Introduction

This project requires the use of learned statistics to process real world data. It requires first performing principal component analysis on the dataset, then estimating the parameters of the reduced dimensionality samples, and finally calculating the error rate using the estimated parameters.

The dataset used in this project comes from images (with modifications) from the Fashion MNIST dataset. This project only uses images for class “T-shirt” and class “Sneaker”. There are 6000 T-shirt images and 6000 Sneaker images in the training dataset and 1000 T-shirt and 1000 Sneaker images in the testing dataset. T-shirt is labeled in “0” and Sneaker is labeled in “1”. Each image is 28*28 pixels.

Method

After I loaded the data, I reshaped each sample from the original 28*28 matrix into a 784-dimensional vector and calculated the sample mean and variance vectors for the training set. Then I normalize the sample set.

Next, I used **np.cov()** to get the dataset covariance matrix and used **np.linalg.eig()** to get the eigenvectors. I used the first and second eigenvectors as the first and second principal components. After that I used **np.linalg.norm()** to standardize principal components.

I then projected the normalized samples in the direction of the principal components and obtained their position in the new coordinate system. I visualized the training samples in this 2-d space in scatterplot and each of their classes looks like a normal distribution.

So I used MLE to estimate mean and variance for each class. I directly used the conclusion that the estimated mean for data in normal distribution is equal to its average and estimated variance is equal to data’s covariance matrix.

Finally I use the obtained parameters to calculate the error rate. For each of the data in the training and test sets, after normalizing them and projecting them in the direction of the principal components, I used the resulting mean and variance to calculate the likelihood p(x|wi). Since the prior probability is the same for both categories, this is done directly by comparing the likelihood values. The judgment is then compared to ground truth, and if it is wrong then loss +1. Finally, I calculate the loss/total sample number to get the minimum error rate.

Result and Observation

Part 1:

```
Part 1, mean vector =  [6.66666667e-04 9.75000000e-03 7.38333333e-02 1.70416667e-01
 2.27333333e-01 3.46833333e-01 7.92666667e-01 2.78008333e+00
 7.73425000e+00 1.65732500e+01 2.91936667e+01 3.82306667e+01
 3.25500000e+01 2.53065000e+01 2.44523333e+01 2.88430000e+01
 3.91677500e+01 3.73315833e+01 2.37889167e+01 1.19668333e+01
 4.83641667e+00 1.44200000e+00 4.58666667e-01 2.89000000e-01
 1.66500000e-01 6.72500000e-02 7.41666667e-03 1.10000000e-02
 3.66666667e-03 1.30000000e-02 1.50416667e-01 4.63916667e-01
 7.49416667e-01 3.23333333e+00 1.33013333e+01 3.08098333e+01
 4.87553333e+01 6.28153333e+01 7.19932500e+01 8.21180000e+01
 8.60692500e+01 8.25206667e+01 8.12066667e+01 8.67872500e+01
 8.76591667e+01 7.76893333e+01 6.88405000e+01 5.80794167e+01
 4.14480000e+01 2.17570000e+01 6.93800000e+00 1.65866667e+00
 6.75250000e-01 2.66500000e-01 6.05833333e-02 3.98333333e-02
 7.41666667e-03 2.63333333e-02 2.21500000e-01 8.14750000e-01
```

