

Project 2: linear classification

Due on Friday, February 21, 2020 at 11.59PM

Name: Ke LIANG

ID: 926791183

email: kul660@psu.edu

contents

1 Introduction.....	2
2 Methods.....	2
2.1 Dataset.....	2
2.2 Linear Discriminate using Least Squares	3
2.3 Fisher's linear discriminant using KNN	3
3 Results.....	5
3.1 Tasks for Linear Discriminate using Least Squares	5
3.1.1 Results on Wine dataset	5
3.1.2 Results on Wallpaper Group dataset	8
3.1.3 Results on Taiji Pose dataset	10
3.2 Tasks for Fisher's linear discriminant using KNN.....	12
3.2.1 Results on Wine dataset	12
3.2.2 Results on Wallpaper Group dataset	14
3.2.3 Results on Taiji Pose dataset	16
4 Conclusions.....	18
Reference	19

1 Introduction

The goal of this project is to get you familiar with common classifiers and how to compare and contrast different classifiers.

The first is Linear Discriminate with least squares method for classification, and the second is Fisher projection with KNN classification.

The dataset we use is WINE, Wallpaper Group Dataset and Taiji Pose Dataset.

2 Methods

2.1 Dataset

The function “loadDataset.m” is used to get the dataset: wine, wallpaper, Taiji Pose Dataset.

As for **wine** dataset, there are 3 classes and 13 dimensions (feature) in the data set including (1) Alcohol, (2) Malic acid, (3) Ash, (4) Alcalinity of ash, (5) Magnesium, (6) Total phenols, (7) Flavanoids, (8) Nonflavanoid phenols, (9) Proanthocyanins, (10) Color intensity, (11) Hue, (12) OD280/OD315 of diluted wines and (13) Proline.^[1]

As for **Wallpaper Group** dataset, there are 17 classes and 500 dimensions (feature) in the data set including (1) P1 group, (2) P2 group, (3) PM group, (4) PG group, (5) CM group, (6) PMM group, (7) PMG group, (8) PGG group, (9) CMM group, (10) P4 group, (11) P4M group, (12) P4G group, (13) P3 group, (14) P3M1 group, (15) P31M group, (16) P6 group and (17) P6M group.^[2]

As for **Taiji Pose** dataset, there are 8 classes and 64 dimensions (feature) in the data set. This is a dataset of the joint angles (in quaternions) of 35 sequences from 4 people performing Taiji in our motion capture lab. You will classify which MoCAP frames are transitional frames 7 different poses (non '0' labels) and the non-transitional frames (the '0' labels).^[3]

2.2 Linear Discriminate using Least Squares

The discriminant functions are for 2 classes or multiple classes. As for the Least Squares method, each class C_k is presented by the linear model:

$$y_k(x) = w_k^T x + \omega_{k0} \quad (1)$$

where $k=1, \dots, K$. We can group these together so that

$$y(x) = W^T x \quad (2)$$

where, W is a matrix whose k th column comprises the $D + 1$ dimensional vector $w_k = (w_{k0}, w_k^T)^T$ and x is the corresponding augmented input vector $(1, x^T)^T$ with a dummy input $x_0 = 1$. A new input x is then assigned to the class for which the output $y_k = w_k^T x$ is largest. [4]

Consider a training data set $\{x_n, t_n\}$ where $n=1, \dots, N$, and define a matrix T whose n^{th} row is the vector t_n^T , together with a matrix X whose n^{th} row is x_n^T . The sum-of-squares error function is shown below:

$$E_D(W) = \frac{1}{2} \text{Tr}\{(XW - T)^T (XW - T)\} \quad (3)$$

In order to make E_D to be minimal, we can get

$$W = (X^T X)^{-1} X^T T \quad (4)$$

Then we multiple the W we get by using training dataset and the X in testing dataset to get Y , and class k which regarding to the largest y_k is the prediction we get for this point. [4]

2.3 Fisher's linear discriminant using KNN

Consider first the case of two classes, and suppose we take the D -dimensional input vector x and project it down to one dimension using:

$$y = W^T x \quad (5)$$

If we place a threshold on y and classify $y \geq -w_0$ as class C_1 , and otherwise class C_2 , then we obtain our standard linear classifier discussed in the previous section. In general, the projection onto one dimension leads to a considerable loss of information, and classes that are well separated in the original D -dimensional space may become strongly overlapping in one dimension. However, by adjusting the components of the

weight vector w , we can select a projection that maximizes the class separation. To begin with, consider a two-class problem in which there are N_1 points of class C_1 and N_2 points of class C_2 , so that the mean vectors of the two classes are given by^[4]

$$m_k = \frac{1}{N_k} \sum_{n \in C_k} X_n \quad (6)$$

S_B is the between-class covariance matrix and S_W is the total within-class covariance matrix.^[4]

$$S_B = \sum_{k=1}^K N_k (m_k - m)(m_k - m)^T \quad (7)$$

$$m = \frac{1}{N} \sum_{n=1}^N X_n \quad (8)$$

$$S_W = \sum_{k=1}^K S_k \quad (9)$$

$$S_k = \sum_{n \in C_k} (x_n - m_k)(x_n - m_k)^T \quad (10)$$

After we get S_W and S_B , we can calculate $J(W) = \text{Tr}\{S_W^{-1}S_B\}$.

