconstruct the original value using 20th dominant PCA eigenvectors.

I run LDA on 0-1, 3-5 two datasets and reconstruct the original value using 20th dominant LDA eigenvectors.

PCA
The first plot is the original data. The 2nd plot is the reconstruction.The 3rd plot is the reconstruction error.

0 dataset

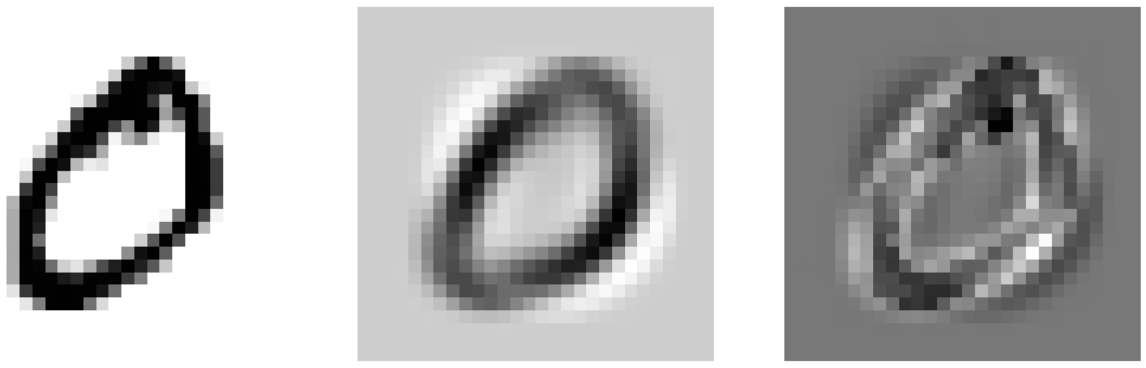

Figure 15: Reconstruction error 0

1 dataset

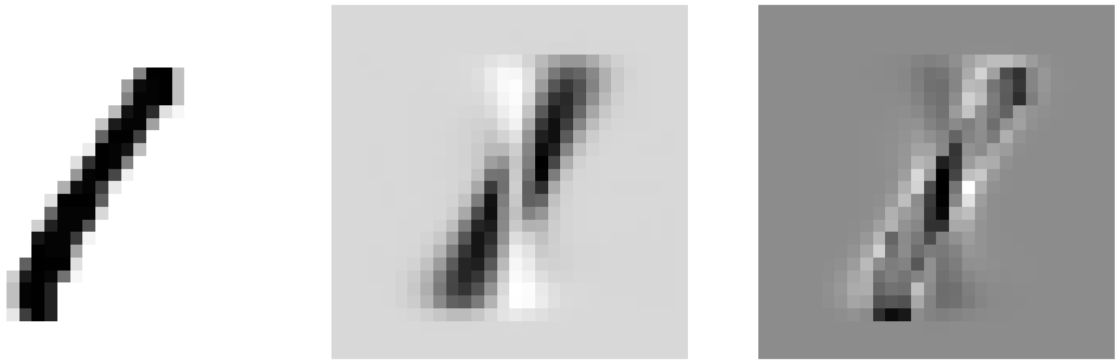