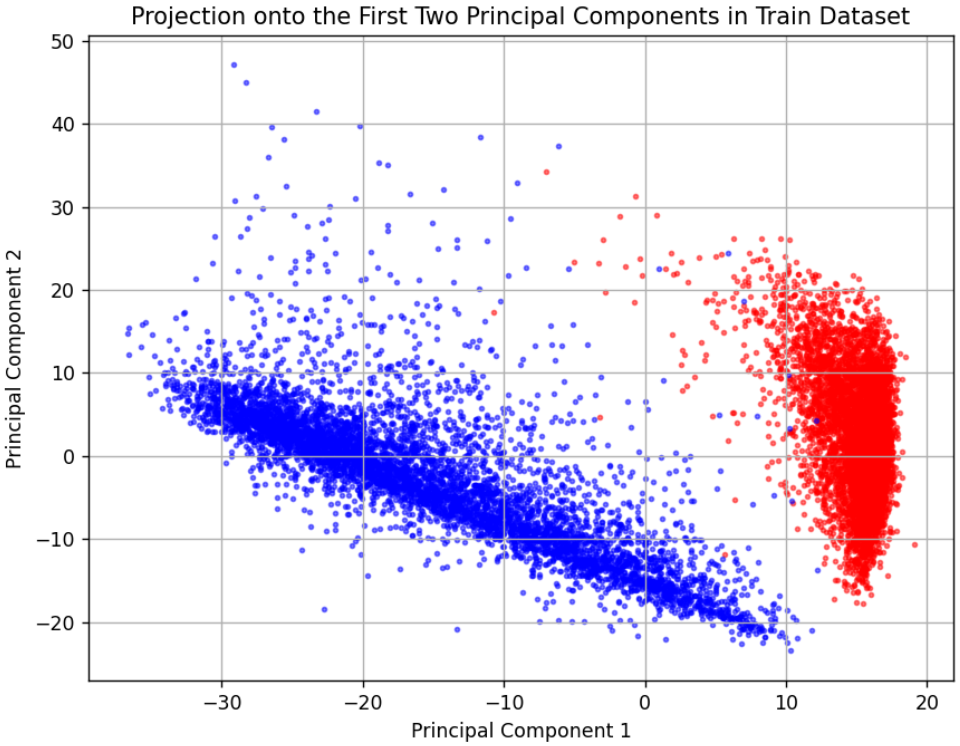
```
Part 1, std vector =  [6.45462797e-02 2.88365978e-01 1.23688398e+00 2.75935527e+00
 3.51178771e+00 4.48339975e+00 7.26844868e+00 1.52362688e+01
 2.76626396e+01 4.07821076e+01 5.41848502e+01 6.30068988e+01
 5.84347342e+01 5.08006535e+01 5.02219148e+01 5.49283080e+01
 6.48622973e+01 6.27247450e+01 4.96575996e+01 3.46353923e+01
 2.10666242e+01 1.09055553e+01 5.22305385e+00 3.98007274e+00
 3.69796490e+00 1.98012477e+00 2.39850633e-01 1.10681781e+00
 1.96179906e-01 3.13577742e-01 3.89342927e+00 7.44102578e+00
 9.97581866e+00 1.88409409e+01 3.94661230e+01 6.06645902e+01
 7.66881388e+01 8.78430631e+01 9.32775207e+01 9.87127866e+01
 1.03287511e+02 1.01208986e+02 1.00502772e+02 1.04424577e+02
 1.03655559e+02 9.65387961e+01 9.17958399e+01 8.43236925e+01
 7.07309607e+01 5.13904283e+01 2.87545850e+01 1.47036897e+01
 9.83806150e+00 5.86301780e+00 2.69587580e+00 2.50634129e+00
 2.10622078e-01 9.66681552e-01 5.32383988e+00 1.01441083e+01
 2.08945840e+01 5.08803576e+01 7.57228699e+01 8.44883153e+01
 8.72591477e+01 8.83335320e+01 9.02848412e+01 9.22603376e+01
 9.51761438e+01 9.53659693e+01 9.55082867e+01 9.67022076e+01
 9.35707707e+01 9.07463621e+01 8.91316307e+01 8.79485372e+01
 8.66088911e+01 8.22807669e+01 6.53836830e+01 3.49719099e+01
```