As for the KNN method, we consider a small sphere centred on the point x at which we wish to estimate the density $\rho(x)$, and we allow the radius of the sphere to grow until it contains precisely K data points. We apply the K -nearest-neighbour density estimation technique to each class separately and then make use of Bayes' theorem. Let us suppose that we have a data set comprising N_k points in class C_k with N points in total. If we wish to classify a new point x , we draw a sphere centred on x containing precisely K points irrespective of their class. Suppose this sphere has volume V and contains K_k points from class C_k .^[4]

$$p(x|C_k) = K_k/(N_k V) \quad (11)$$

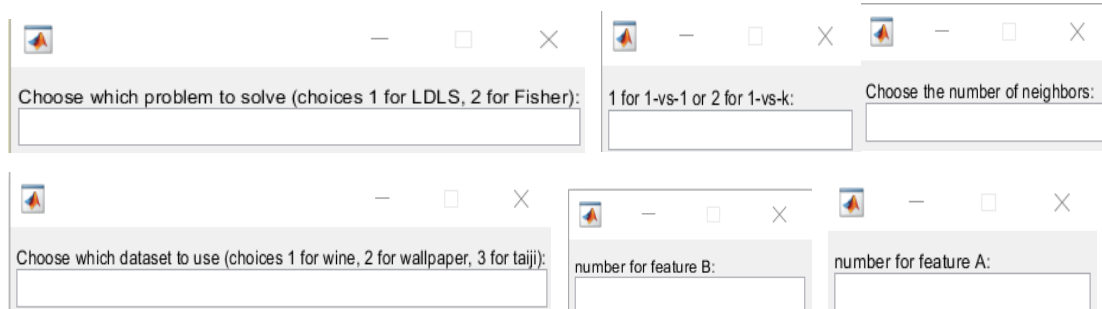
$$p(x) = K/(NV) \quad (12)$$

$$p(C_k) = N_k/N \quad (13)$$

Then we can get $P(C_k|x) = K_k/K$.

3 Results

When we run the code start.m, there will appear some instructions as follow.



The image shows five MATLAB dialog boxes arranged in two rows. The top row contains three boxes: 'Choose which problem to solve (choices 1 for LDLS, 2 for Fisher):', '1 for 1-vs-1 or 2 for 1-vs-k:', and 'Choose the number of neighbors:'. The bottom row contains three boxes: 'Choose which dataset to use (choices 1 for wine, 2 for wallpaper, 3 for taiji):', 'number for feature B:', and 'number for feature A:'. Each box has a text input field below the label.

3.1 Tasks for Linear Discriminate using Least Squares

- You must write a function which takes the training features and labels from the dataset and returns the linear discriminant functions for a 'one-vs-one' or a 'one-vs-all' scheme. You should be familiar with least squares after the first project. You may use the multiclass classification from Bishop 4.1.2: Eq. 4.9 and the condition below that equation.
- A function that takes your linear discriminant functions (from the previous function) and a set of features to test and returns class labels found with your classifier features.

3.1.1 Results on Wine dataset

We choose 1 for “Choose which dataset to use” in this section.

3.1.1.1 Results for 2 features.

We choose 1 for “choose which problem to solve”, 1 for “1-vs-1”. We 1 and 7 for “number of feature A” and “number of feature B” separately. We can get the Figure 1, Figure 2 below, and 4 tables for classification matrix and confusion matrix.

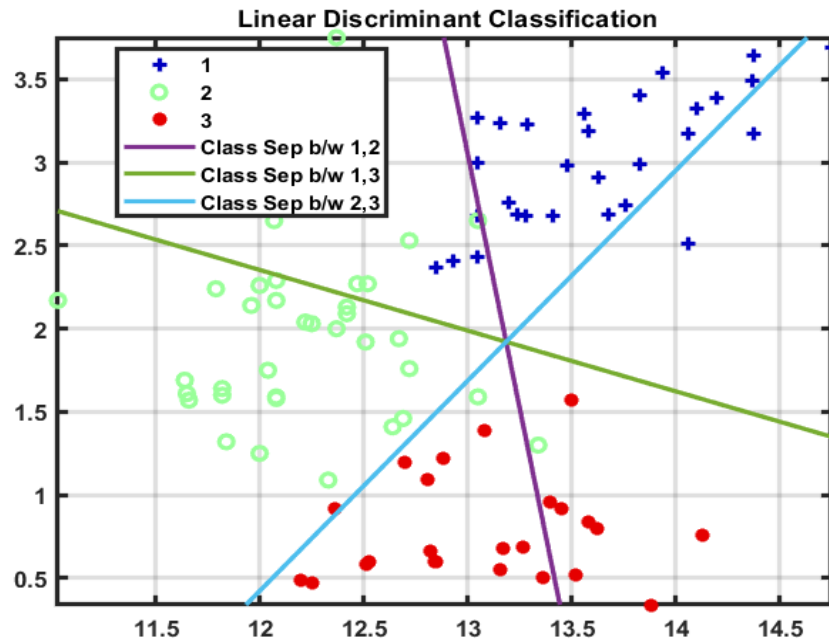


Figure 1 Wine Linear Discriminant Least Square Feature = [1,7]

