HW-2: Linear Models for Classification

Final Report

- Abstract

In this homework, we must implement Principal components analysis (PCA) to reduce the number of dimension of our input data. After PCA, we perform linear models of classification using Probabilistic Generative Models and Probabilistic Discriminative Models to predict the belonging class of input data.

- Principal Components Analysis

Considering data sample of a vector of length N. After getting M measurements, we would obtain the matrix of MN elements, which grows large and brings inconvenience when M or N is too large. Here comes out a technique called Principal Components Analysis to overcome this potential problem.

With such a structure, measuring the mean and variance of each sample and also covariance between sample point can help us to generate a new basis which can represent these most of the variance without using too many components and memory.

Suppose in our training case, we have 2700 images with size 30 x 30. We then form a 2700 x 900 matrix X to store it (M=).

$$\mathrm{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{2700} \end{bmatrix}$$

where $x_1,\ldots, x_{2700}$are row vector of a sample. Then, compute the mean vector of each element of row vector.

$$\mu_x = \frac{1}{2700}\sum_{n=1}^{2700} x_n$$

Next, we recenter our data to zero mean for later calculation.

$$\tilde{\mathrm{X}} = \begin{bmatrix} x_1 - \mu_x \\ x_2 - \mu_x \\ x_3 - \mu_x \\ \vdots \\ x_{2700} - \mu_x \end{bmatrix}$$

Then, do column normalization. With zero mean, we can obtain the covariance matrix R=$\tilde{X}^H\tilde{X}$, where H means conjugate transpose. Since R is always symmetric in our case, the SVD decomposition will have the form R= VDV^{H}, where V is the eigenvector matrix of R and D is a diagonal matrix of singular value. Luckily, matlab will sort these eigenvalue with decreasing order. Thus, We can easily obtain the first two largest direction of variance to reduce our 2700 x 900 matrix into 2700 x 2 matrix. After getting two effectively basis, we perform projection to our original data into new basis.

$$X_{pca} = V^{H}\tilde{X}_{orig}$$

- Probabilistic Generative Models

  In our class, we have more than 2 classes. We have the posterior probability with the form

$$\mathrm{p}(C_k|\mathrm{x}) = \frac{p(x|C_k)p(C_k)}{\sum_j p(x|C_j)p(C_j)}$$

$$= \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

where is known as the normalized exponential or softmax function to make bigger the difference between a and can be regarded as a multiclass generalization of the logistic sigmoid.

Next, we assume that the class-conditional densities are Gaussian and find that the density of class C_k sharing the same covariance matrix is given by

$$\mathrm{p}(\mathrm{x}|C_k) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\Sigma|^{1/2}}exp\left\{-\frac{1}{2}(x-\mu_x)^T\Sigma^{-1}(x-\mu_x)\right\}$$

Then, we may have $a_k(x)$=$\mathrm{w}_k^T x + \omega_{k0}$, where we define

$$w_k = \Sigma^{-1}\mu_k$$

$$\omega_{k0}\text{=-}\frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \ln(p(C_k))$$

The resulting decision boundaries, corresponding to the minimum misclassification rate, will occur when two posterior probabilities equal.

- Probabilistic Discriminative Models

  An alternative method based on Probabilistic Generative Models is to use functional form of the generalized linear model and to determine its parameter directly by using maximum likelihood. We shall see there is an efficient algorithm finding such solutions known as iterative reweighted least square.

$$\mathrm{p}(C_K|\varphi) = y_k(\varphi)\text{=} \frac{\exp(a_k)}{\sum_j \exp(a_j)}$$

where $a_k = w_k^T\varphi$. In our case, I define φ as x. The likelihood function is then given by

$$\mathrm{p}(\mathrm{T}|w_1,\ldots,w_k) = \prod_{n=1}^{N}\prod_{k=1}^{K} y_{nk}^{t_{nk}}$$

where T is an N x K matrix of target variable with elements t_{nk}.