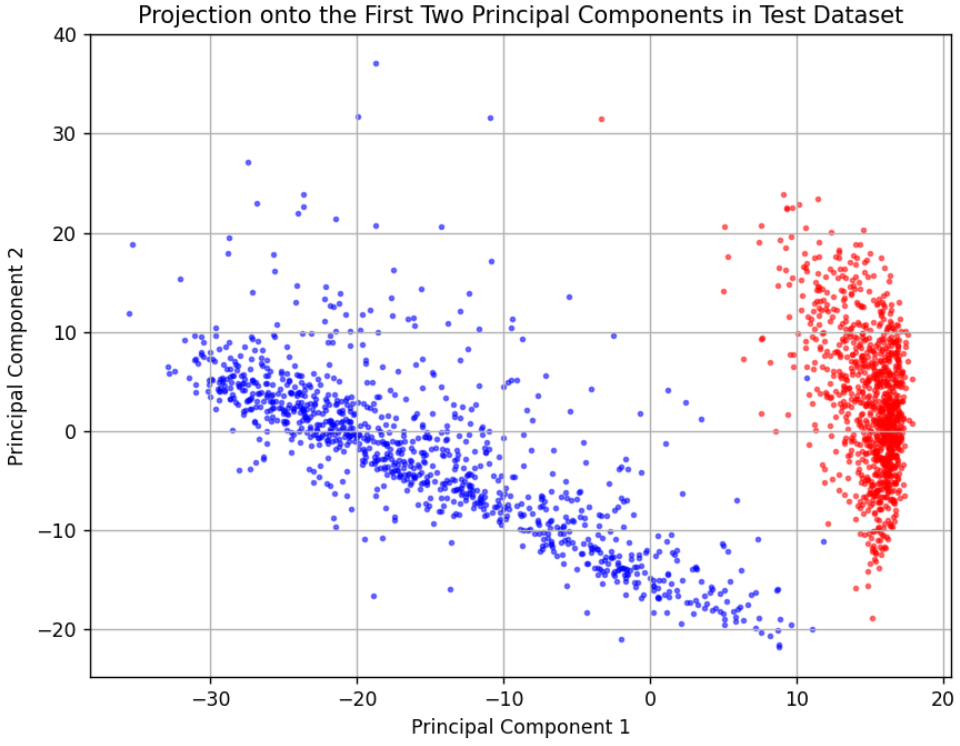
Part 2:

```
============================================================
First PCA:  [-6.64461141e-04 -2.10239358e-03 -3.72519317e-03 -3.29025897e-03
 -4.01885380e-03 -4.96809009e-03 -6.35460152e-03 -1.01534045e-02
 -1.64850021e-02 -2.41267167e-02 -3.22589368e-02 -3.61794719e-02
 -3.29855480e-02 -2.98896816e-02 -2.92143258e-02 -3.12336583e-02
 -3.51197480e-02 -3.49700847e-02 -2.83077474e-02 -2.01043561e-02
 -1.26913609e-02 -7.07754442e-03 -5.00744050e-03 -3.90573083e-03
 -2.65044469e-03 -1.53462408e-03 -1.49134011e-03 -2.75162426e-04
 -5.43098285e-04 -2.22896564e-03 -2.28417575e-03 -4.04821449e-03
 -5.10328585e-03 -1.12836135e-02 -2.21666963e-02 -3.33053800e-02
 -4.20871667e-02 -4.72122783e-02 -4.99110649e-02 -5.17274921e-02
 -5.02451414e-02 -4.79064868e-02 -4.68806977e-02 -4.90809960e-02
 -5.08006516e-02 -5.06108897e-02 -4.88629329e-02 -4.53594349e-02
```

```
Second PCA: [ 2.65302063e-04  5.24337961e-03  5.53663175e-03  4.33290888e-03
  6.64708440e-03  5.65805224e-03  4.76100149e-03  2.31190846e-03
  2.18227454e-04 -9.48351797e-03 -2.08669002e-02 -2.68049906e-02
 -2.37244946e-02 -1.94199903e-02 -1.85653162e-02 -2.14317786e-02
 -2.85795673e-02 -2.56994764e-02 -1.45396953e-02 -4.30601053e-03
  1.19669468e-03  3.65954115e-03  4.25778360e-03  4.21387901e-03
  6.52520594e-03  4.08876080e-03  3.38069453e-04  4.89777315e-03
  1.59128192e-03  5.14883157e-03  7.08527307e-03  9.24331382e-03
  1.06499799e-02  6.39207411e-03  9.62553250e-04 -3.65532805e-03
 -6.22977022e-03 -9.47465608e-03 -1.40809685e-02 -2.19096287e-02
 -2.74796979e-02 -3.07504609e-02 -3.20491306e-02 -3.03713013e-02
 -2.80041767e-02 -1.80838263e-02 -1.05648573e-02 -7.56323561e-03
 -5.94273633e-03 -2.90920151e-03  3.83932834e-03  8.57972462e-03
```

Part 3:


Projection onto the First Two Principal Components in Train Dataset
Principal Component 2
Principal Component 1
50
40
30
20
10
0
−10
−20
−30
−20
−10
0
10
20


Projection onto the First Two Principal Components in Test Dataset
Principal Component 2
Principal Component 1
40
30
20
10
0
−10
−20
−30
−20
−10
0
10
20

Part 4 and 5:

```
============================================================
Estimated mean of T shirt:  [-14.96013579  -2.53759713]
Estimated cov of T shirt:  [[103.38209995 -69.15661464]
 [-69.15661464  73.43418283]]
Estimated mean of Sneaker:  [14.96013579  2.53759713]
Estimated cov of Sneaker:  [[ 4.94656978 -6.78163691]
 [-6.78163691 54.67473161]]
============================================================
Error Rate in Training Dataset:  0.0025
Error Rate in Testing Dataset:  0.001
PS E:\workplace\CSE-569>
```

Conclusion

This project was very successful in performing principal component analysis on the training set and estimating the parameter of the projection in the direction of the principal components. And the error rate of the final speculation on the samples of the test set is very small.