Figure 16. Reconstruction error 1

3 dataset

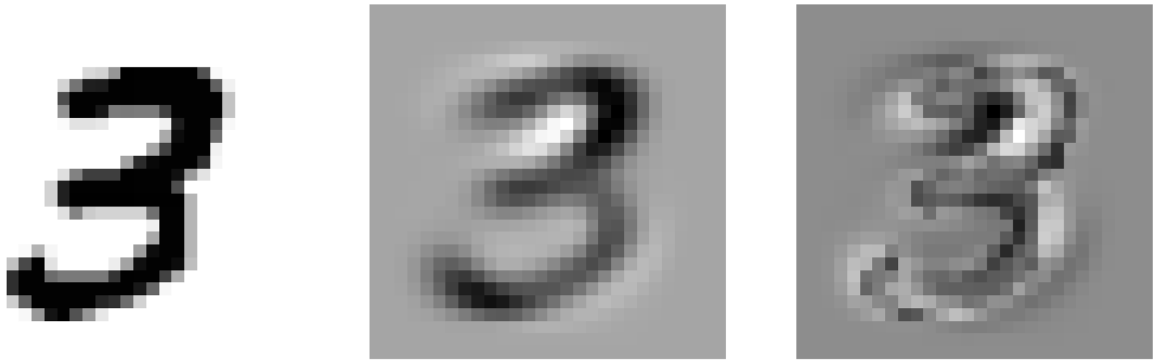

Figure 17. Reconstruction error 3

5 dataset

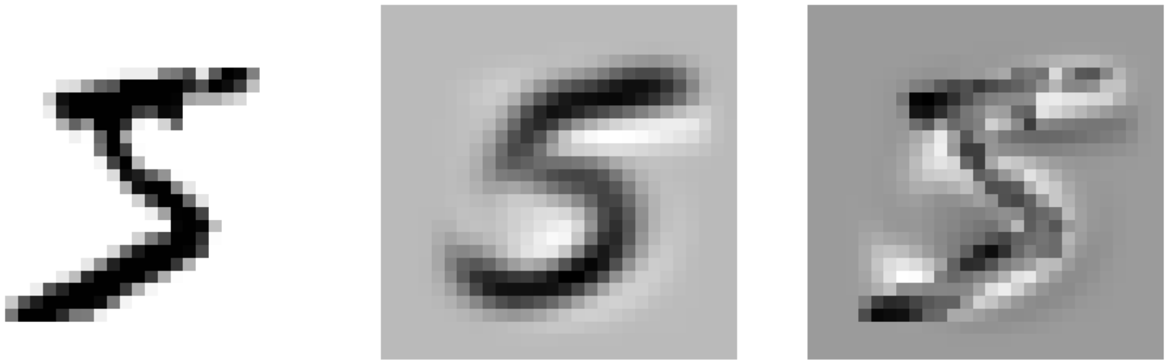

Figure 18. Reconstruction error 5

LDA

1st. Original figure; 2nd, projected; 3rd reconstructed; 4th difference

0,1 dataset

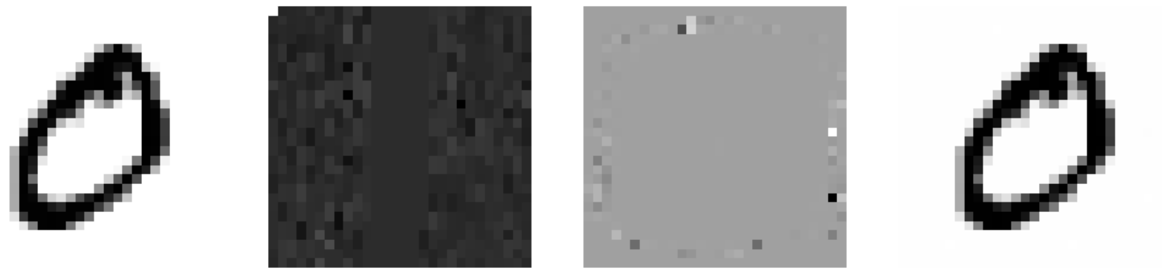

Figure 19. LDA reconstruction 0

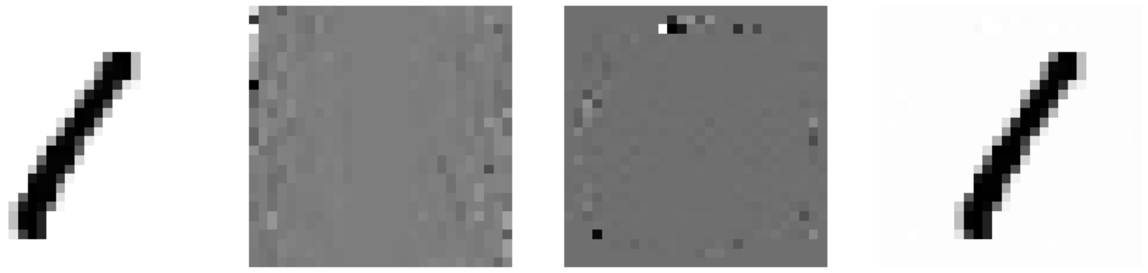

Figure 20. LDA reconstruction 1