After getting likelihood function, we then take the negative logarithm and take the gradient of the error function with respect to one of the parameter vectors w_j.

$$\mathrm{E}(w_1,\ldots,w_k) = -\ln p(\mathrm{T}|w_1,\ldots,w_k) = -\sum_{n=1}^{N}\sum_{K=1}^{K} t_{nk}\ln y_{nk}$$

$$\nabla_{w_j}\mathrm{E}(w_1,\ldots,w_k)=\sum_{n=1}^{N}(y_{nj}-t_{nj})\varphi_n$$

To find a batch algorithm, we appeal to the Newton-Raphson update to obtain the corresponding IRIS algorithm and we make up Hessian matrix as follow.

$$\nabla_{w_k}\nabla_{w_j}\mathrm{E}(w_1,\ldots,w_k)=\sum_{n=1}^{N} y_{nk}(I_{nj}-y_{nj})\varphi_n\varphi_n^T$$

The Newton-Raphson, updated for the minimization of E(w), takes the form

$$w^{new} = w^{old} - H^{-1}\nabla E(w)$$

At last, we can use this convergent parameters w to do the further prediction of classification.

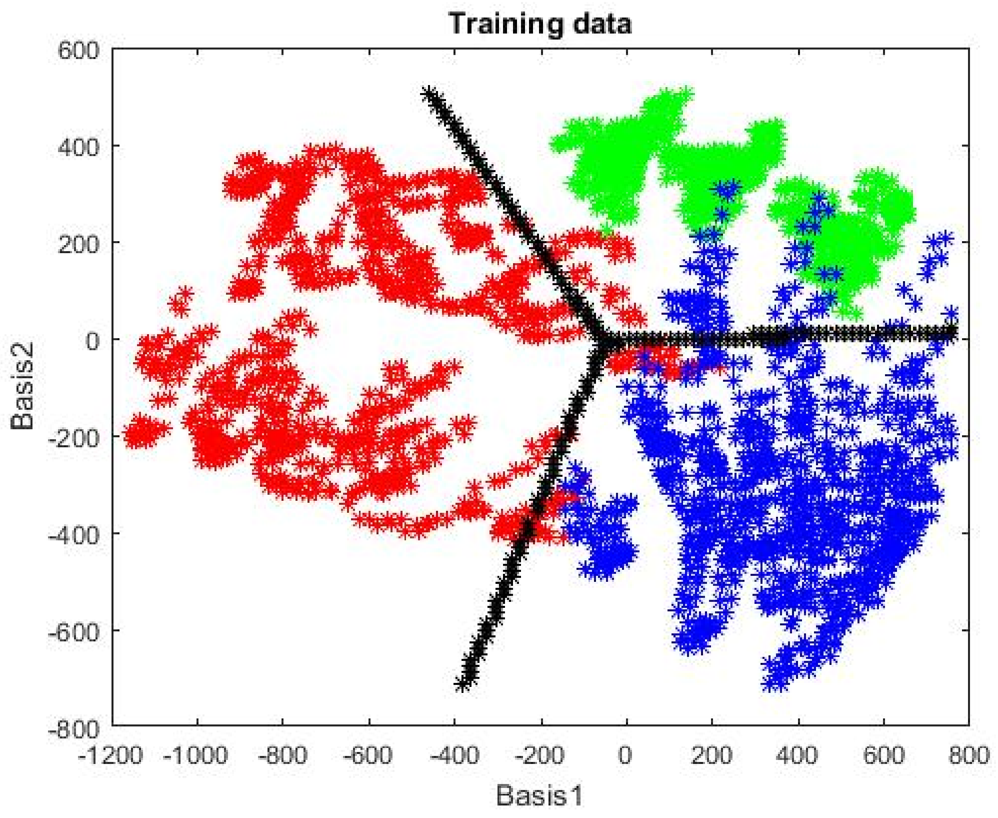
- Result

Probabilistic Generative Models:

- Balanced Data
  - Setting:

Stage	Class	Number of Images
Training	1	900
	2	900
	3	900
Testing	1	100
	2	100
	3	100

