CSE586 Project 2: Supervised Classifiers

Sree Sai Teja Lanka

February 17, 2019

Abstract

The goal of this project is to get you familiar with common classifiers and how to compare and contrast different classifiers. Furthermore, learn how to implement two supervised classifiers, to use Fisher projection on your data, and evaluate their performance on real data using Wine, Wallpaper, Thiji datasets. The objectives of this project involve: 1. Linear Discriminate using Least Squares for classification: To makeup a function which takes the training features and labels from the considered dat aset and returns the linear discriminant functions for a 'one-vs-all' scheme, To assemble a function that takes your linear discriminant function and a set of features to test and return class labels found with classifier features. 2. Linear Discriminate using Fischer Projection:To build a function to find the Fisher projection using the training features and labels which returns the Fisher projection coefficients. And to train a classifier to the Fisher projected training data using K-nearest neighbors (KNN) which returns the corresponding fitted classifier necessary for the testing function. Furthermore, To build a function which takes the output of the Fisher projection and the classifier, and a set of features to test and returns the class labels of the features found by (KNN). 3. Compare the performance of the both Classifiers.

Contents

1 Introduction

Supervised learning is the machine learning method of inferring a function from labeled training data. A supervised learning algorithm analyzes the training data and produces a classifier function, which can be used for mapping the test data. Here, the supervised learning is accomplished using the Linear Discriminate function given by two approaches 'Least squares Method' and 'Fisher Projection':

- Least Squares Method- It is also known as the Direct Error minimization method, which has been popular for a long time. Least Squares minimizes the square of the error between the original data and the values predicted by the equation.The classifier used for this method could be one-vs-one or one-vs-rest. Here, I have implemented using the one-vs-all classifier.

- One vs all classifier - The one-vs-all, method involves training a single classifier per class, with the samples of that class as positive samples and all other samples as negatives. This strategy requires the base classifiers to produce a real-valued confidence score for its decision, and the maximum of the score is obtained. The corresponding class for the maximum score is assigned to the test label.

- Fischer Projection - LDA seeks to reduce dimensionalty while preserving as much of the class discriminatory information as possible. Fischer projection method suggested to maximize a function that will give a large separation between the projected classes means while also give small variance within each class, thus minimizing the class overlapping the projected space.The classifiers used to assign each input vector to one of a finite number of discrete categories for this method is

- The K-nearest neighbors (KNN) is a non-parametric method used for classification. The input consists of the K closest training examples in the feature space. The output in the KNN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its K nearest neighbors. And I have used an optimal K for change in number of dataset features.

2 Approach

2.1 Least Squares for Classification

Consider a general classification problem with K classes, with a 1-of-K binary coding scheme for the target vector t. One justification for using least squares in such a context is that it approximates the conditional expectation $E[t|x]$ of the target values given the input vector. For the binary coding scheme, this conditional expectation is given by the vector of posterior class probabilities. Each class Ck is described by its own linear model so that,

$$y_k(x) = w_{k0} + w_k^T x \tag{1}$$

This above equation would state us good approximate predictions for the new data. This can be utilized to classify the data using the Least Squares for classification.

Tilde W if T is a vector of target labels, the D-dimensional error function is given by:

$$\tilde{E_D}(\tilde{W}) = \frac{1}{2}Tr\{(\tilde{X}\tilde{W} - T)^T(\tilde{X}\tilde{W} - T)\} \tag{2}$$

Upon taking the derivative wrt $\tilde{W}$, we obtain:

$$\tilde{E'}(\tilde{W}) = \frac{\mathrm{d}}{\mathrm{d}\tilde{W}}\left\{\frac{1}{2}\left[(\tilde{X}\tilde{W} - T)^T(\tilde{X}\tilde{W} - T)\right]\right\} = 0 \tag{3}$$

Solving equation
refeq:1, we get,

$$\frac{\mathrm{d}}{\mathrm{d}\tilde{W}}\left\{\frac{1}{2}\left[(\tilde{X}\tilde{W} - T)^T(\tilde{X}\tilde{W} - T)\right]\right\} = 0 \tag{4}$$

$$\left[\tilde{X}^T\tilde{X}\tilde{W}^* - \tilde{X}^TT\right] = 0 \tag{5}$$

$$(\tilde{X}^T\tilde{X})\tilde{W}^* = \tilde{X}^TT \tag{6}$$

$$\tilde{W}^* = (\tilde{X}^T\tilde{X})^{-1}\tilde{X}^TT \tag{7}$$

$$Y = \tilde{X}(\tilde{X}^T\tilde{X})^{-1}\tilde{X}^TT \tag{8}$$

For predicting the labels using one-vs-all classifier, we take the maximum value of all the classes in Y and the corresponding index gives the label/class of the data point. The classifier used for this methos is 'One-vs-all' classifier.