3,5 dataset

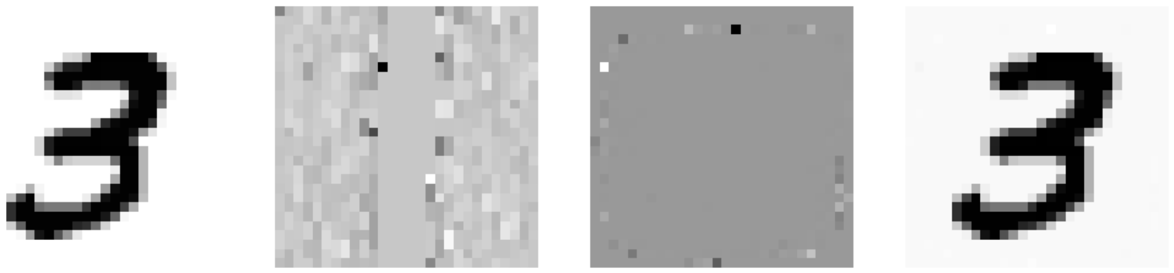

Figure 21. LDA reconstruction 3

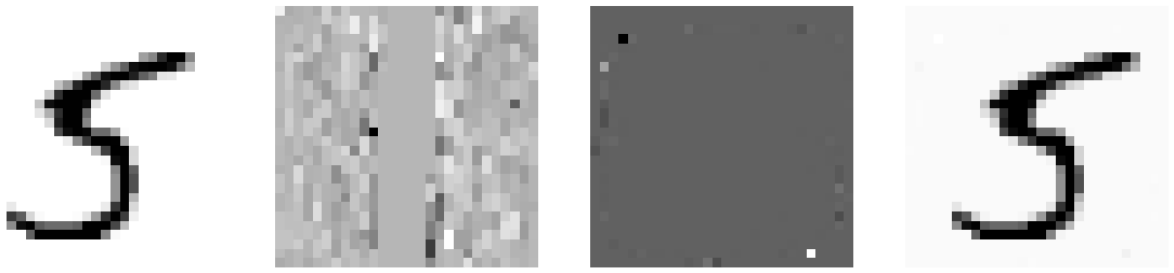

Figure 22. LDA reconstruction 5

We can see that for PCA, only using first 4 vectors can make great reconstruction. This prove that they contains great amount of information of original data.

For LDA, it is hard to retrieve original data because malfunction with a wierd Sigma calculation and information losses.