  - Decision boundary



- Error Rate

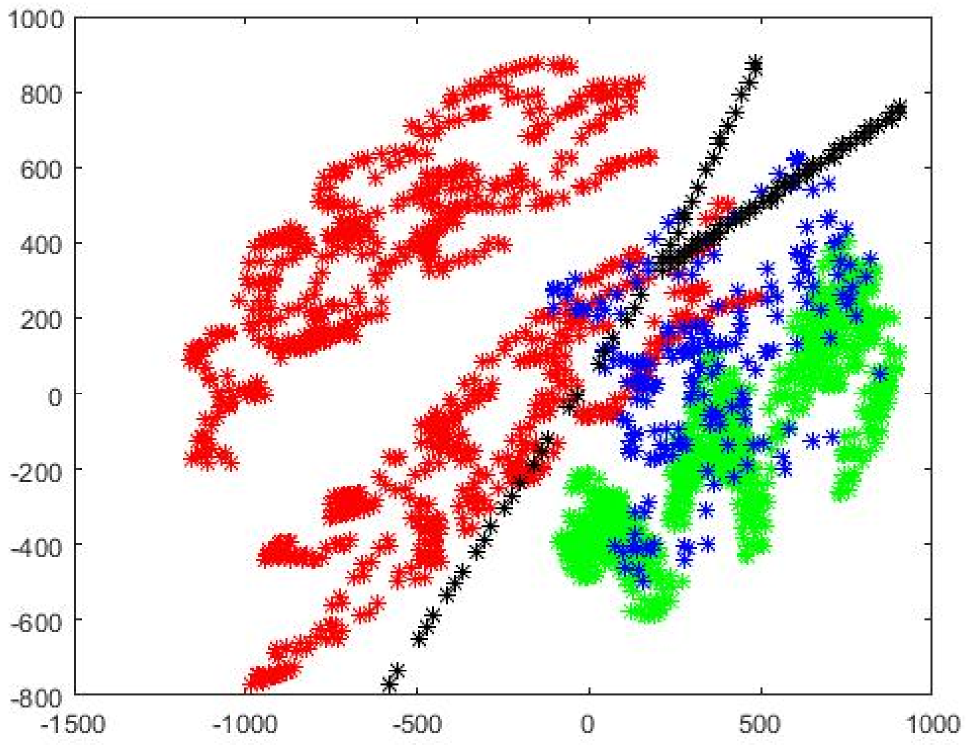
  (Due to use of random function in my training, I may run testing for several times to ensure the stable misclassification rate)

	1	2	3	4	5
Error Rate(%)	5.67	3.67	4.33	3.67	5.00

- Unbalanced Data
  - Setting

Stage	Class	Number of Images
Training	1	900
	2	900
	3	200
Testing	1	100
	2	100
	3	100

■ Decision boundary



■ Error Rate

(Due to use of random function in my training, I may run testing for several times to ensure the stable misclassification rate)

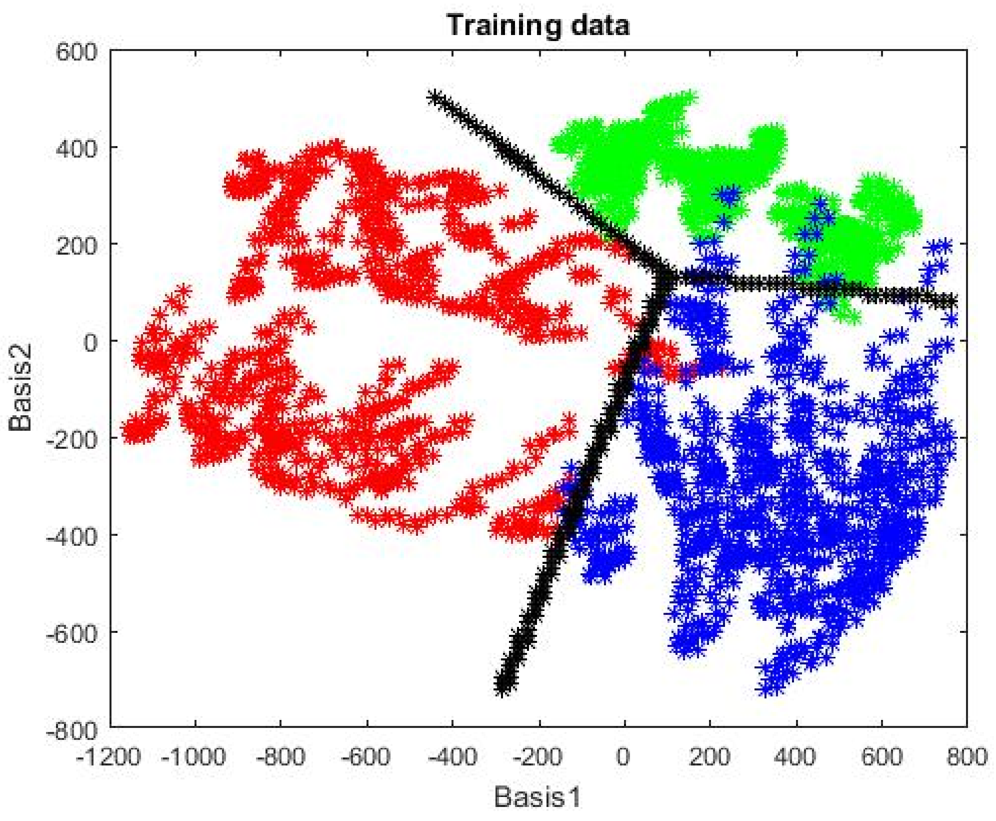
	1	2	3	4	5
Total Error Rate (%)	34.33	34.00	34.00	36.33	33.67
Misclassification Rate for Class 1 data (%)	6.00	8.00	6.00	11.00	8.00
Misclassification Rate for Class 2 data (%)	0.00	0.00	0.00	0.00	0.00
Misclassification Rate for Class 3 data (%)	97.00	94.00	96.00	98.00	93.00