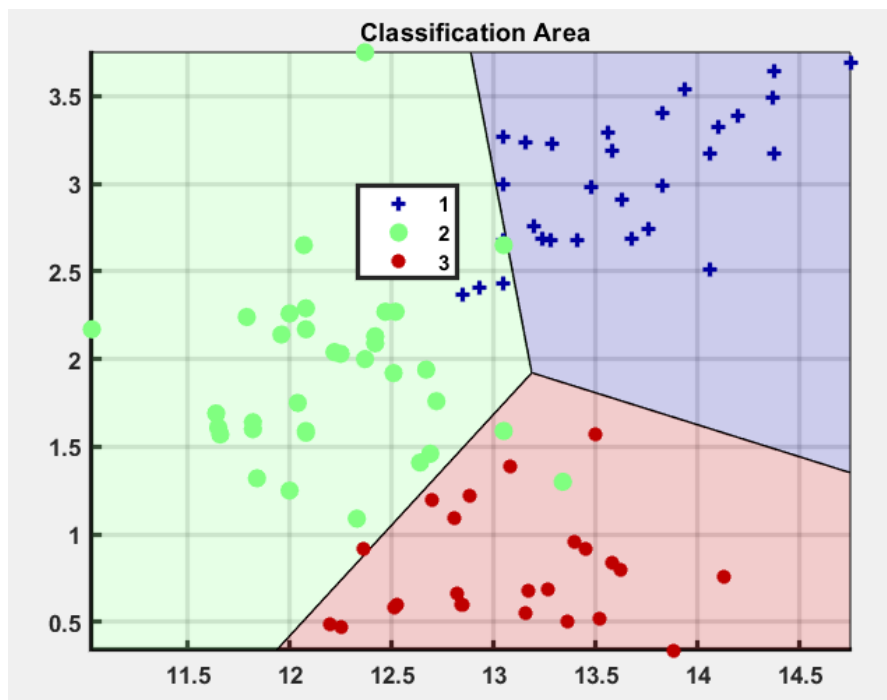


Figure 2 Wine Linear Discriminant Least Square Classification Area Feature = [1,7]

	class1	class2	class3
class1	1	0	0
class2	0.111111	0.833333	0.055556
class3	0	0.041667	0.958333

	class1	class2	class3
class1	30	0	0
class2	4	30	2
class3	0	1	23

Table 1 Wine1 Classification Matrix for Training Dataset Table 2 Wine1 Confusion Matrix for Training Dataset

	class1	class2	class3
class1	0.862069	0.137931	0
class2	0	0.942857	0.057143
class3	0	0.041667	0.958333

Table 3 Wine1 Classification Matrix for Testing Dataset

	class1	class2	class3
class1	25	4	0
class2	0	33	2
class3	0	1	23

Table 4 Wine1 Confusion Matrix for Testing Dataset

And besides, we can also get the accuracy of the training part and the testing part which are 93.06% and 92.11%. The accuracy of the training part is bigger than the testing part. And the standard deviation is 5.17% for testing part and it is smaller than training part which is 8.67%.

3.1.1.1 Results for all features.

We choose 1 for “choose which problem to solve”, 2 for “1-vs-k”. We can get 4 tables for classification matrix and confusion matrix.

	class1	class2	class3
class1	1	0	0
class2	0	1	0
class3	0	0	1

Table 5 Wine1-kClassification Matrix for Training Dataset

	class1	class2	class3
class1	30	0	0
class2	0	36	0
class3	0	0	24

Table 6 Wine1-kConfusion Matrix for Training Dataset

	class1	class2	class3
class1	1	0	0
class2	0	0.971429	0.028571
class3	0	0	1

Table 7 Wine1-kClassification Matrix for Testing Dataset

	class1	class2	class3
class1	29	0	0
class2	0	34	1
class3	0	0	24

Table 8 Wine1-kConfusion Matrix for Testing Dataset

For all features classification, the results of the training part is perfect where the accuracy is 1 and standard deviation is 0, and it is same with the testing part where accuracy is 99.05% and standard deviation is 1.65%.

Overall it seems that Linear Discriminate using Least Squares method does well in Wine dataset.

3.1.2 Results on Wallpaper Group dataset

In this section, we use all the features to classify the dataset to each class.

We choose 2 for “Choose which dataset to use” in this section. And then we choose 1 for “choose which problem to solve”, 2 for “1-vs-k”. We can get 1 figure and we can also get 4 tables for classification matrix and confusion matrix. (Since the table is too big so we use the picture in this section)

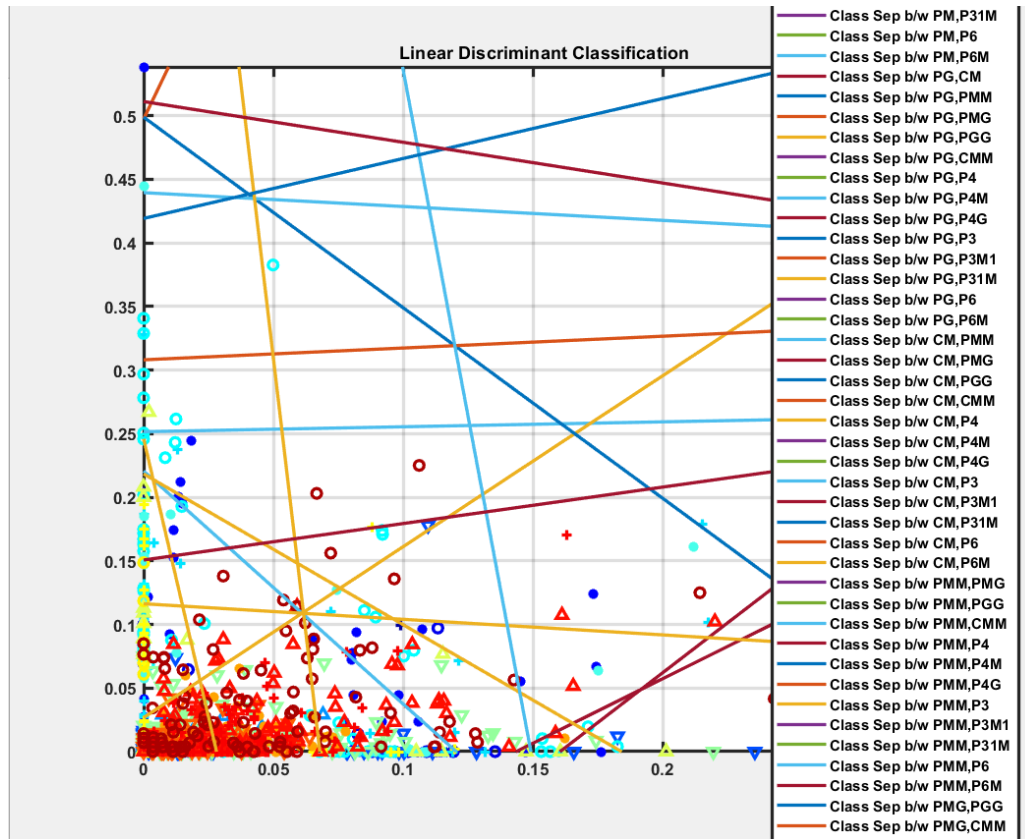


Figure 3 Wallpaper Group1 Linear Discriminant Least Square Classification all Feature