5. Confusion matrix

LDA

The confusion matrix for Wine dataset

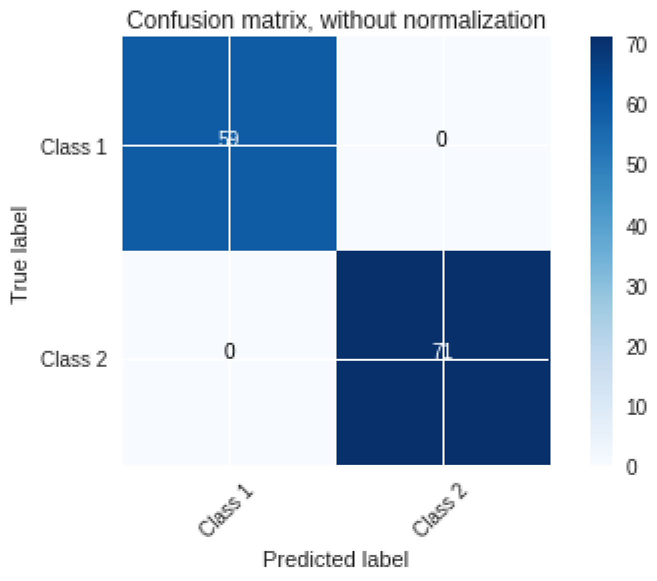

Figure 23: confusion matrix for wine set

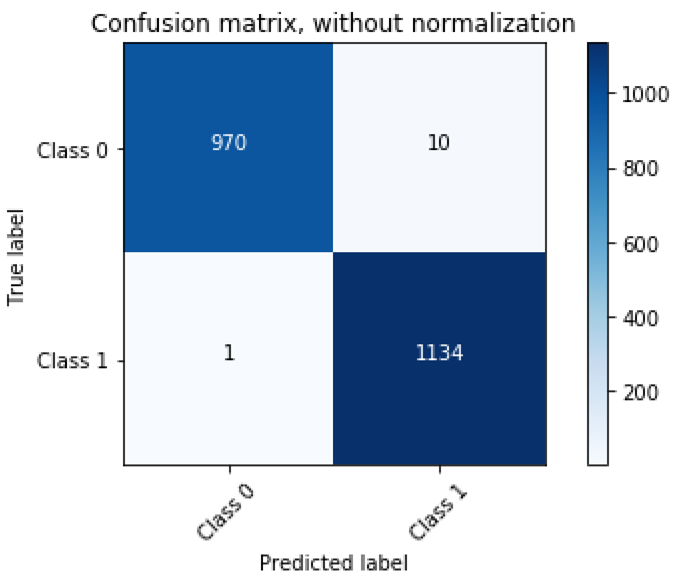

Figure 24: confusion matrix for 0-1 set

Confusion_matrix for 3, 5 testset

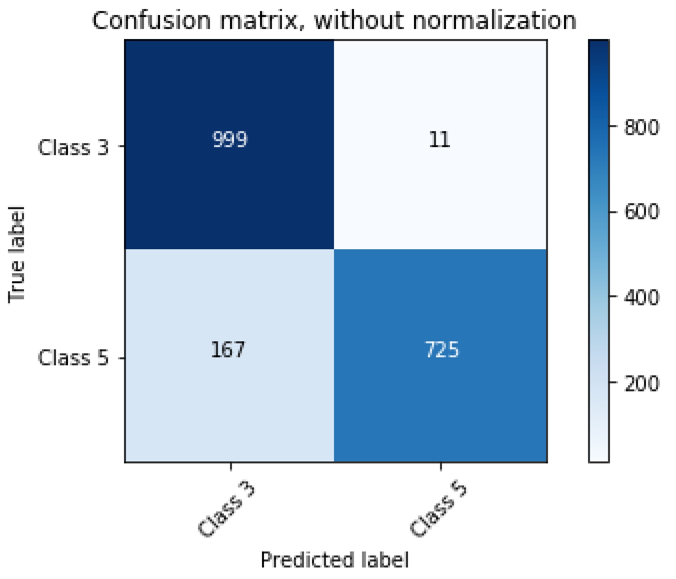

Figure 25: confusion matrix for 3-5 set

NaiveBayes

Confusion Matrix for wine classify.