Probabilistic Discriminative Models:

- Balanced Data
  - Setting:

Stage	Class	Number of Images
Training	1	900
	2	900
	3	900
Testing	1	100
	2	100
	3	100

  - Decision boundary



  - Error Rate

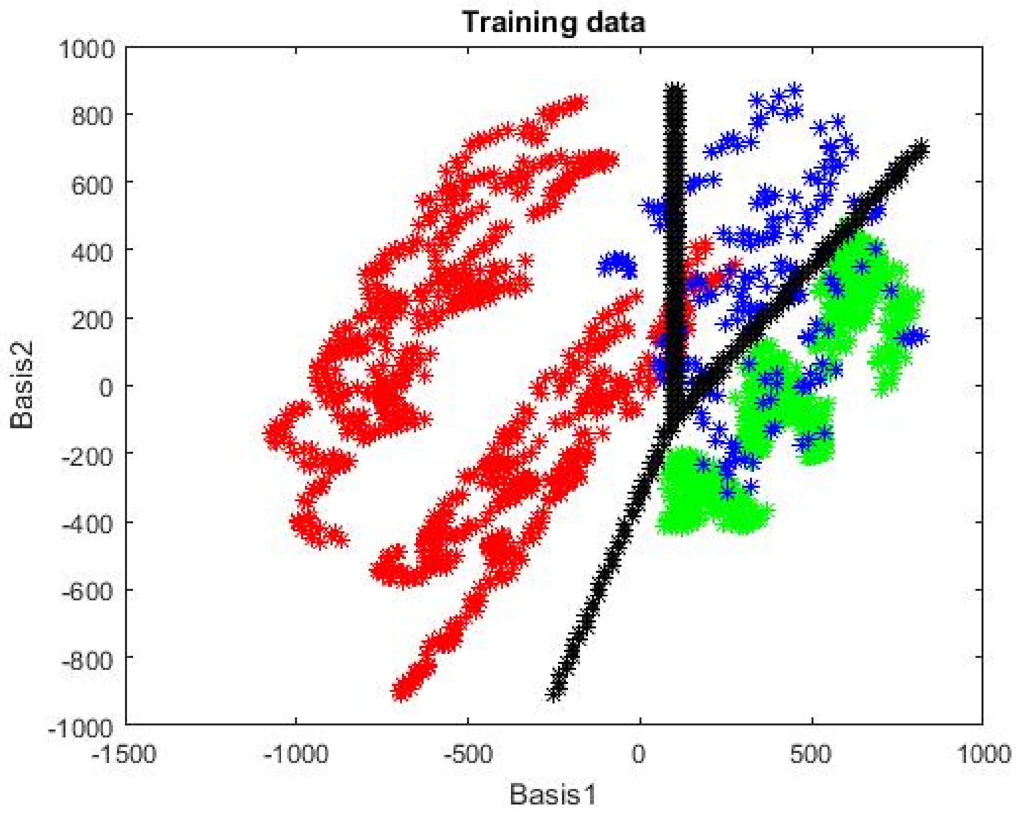
(Due to use of random function in my training, I may run testing for several times to ensure the stable misclassification rate)

	1	2	3	4	5
Error Rate(%)	1.67	2.00	3.00	3.00	2.00

- Unbalanced Data
  - Setting

Stage	Class	Number of Images
Training	1	900
	2	900
	3	200
Testing	1	100
	2	100
	3	100

- Decision boundary



- Error Rate

(Due to use of random function in my training, I may run testing for several times to ensure the stable misclassification rate)