	class1	class2	class3	class4	class5	class6	class7	class8	class9	class10	class11	class12	class13	class14	class15	class16	class17
class1	0.96	0.01	0	0.02	0	0	0	0	0	0.01	0	0	0	0	0	0	0
class2	0.02	0.9	0.01	0.05	0	0	0	0	0	0	0.01	0	0	0	0.01	0	0
class3	0.01	0	0.97	0	0	0	0.02	0	0	0	0	0	0	0	0	0	0
class4	0.02	0.02	0	0.91	0	0	0.01	0.03	0	0.01	0	0	0	0	0	0	0
class5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
class6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
class7	0	0	0.03	0	0	0	0.97	0	0	0	0	0	0	0	0	0	0
class8	0	0	0.01	0.02	0	0	0	0.91	0	0	0	0.06	0	0	0	0	0
class9	0	0	0	0	0.03	0	0	0	0.97	0	0	0	0	0	0	0	0
class10	0.01	0.01	0	0	0	0.01	0	0	0	0.95	0.02	0	0	0	0	0	0
class11	0	0	0	0	0	0	0	0	0	0	0.99	0.01	0	0	0	0	0
class12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
class13	0.01	0	0	0	0.01	0	0	0	0	0	0	0	0.97	0	0.01	0	0
class14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
class15	0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0.99	0	0
class16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.97	0.03
class17	0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0.99

Table 9 Wallpaper Group1 Classification Matrix for Training Dataset

	class1	class2	class3	class4	class5	class6	class7	class8	class9	class10	class11	class12	class13	class14	class15	class16	class17
class1	96	1	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0
class2	2	90	1	5	0	0	0	0	0	0	1	0	0	0	1	0	0
class3	1	0	97	0	0	0	2	0	0	0	0	0	0	0	0	0	0
class4	2	2	0	91	0	0	1	3	0	1	0	0	0	0	0	0	0
class5	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
class6	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0
class7	0	0	3	0	0	0	97	0	0	0	0	0	0	0	0	0	0
class8	0	0	1	2	0	0	0	91	0	0	0	6	0	0	0	0	0
class9	0	0	0	0	3	0	0	0	97	0	0	0	0	0	0	0	0
class10	1	1	0	0	0	1	0	0	0	95	2	0	0	0	0	0	0
class11	0	0	0	0	0	0	0	0	0	0	99	1	0	0	0	0	0
class12	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
class13	1	0	0	0	1	0	0	0	0	0	0	0	97	0	1	0	0
class14	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
class15	0	0	0	0	0	0	0	0	0	0	0	0	1	0	99	0	0
class16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	97	3
class17	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	99

Table 10 Wallpaper Group1 Confusion Matrix for Training Dataset

	class1	class2	class3	class4	class5	class6	class7	class8	class9	class10	class11	class12	class13	class14	class15	class16	class17
class1	0.13	0.19	0.04	0.23	0.01	0.05	0.02	0.04	0.03	0.13	0.02	0.02	0.01	0.04	0.04	0	0
class2	0.12	0.19	0.02	0.16	0.01	0.1	0.07	0.09	0	0.1	0.03	0.02	0.03	0.01	0.01	0.02	0.02
class3	0.02	0.02	0.62	0	0	0.03	0.26	0	0.01	0.01	0	0.01	0.01	0	0.01	0	0
class4	0.08	0.11	0.02	0.37	0	0.04	0.06	0.13	0	0.05	0.03	0.04	0.01	0.03	0.03	0	0
class5	0	0	0	0	0.9	0	0	0	0.02	0	0	0	0.04	0.02	0	0.02	0
class6	0.09	0.06	0.07	0.03	0.03	0.36	0.09	0.03	0.03	0.08	0.11	0	0	0	0	0	0.02
class7	0	0.02	0.14	0.02	0	0.02	0.71	0.01	0	0	0.03	0.02	0	0	0	0.01	0.02
class8	0.01	0.04	0	0.06	0.01	0	0.04	0.46	0	0	0	0.37	0	0	0	0.01	0
class9	0	0.03	0.02	0	0.16	0.01	0	0	0.64	0	0	0	0.01	0	0.04	0.05	0.04
class10	0.05	0.15	0	0.1	0	0.03	0.01	0.03	0.01	0.28	0.2	0.05	0.05	0.01	0	0.01	0.02
class11	0	0.02	0	0.02	0	0.11	0	0	0.01	0.1	0.7	0.03	0	0	0	0	0.01
class12	0	0	0	0	0	0	0	0.07	0	0	0	0.93	0	0	0	0	0
class13	0	0	0	0	0	0	0	0	0.01	0	0	0	0.72	0.11	0.15	0.01	0
class14	0	0	0	0	0	0	0	0	0	0	0	0	0.02	0.98	0	0	0
class15	0	0.01	0	0	0	0	0	0	0	0.01	0	0	0.12	0.01	0.82	0.03	0
class16	0.01	0.01	0.01	0	0	0	0	0	0	0	0	0	0.02	0	0.02	0.86	0.07
class17	0	0	0.04	0	0	0	0	0	0.04	0.01	0	0	0	0	0	0.09	0.82