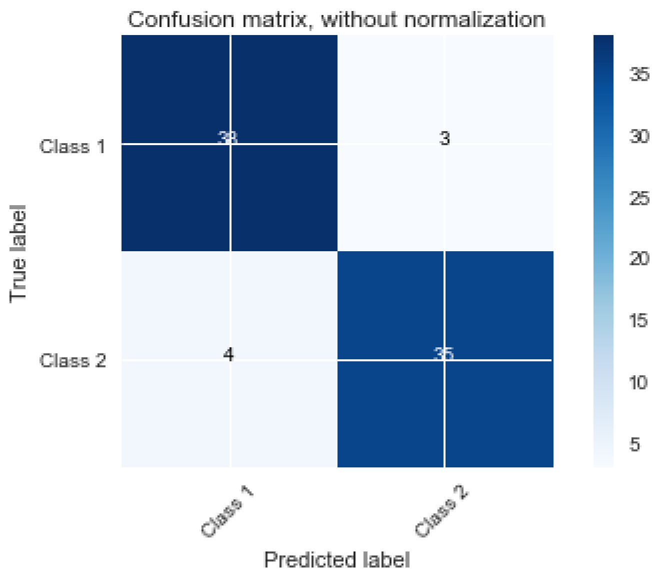

Figure 26: confusion matrix for wine set

Confusion_matrix for 0, 1 testset

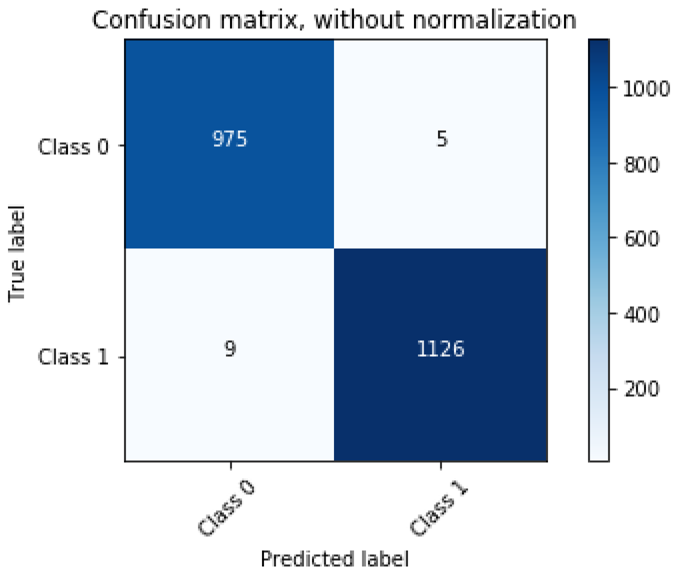

Figure 27: confusion matrix for 0-1 set

Confusion_matrix for 3, 5 testset

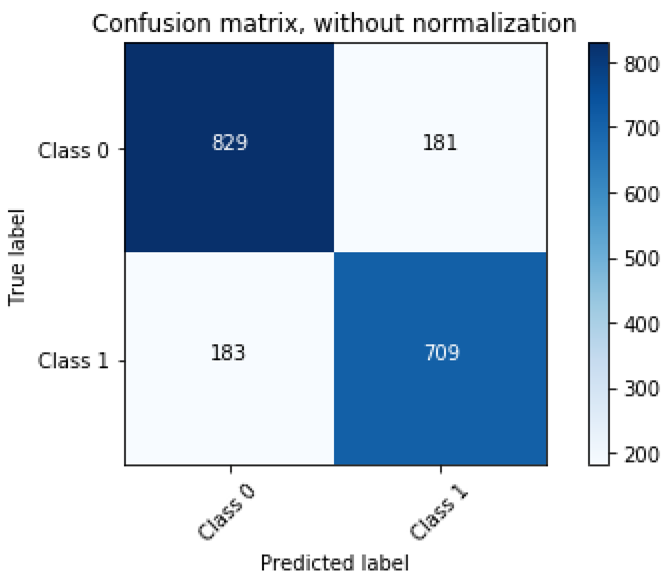

Figure 28: confusion matrix for 3-5 set

6. Accurate table

LDA
Accurate table for wine dataset

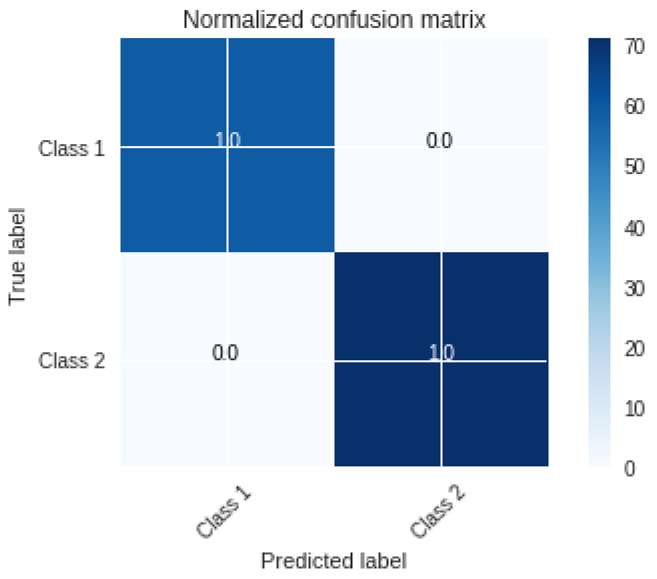

Figure 29: Accurate table for wine dataset

Accurate table for 0, 1 testset