	1	2	3	4	5
Total Error Rate (%)	14.00	14.67	16.00	14.33	15.33
Misclassification Rate for Class 1 data (%)	0.00	5.00	2.00	4.00	3.00

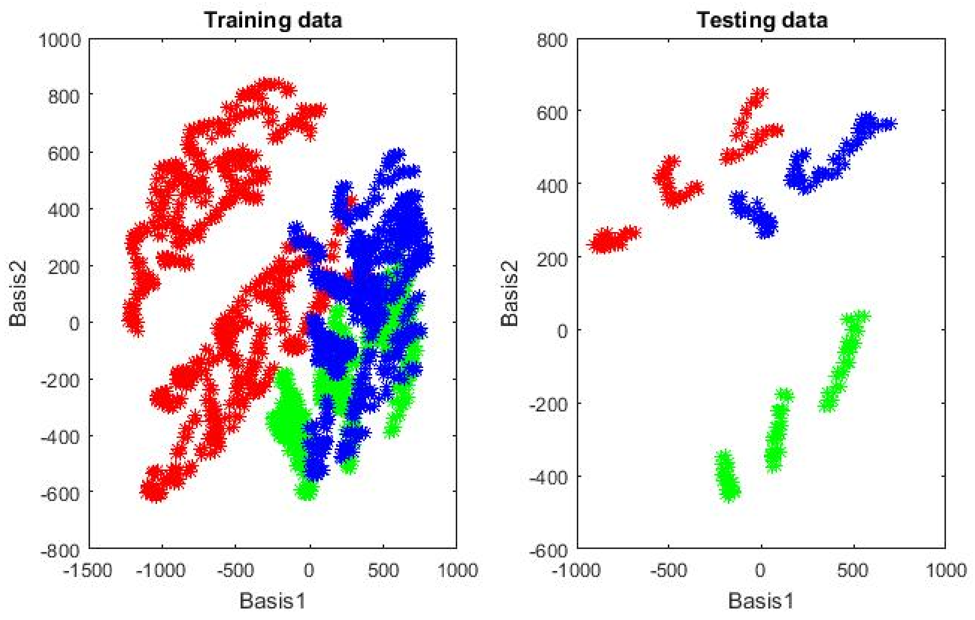
Misclassification Rate for Class 2 data (%)	1.00	0.00	0.00	0.00	2.00
Misclassification Rate for Class 3 data (%)	41.00	39.00	46.00	39.00	41.00

- Discussion
  - Balanced data set VS Unbalanced data set

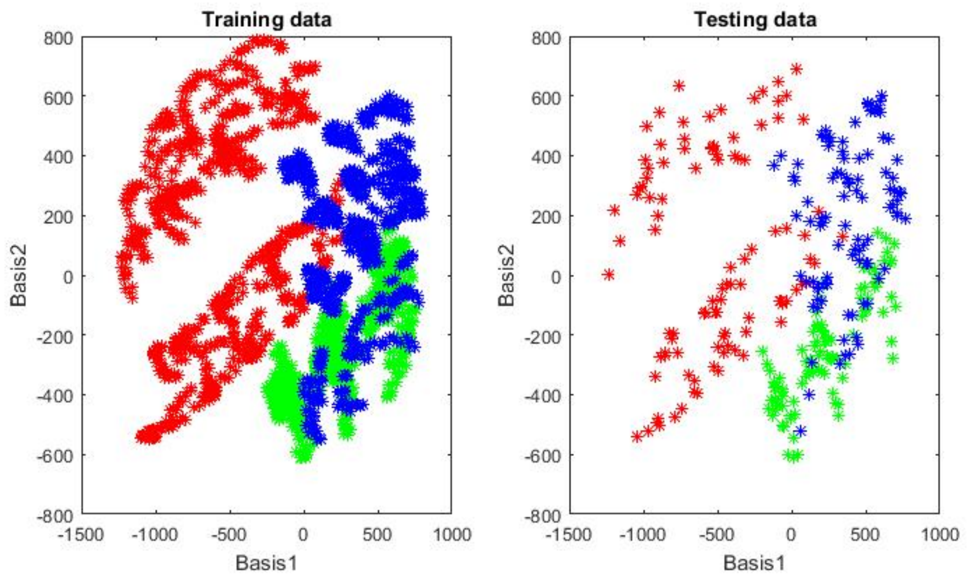
    If we make the training data unbalance, in other words, we decrease the number of data in certain class, we can find that the classification for the validation data in that class has really higher error rate. Also we can see the decision boundary is significantly changed. To overcome the Unbalanced data set problem in the future, we can use some technique to generate new data for that class to reach equal number of images in each class, such as transformed data.