Table 11 Wallpaper Group1 Classification Matrix for Testing Dataset

	class1	class2	class3	class4	class5	class6	class7	class8	class9	class10	class11	class12	class13	class14	class15	class16	class17
class1	13	19	4	23	1	5	2	4	3	13	2	2	1	4	4	0	0
class2	12	19	2	16	1	10	7	9	0	10	3	2	3	1	1	2	2
class3	2	2	62	0	0	3	26	0	1	1	0	1	1	0	1	0	0
class4	8	11	2	37	0	4	6	13	0	5	3	4	1	3	3	0	0
class5	0	0	0	0	90	0	0	0	2	0	0	0	4	2	0	2	0
class6	9	6	7	3	3	36	9	3	3	8	11	0	0	0	0	0	2
class7	0	2	14	2	0	2	71	1	0	0	3	2	0	0	0	1	2
class8	1	4	0	6	1	0	4	46	0	0	0	37	0	0	0	1	0
class9	0	3	2	0	16	1	0	0	64	0	0	0	1	0	4	5	4
class10	5	15	0	10	0	3	1	3	1	28	20	5	5	1	0	1	2
class11	0	2	0	2	0	11	0	0	1	10	70	3	0	0	0	0	1
class12	0	0	0	0	0	0	0	7	0	0	0	93	0	0	0	0	0
class13	0	0	0	0	0	0	0	0	1	0	0	0	72	11	15	1	0
class14	0	0	0	0	0	0	0	0	0	0	0	0	2	98	0	0	0
class15	0	1	0	0	0	0	0	0	0	1	0	0	12	1	82	3	0
class16	1	1	1	0	0	0	0	0	0	0	0	0	2	0	2	86	7
class17	0	0	4	0	0	0	0	0	4	1	0	0	0	0	0	9	82

Table 12 Wallpaper Group1 Confusion Matrix for Testing Dataset

For all features classification, the results of the training part is perfect where accuracy of Training set is 96.76% and standard deviation of Training set is 3.29%, while the results of the testing part performances bad where the accuracy is 61.7% and the standard deviation is 26.69%

Overall it seems that Linear Discriminate using Least Squares method does well in training dataset of Wallpaper Group while the weight matrix it gets cannot predict the test set well..

3.1.3 Results on Taiji Pose dataset

In this section, we use all the features to classify the dataset to each class.

We choose 2 for “Choose which dataset to use” in this section. And then we choose 1 for “choose which problem to solve”, 2 for “1-vs-k”. We can get 1 figure and we can also get 4 tables for classification matrix and confusion matrix.

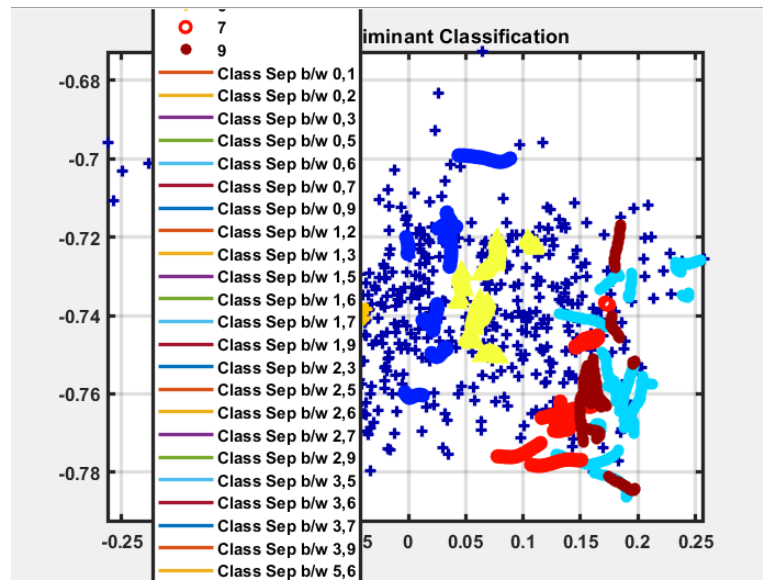


Figure 4 Taiji1 Linear Discriminant Least Square Classification all-Feature

	class1	class2	class3	class4	class5	class6	class7	class8
class1	0.69	0.036	0.062	0.033	0.037	0.052	0.031	0.0589
class2	0	1	0	0	0	0	0	0
class3	0	0	1	0	0	0	0	0
class4	0	0	0	1	0	0	0	0
class5	0	0	0	0	1	0	0	0
class6	0	0	0	0	0	1	0	0
class7	0	0	0	0	0	0	1	0
class8	0	0	0	0	0	0	0	1

Table 13 Taiji 1 Classification Matrix for Training Dataset

	class1	class2	class3	class4	class5	class6	class7	class8
class1	1220	63	109	59	66	92	54	104
class2	0	1066	0	0	0	0	0	0
class3	0	0	2132	0	0	0	0	0
class4	0	0	0	1066	0	0	0	0
class5	0	0	0	0	1066	0	0	0
class6	0	0	0	0	0	2132	0	0
class7	0	0	0	0	0	0	1066	0
class8	0	0	0	0	0	0	0	1066

Table 14 Taiji 1 Confusion Matrix for Training Dataset

	class1	class2	class3	class4	class5	class6	class7	class8
class1	0.43	0.053	0.083	0.045	0.176	0.071	0.035	0.1078
class2	0	1	0	0	0	0	0	0
class3	0	0	1	0	0	0	0	0
class4	0	0	0	1	0	0	0	0
class5	0	0	0	0	1	0	0	0
class6	0	0	0	0	0	1	0	0
class7	0	0	0	0	0	0	1	0
class8	0	0	0	0	0	0	0	1

Table 15 Taiji 1 Classification Matrix for Testing Dataset

	class1	class2	class3	class4	class5	class6	class7	class8
class1	259	32	50	27	106	43	21	65
class2	0	369	0	0	0	0	0	0
class3	0	0	738	0	0	0	0	0
class4	0	0	0	369	0	0	0	0
class5	0	0	0	0	369	0	0	0
class6	0	0	0	0	0	738	0	0
class7	0	0	0	0	0	0	369	0
class8	0	0	0	0	0	0	0	369

Table 16 Taiji 1 Confusion Matrix for Testing Dataset

For all features classification, the results of the training part is perfect where Training accuracy is 96.13% and Training standard deviation is 10.94% , and the results of the testing part performances a little worse than the training part where the accuracy is 92.87% and the standard deviation is 20.17%

Overall it seems that Linear Discriminate using Least Squares method does well in both training and testing dataset of Taiji Pose dataset.