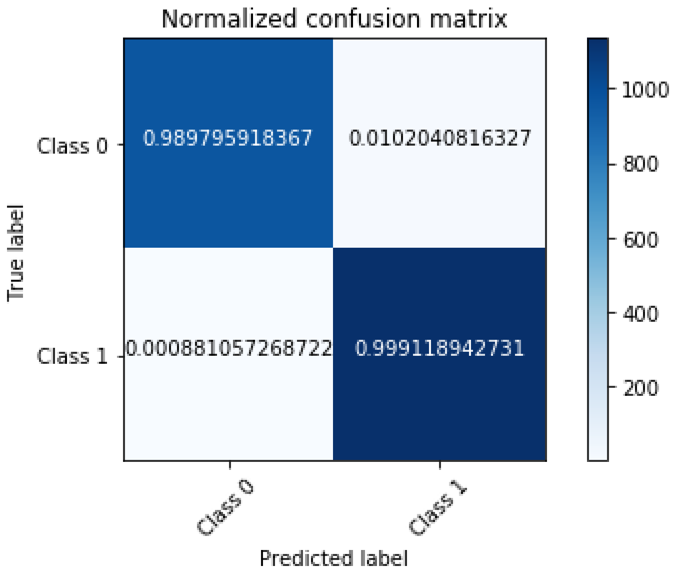

Figure 30: Accurate table for 0-1 dataset

Accurate table for 3, 5 testset

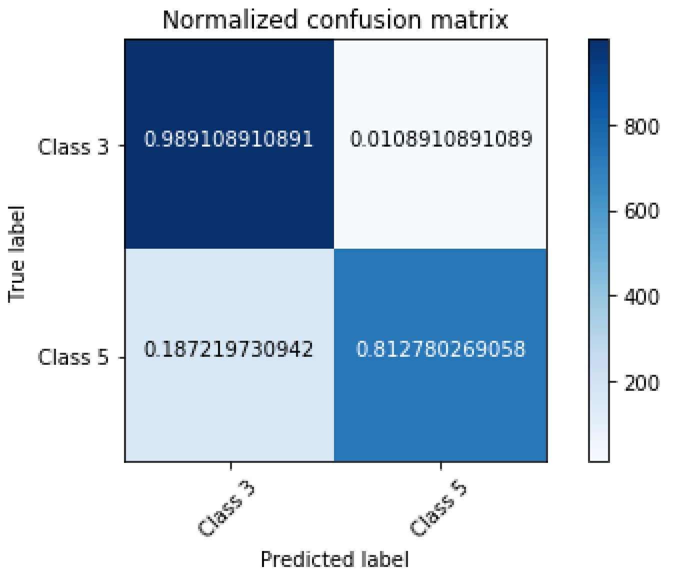

Figure 31: Accurate table for 3-5 dataset

NaiveBayes

Accuracy table for Wine classify

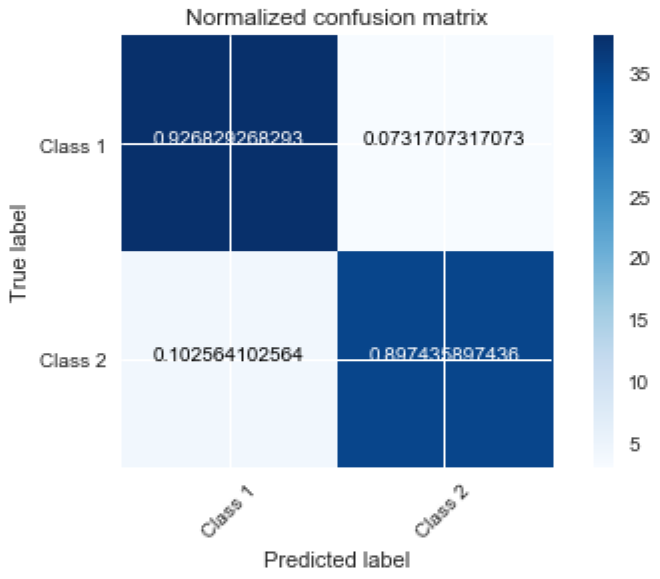

Figure 32: Accurate table for wine dataset

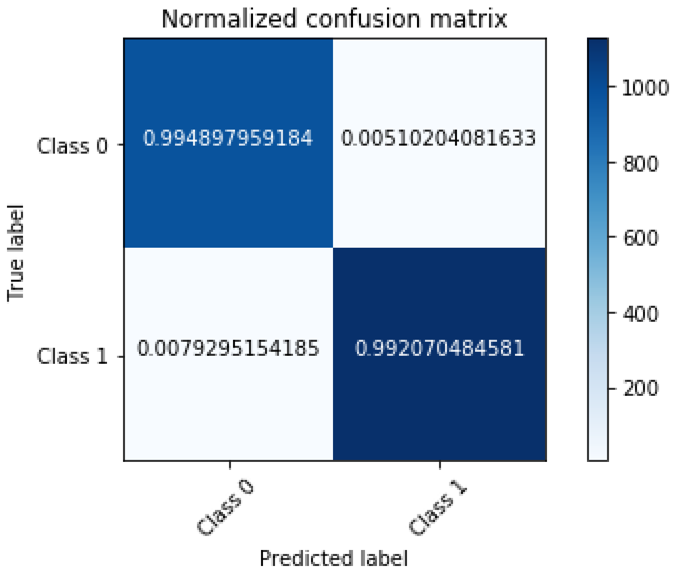

Figure 33: Accurate table for 0-1 dataset

Accurate table for 3, 5 testset