  - Rearrange data

    Because in each class folder, the photo angle of each corresponding number are identical. However, if I assign the same part of data to be our training data, it would be possible to generate weak model which seems easily to classify but might have difficulty to make correct classification of other test data. The following figure demonstrate the features of figure in new two basis. We can see the test data is evidently separated in new coordinate. Thus, we may not test the model thoroughly.



Then, to overcome this problem, I rearrange with the training data in each class with matlab random function. We can see the following figure showing the features of testing data become more uniformly distributed.



■ Relationship bewteen number of dimension and error rate

As we can see, we have 900 features,or pixels, to represent one input image. Then, we use PCA algorithm to reduce its number of dimension. We guess that as the number of dimension decrease, the error rate will increase. The following table shows the relationship between the number of dimension and the classification error rate. If we use only one feature to represent one input 30 x 30 image, it would intuitively be hard to make correct classification. Thus, its error rate is bigger than those with higher number of dimension.

Number of Dimension	1	2	3	4	5
Average of Error rate (%)	31.53	4.47	2.33	1.8	0.4

■ Fisher's discriminant analysis (Bonus)

The following are the algorirhm I implement:

$$S_W = \sum_{k=1}^{K} S_K \quad \text{where} \quad S_k = \sum_{n \in C_k} (x_n - m_k)\,(x_n - m_k)^T$$

$$m_k = \frac{1}{N_k} \sum_{n \in C_k} x_n$$

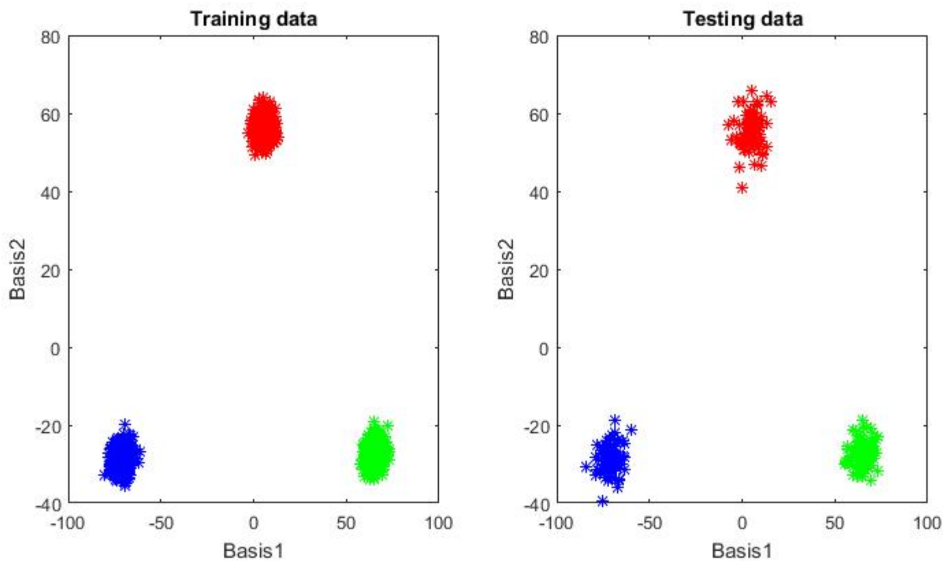
$$S_B = \sum_{k=1}^{K} N_k (m_k - m)(m_k - m)^T$$

$$J(w) = \frac{(WS_wW^T)}{(WS_BW^T)}$$

To maximize J(w) for maximization of the class separation and minimization of the class overlap, it's identical to solve eigenvalue and eigenvector for the following equation:

$$S_B w = \lambda\, S_W w$$

The following plot demonstrates the training data and the testing data with new basis vectors.



We can see the separation bewteen data of each class is bigger than Generative and Discriminative models. Thus, we can correctly classification for test data at almost 100% accuracy.

- Reference

[1] Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer, 2007

[2] University of Illinois, Urbana-Champaign ECE 311: Digital Signal Processing Lab 6 Tutorial, November 1,2016

[3] https://ccjou.wordpress.com/2014/03/14/費雪的判別分析與線性判別分析/