3.2 Tasks for Fisher's linear discriminant using KNN

- A function to find the Fisher projection using the training features and labels (Bishop 4.1.4 and 4.1.6) and also train a classifier to the Fisher projected training data. The classifier can be a KNN classifier (Bishop 2.5.2) or from Decision Theory (end of Bishop 4.1.4) using an optimum threshold. This function returns the Fisher projection coefficients and the corresponding fitted classifier necessary for the testing function.
- A function which takes the output of the previous function (the Fisher projection and the classifier) and a set of features to test and returns the class labels of the features found by your classifier.

3.2.1 Results on Wine dataset

We choose 1 for “Choose which dataset to use” in this section.

3.2.1.1 Results for 2 features.

We choose 2 for “choose which problem to solve”, 1 for “1-vs-1”. We 1 and 7 for “number of feature A” and “number of feature B” separately, and 5 for “number of neighbors”. We can get the 1 Figure, below, and 4 tables for classification matrix and confusion matrix.

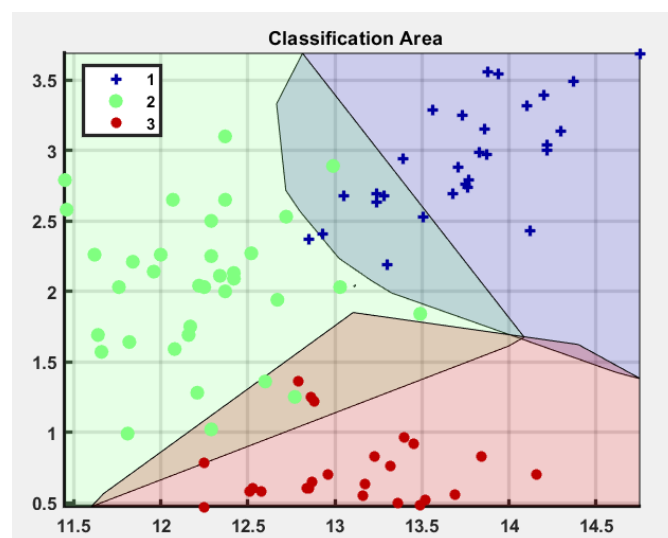


Figure5 Wine Fisher-KNN Classification Area Feature = [1,7]

	class1	class2	class3
class1	1	0	0
class2	0.0833	0.8611	0.0556
class3	0	0.125	0.875

	class1	class2	class3
class1	30	0	0
class2	3	31	2
class3	0	3	21

Table 17 Wine2 Classification Matrix for Training Dataset Table 18 Wine2 Confusion Matrix for Training Dataset

	class1	class2	class3
class1	0.965517	0.034483	0
class2	0.028571	0.942857	0.028571
class3	0	0	1

	class1	class2	class3
class1	28	1	0
class2	1	33	1
class3	0	0	24

Table 19 Wine2 Classification Matrix for Testing Dataset Table 20 Wine2 Confusion Matrix for Testing Dataset

And besides, we can also get the accuracy of the training part and the testing part which are 91.2% and 96.95%. The accuracy of the training part is bigger than the testing part. And the standard deviation is 2.88% for testing part and it is smaller than training part which is 7.56%

When we change the value of k, the accuracy will be influenced, and the best performance for feature 1 and feature 7 is based on $k = 5$.

3.2.1.2 Results for all features.

We choose 2 for “choose which problem to solve”, 1 for “1-vs-1”. We 1 and 7 for “number of feature A” and “number of feature B” separately, and 5 for “number of neighbors”. We can get the 4 tables for classification matrix and confusion matrix.

	class1	class2	class3
class1	1	0	0
class2	0	1	0
class3	0	0	1

	class1	class2	class3
class1	30	0	0
class2	0	36	0
class3	0	0	24

Table 21 Wine2-1 Classification Matrix for Training Dataset Table 22 Wine2-1 Confusion Matrix for Training Dataset

	class1	class2	class3		class1	class2	class3
class1	1	0	0	class1	29	0	0
class2	0	0.971429	0.028571	class2	0	34	1
class3	0	0	1	class3	0	0	24

Table 23 Wine2-1Classification Matrix for Testing Dataset Table 24 Wine2-1Confusion Matrix for Testing Dataset

In this part, we can also get the accuracy of the training part and the testing part which are 100% and 99.05%. And the standard deviation is 1.65% for testing part and is 0% for training part.

3.2.2 Results on Wallpaper Group dataset

In this section, we use all the features to classify the dataset to each class.

We choose 2 for “Choose which dataset to use” in this section. And then we choose 2 for “choose which problem to solve”, 2 for “1-vs-k”, and 15 for “number of neighbors”.