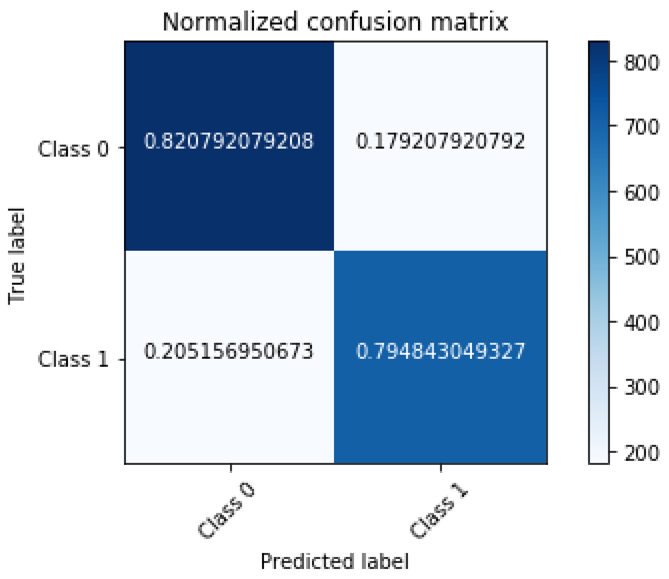

Figure 34: Accurate table for 3-5 dataset

Techniques comparison:

In this case, we use 0-1 and 3-5 MNIST dataset, the error rate of LDA is less than the Naive Bayes. The Naive Bayes assumes that all the dimensions are independent with each other. However, in real life, it is very usual that features are interdependent, so we have to select features before using the NB. For LDA, it can take care for the correlation and only assume the gaussian distribution.