We can get 4 tables for classification matrix and confusion matrix.

	class1	class2	class3	class4	class5	class6	class7	class8	class9	class10	class11	class12	class13	class14	class15	class16	class17
class1	0.98	0.01	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0
class2	0.03	0.92	0	0.04	0	0	0	0	0	0.01	0	0	0	0	0	0	0
class3	0.01	0	0.96	0.01	0	0	0.02	0	0	0	0	0	0	0	0	0	0
class4	0.01	0	0	0.96	0	0	0	0.02	0	0.01	0	0	0	0	0	0	0
class5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
class6	0	0	0.01	0	0	0.99	0	0	0	0	0	0	0	0	0	0	0
class7	0	0	0.02	0	0	0	0.98	0	0	0	0	0	0	0	0	0	0
class8	0	0	0	0.03	0	0	0	0.94	0	0	0	0.03	0	0	0	0	0
class9	0	0	0	0	0.01	0	0	0	0.99	0	0	0	0	0	0	0	0
class10	0.01	0.01	0	0.01	0	0	0	0	0	0.96	0.01	0	0	0	0	0	0
class11	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
class12	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
class13	0	0	0	0	0.01	0	0	0	0	0	0	0	0.98	0	0.01	0	0
class14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
class15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
class16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.98	0.02
class17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 25 Wallpaper Group2 Classification Matrix for Training Dataset

	class1	class2	class3	class4	class5	class6	class7	class8	class9	class10	class11	class12	class13	class14	class15	class16	class17
class1	98	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
class2	3	92	0	4	0	0	0	0	0	1	0	0	0	0	0	0	0
class3	1	0	96	1	0	0	2	0	0	0	0	0	0	0	0	0	0
class4	1	0	0	96	0	0	0	2	0	1	0	0	0	0	0	0	0
class5	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
class6	0	0	1	0	0	99	0	0	0	0	0	0	0	0	0	0	0
class7	0	0	2	0	0	0	98	0	0	0	0	0	0	0	0	0	0
class8	0	0	0	3	0	0	0	94	0	0	0	3	0	0	0	0	0
class9	0	0	0	0	1	0	0	0	99	0	0	0	0	0	0	0	0
class10	1	1	0	1	0	0	0	0	0	96	1	0	0	0	0	0	0
class11	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0
class12	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
class13	0	0	0	0	1	0	0	0	0	0	0	0	98	0	1	0	0
class14	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
class15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0
class16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	98	2
class17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100

Table 26 Wallpaper Group2 Confusion Matrix for Training Dataset

	class1	class2	class3	class4	class5	class6	class7	class8	class9	class10	class11	class12	class13	class14	class15	class16	class17
class1	0.3	0.22	0.02	0.24	0	0.06	0	0.01	0	0.13	0.02	0	0	0	0	0	0
class2	0.19	0.27	0.03	0.13	0	0.11	0.02	0.04	0	0.19	0.02	0	0	0	0	0	0
class3	0.02	0.03	0.69	0.02	0	0.02	0.22	0	0	0	0	0	0	0	0	0	0
class4	0.12	0.15	0.02	0.44	0	0.02	0.05	0.1	0	0.09	0.01	0	0	0	0	0	0
class5	0	0	0	0	0.94	0	0	0	0.01	0	0	0	0.05	0	0	0	0
class6	0.12	0.09	0.03	0	0	0.54	0.07	0	0	0.07	0.08	0	0	0	0	0	0
class7	0.03	0.02	0.16	0.02	0	0.01	0.71	0.02	0	0.01	0.02	0	0	0	0	0	0
class8	0.02	0.01	0	0.07	0	0	0.03	0.64	0	0	0	0.23	0	0	0	0	0
class9	0	0	0.01	0	0.11	0	0	0	0.79	0	0	0	0	0	0.05	0.03	0.01
class10	0.12	0.21	0	0.04	0	0.01	0	0.07	0	0.37	0.16	0.02	0	0	0	0	0
class11	0.01	0.02	0	0.02	0	0.05	0	0	0	0.15	0.75	0	0	0	0	0	0
class12	0	0	0	0	0	0	0	0.06	0	0	0	0.94	0	0	0	0	0
class13	0	0	0	0	0	0	0	0	0	0	0	0	0.96	0.02	0.02	0	0
class14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
class15	0	0	0	0	0	0	0	0	0	0	0	0	0.05	0	0.93	0.02	0
class16	0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0.02	0.92	0.05
class17	0	0	0	0	0	0	0	0	0.04	0	0	0	0	0	0	0.05	0.91

Table 27 Wallpaper Group2 Classification Matrix for Testing Dataset

	class1	class2	class3	class4	class5	class6	class7	class8	class9	class10	class11	class12	class13	class14	class15	class16	class17
class1	30	22	2	24	0	6	0	1	0	13	2	0	0	0	0	0	0
class2	19	27	3	13	0	11	2	4	0	19	2	0	0	0	0	0	0
class3	2	3	69	2	0	2	22	0	0	0	0	0	0	0	0	0	0
class4	12	15	2	44	0	2	5	10	0	9	1	0	0	0	0	0	0
class5	0	0	0	0	94	0	0	0	1	0	0	0	5	0	0	0	0
class6	12	9	3	0	0	54	7	0	0	7	8	0	0	0	0	0	0
class7	3	2	16	2	0	1	71	2	0	1	2	0	0	0	0	0	0
class8	2	1	0	7	0	0	3	64	0	0	0	23	0	0	0	0	0
class9	0	0	1	0	11	0	0	0	79	0	0	0	0	0	5	3	1
class10	12	21	0	4	0	1	0	7	0	37	16	2	0	0	0	0	0
class11	1	2	0	2	0	5	0	0	0	15	75	0	0	0	0	0	0
class12	0	0	0	0	0	0	0	6	0	0	0	94	0	0	0	0	0
class13	0	0	0	0	0	0	0	0	0	0	0	0	96	2	2	0	0
class14	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
class15	0	0	0	0	0	0	0	0	0	0	0	0	5	0	93	2	0
class16	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	92	5
class17	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	5	91

Table 28 Wallpaper Group2 Confusion Matrix for Testing Dataset

For all features classification, the results of the training part is perfect where Training accuracy is 97.88% and standard deviation is 2.37%, while the results of the testing part performances bad where the accuracy is 71.18% and the standard deviation is 24.71%. The method does well in training part while does not that bad in testing part.