Compare the 0,1 and 3,5 MNIST test cases

We can see that the accuracy of both LDA and Naive Bayes classifiers for 0, 1 dataset is greater than 3,5 dataset. I hypotheses that it easier to classify the 0-1 than to classify 3-5. For Naive Bayes, it takes care of the key pixels (dimensions) that have great difference between 2 cases. Therefore, we can see great overlap from 3 and 5 and very little overlap from 0 and 1. Therefore, there are less informative (helpful to classify) pixels in 3,5 case and much vulnerable to outliers. For LDA approach, it find a dimension to maximize the separation of 2 classes and the first eigenvector is used to project 31D vectors to 1D. If there are great overlaps between 2 cases, it also makes it harder to find direction that

Cross validation

The method of cross validation is to divide the training data into several folds and choose one of them as the validation fold and the others are training folds. Running Naive Bayes method before we actually using the test data. The reason to do crossing validation is to test how to the training data is and will be able to get rid some bad training data or hyperparameter.

For Wine data, there are 130 data total for class 1 and class 2. I random order the data to make the cross validation make sense. Then I divide the data into 10 folds, which means the size of the validation fold will be 13 and the size of the training folds will be 130 - 13 = 117.

```
The confusion matrix is:
[[7 0]
 [1 5]]
The confusion matrix is:
[[6 0]
 [2 5]]
The confusion matrix is:
[[6 0]
 [1 6]]
The confusion matrix is:
[[4 0]
 [0 9]]
The confusion matrix is:
[[6 0]
 [0 7]]
The confusion matrix is:
[[8 0]
 [1 4]]
The confusion matrix is:
[[ 3  0]
 [ 0 10]]
The confusion matrix is:
[[7 0]
 [0 6]]
The confusion matrix is:
[[5 1]
 [0 7]]
The confusion matrix is:
[[6 0]
 [0 7]]
```

The result of the cross validation for wine data set is pretty good. From the confusion matrix, we can see most of the validation sets are very close to the predicted result. At most 2 out of 13 data are not right during the cross validation, which means the training data is good.

For 0-1 data set, the result of cross validation contains two parts, one part is the accuracy of the predicted result and the second part is the confusion matrix.

```
The accuracy is:
0.98816101026
[[565  13]
 [  2 687]]
The accuracy is:
0.988950276243
[[577  14]
 [  0 676]]
The accuracy is:
0.989739542226
[[557  12]
 [  1 697]]
The accuracy is:
0.989739542226
[[602  13]
 [  0 652]]
The accuracy is:
0.988950276243
[[558  14]
 [  0 695]]
The accuracy is:
0.993680884676
[[593   8]
 [  0 665]]
The accuracy is:
0.992890995261
[[576   8]
 [  1 681]]
The accuracy is:
0.990521327014
[[593  11]
 [  1 661]]
The accuracy is:
0.99131121643
[[607  11]
 [  0 648]]
The accuracy is:
0.987361769352
[[576  15]
 [  1 674]]
```
I divide the training set into 10 folds as well. We can see from the confusion matrix for each validation folds, the accuracy is very high, which is exactly what we expect because 0 and 1 are very easy to distinguish.

For 3-5 data set, the result of cross validation is:

```
The accuracy is:
0.867647058824
[[537  71]
 [ 82 466]]
The accuracy is:
0.852076124567
[[540  85]
 [ 86 445]]
The accuracy is:
0.861471861472
[[532  92]
 [ 68 463]]
The accuracy is:
0.858874458874
[[548  79]
 [ 84 444]]
The accuracy is:
0.862337662338
[[562  79]
 [ 80 434]]
The accuracy is:
0.881385281385
[[526  79]
 [ 58 492]]
The accuracy is:
0.87619047619
[[527  72]
 [ 71 485]]
The accuracy is:
0.851948051948
[[513  83]
 [ 88 471]]
The accuracy is:
0.867532467532
[[530  75]
 [ 78 472]]
The accuracy is:
0.861471861472
[[521  80]
 [ 80 474]]
```

From the result of confusion matrix, we can see the accuracy is about 85 percent. The result is not as ideal as 0-1 data is. It also makes sense because 3 and 5 are much more similar compared to 0 and 1, making it more difficult to tell the difference.

For more classes

When we have more than 2 classes, PCA method is the same as 2 classes. When we implement the LDA method, we should use have more than 1 eigenvector as our w. In addition, we need to calculate the variance for each class. For naive bayes, when we calculate the posterior probability, we should calculate the posterior probability for each class and pick the largest one to be our prediction.