When we change the value of k, the best performance is based on $k = 15$.

3.2.3 Results on Taiji Pose dataset

In this section, we use all the features to classify the dataset to each class.

We choose 3 for “Choose which dataset to use” in this section. And then we choose 2 for “choose which problem to solve”, 2 for “1-vs-k” and $k=15$. We can get 4 tables for classification matrix and confusion matrix.

	class1	class2	class3	class4	class5	class6	class7	class8
class1	0.844	0.025	0.041	0.019	0.022	0.022	0.014	0.013
class2	0	1	0	0	0	0	0	0
class3	0	0	1	0	0	0	0	0
class4	0	0	0	1	0	0	0	0
class5	0	0	0	0	1	0	0	0
class6	0	0	0	0	0	1	0	0
class7	0	0	0	0	0	0	1	0
class8	0	0	0	0	0	0	0	1

Table 29 Taiji 2 Classification Matrix for Training Dataset

	class1	class2	class3	class4	class5	class6	class7	class8
class1	1491	45	72	34	39	38	25	23
class2	0	1066	0	0	0	0	0	0
class3	0	0	2132	0	0	0	0	0
class4	0	0	0	1066	0	0	0	0
class5	0	0	0	0	1066	0	0	0
class6	0	0	0	0	0	2132	0	0
class7	0	0	0	0	0	0	1066	0
class8	0	0	0	0	0	0	0	1066

Table 30 Taiji 2 Confusion Matrix for Training Dataset

	class1	class2	class3	class4	class5	class6	class7	class8
class1	0.741	0.027	0.045	0.028	0.071	0.043	0.023	0.0216
class2	0.03	0.97	0	0	0	0	0	0
class3	0.016	0	0.984	0	0	0	0	0
class4	0	0	0	1	0	0	0	0
class5	0	0	0	0	1	0	0	0
class6	0	0	0	0	0	1	0	0
class7	0	0	0	0	0	0	1	0
class8	0	0	0	0	0	0	0	1

Table 31 Taiji 2 Classification Matrix for Testing Dataset

	class1	class2	class3	class4	class5	class6	class7	class8
class1	447	16	27	17	43	26	14	13
class2	11	358	0	0	0	0	0	0
class3	12	0	726	0	0	0	0	0
class4	0	0	0	369	0	0	0	0
class5	0	0	0	0	369	0	0	0
class6	0	0	0	0	0	738	0	0
class7	0	0	0	0	0	0	369	0
class8	0	0	0	0	0	0	0	369

Table 32 Taiji 2 Confusion Matrix for Testing Dataset

For all features classification, the results of the training part is perfect where Training accuracy is 98.05% and standard deviation is 5.52%, while the results of the testing part performances bad where the accuracy is 96.19% and the standard deviation is 8.98%.

The method does well in training part while does not that bad in testing part.

When we change the value of k, the accuracy will be influenced, and the best performance is based on $k = 15$.

4 Conclusions

We learn deeply by implement this two classifications.

It is easy to find that almost for every data set, Fisher's with KNN classifier can obtain higher accuracy rate. Showing in the table below.

	Accuracy			
	LDLS	FKNN	LDLS	FKNN
Dataset	Training		Testing	
Wine	100%	100%	99.05%	99.05%
Wallpaper	96.76%	97.88%	61.70%	71.18%
Taiji	96.13%	98.05%	92.87%	96.19%

Table 33 Comparison of the accuracy

And the Discriminant function with Least squares method is lack of robustness, we find notice this by comparing Figure 6 and Figure 7, we can find that they are both classification for feature 1 and feature 7 of wine dataset, but when the points of class 1 change a little the outliers changes a lot. And when we

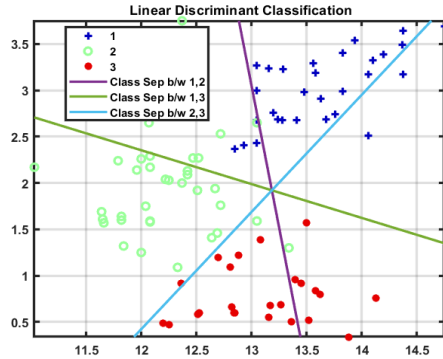


Figure 6 Wine Linear Discriminant
Least Square Feature = $[1,7]1$

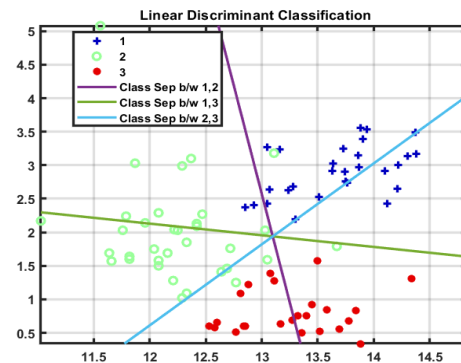


Figure 7 Wine Linear Discriminant
Least Square Feature = $[1,7]2$

When we observe, Figure 2 and Figure 5, we can also get the conclusion that the method of Fisher KNN is more robust and more accuracy.

Reference

- 1 <http://archive.ics.uci.edu/ml/datasets/Wine>
- 2 https://en.wikipedia.org/wiki/Wallpaper_group
- 3 Project 2 description (professor gives to us on Canvas)
- 4 Christopher, M. Bishop. PATTERN RECOGNITION AND MACHINE LEARNING.
Springer-Verlag New York, 2016.