2.1.1 One-versus-the-Rest Classifier

The one-vs-all, method involves training a single classifier per class, with the samples of that class as positive samples and all other samples as negatives. This strategy requires the base classifiers to produce a real-valued confidence score for its decision, and the maximum of the score is obtained. The corresponding class for the maximum score is assigned to the test label as mentioned in the reference [1]

$$f(x) = argmax_i(f_i(x))$$

where $f_i(x)$ individual score of each class.

2.2 Fishers Linear Discriminant

Fisher's Linear Discriminant/LDA seeks to reduce dimensionality while preserving as much of the class discriminatory information as possible. The dimensionality can be reduced by maximizing the class separation. It can be achieved by two methods- By maximizing the distance between the projected class means. However, this is not a optimal measure since it does not account for the standard deviation within the classes. By minimizing the projected intra-class scatters or variances. However, this method also fails when the classes have small intra-class variances but their means overlap. To tackle this problem, we use the idea proposed by Fischer. In Fischers projection, we maximize a function that will give a large separation between the projected class means while also give small variance within each class, thus minimizing the class overlapping the projected

space. In simple words, it can be visualized as maximizing the ratio of the between-class variance to the within-class variance OR the difference of the projected means and decrease the sum of the intra-class variances. The projection formula for an input vector x of dimensions D projected down to one dimension is given by,

$$y = w^T * x \tag{9}$$

Now assume a case of two classes, in which there are N1 points of class C_1 and N_2 points of class C_2. The mean vectors of the each class are given by,

$$\begin{aligned} m1 &= \frac{1}{N1} \sum_{n \in C_1} x_n \\ m2 &= \frac{1}{N2} \sum_{n \in C_2} x_n \end{aligned} \tag{10}$$

The simplest measure of the separation of the classes, when projected onto w, is the separation of the projected class means. This suggests that we might choose w so as to maximize,

$$m2 - m1 = w^T(m2 - m1) \tag{11}$$

This can be generalized as,

$$m_k = w^T * m_k \tag{12}$$

The within-class variance of the transformed data from class therefore given by the following equation with class c_k,

$$var_k^2 = \sum_{y_n \epsilon c_k} (y_n - m_k)^2 \tag{13}$$

The Fisher criterion is defined to be the ratio of the between-class variance to the within-class variance and is given by,

$$J(w) = \frac{(m_1 - m_2)^2}{var_1^2 + var_2^2} \tag{14}$$

The numerator can be simplified as follows

$$\begin{aligned} (m_2 - m_1)^2 &= (w^T * m_2 - w^T * {m_1}^2 \\ &= Tr[(w^T * m_2 - w^T * m_1)^T * (w^T * m_2 - w^T * m_1)] \\ &= Tr[(w^T)^T * (m_2 - m_1)^T * w^T * (m_2 - m_1)] \\ &= Tr[(m_2 - m_1)^T * w * w^T * (m_2 - m_1)] \\ &= Tr[w^T * (m_2 - m_1) * (m_2 - m_1)^T * w] \end{aligned} \tag{15}$$

(note RHS is projected mean. LHS is mean prior projection)

$$(m_2 - m_1)^2 = w_T * var_B * w \tag{16}$$

The var_B the between class co-variance matrix and is given by the equation,

$$var_B = (m_2 - m_1) * (m_2 - m_1)^T \tag{17}$$

Now the denominator can be simplified as with class as c_k and some constant m' for making the calculations easy,

$$\begin{aligned} var_k^2 &= \sum_{n \epsilon c_k} (y_n - m_k)^2 \\ &= \sum_{n \epsilon c_k} (w^T * x_n - w^T * m')^2 \\ &= \sum_{n \epsilon c_k} Tr[(w^T * x_n - w^T * m')^T (w^T * x_n - w^T * m')] \\ &= \sum_{n \epsilon c_k} Tr[(x_n - m')^T * w * w^T * (x_n - m')] \\ &= \sum_{n \epsilon c_k} Tr[w^T * (x_n - m') * (x_n - m')^T * w] \\ &= [w^T * var_W * W] \\ &= [W^T var_W W] \end{aligned} \tag{18}$$

The var_W is the total within-class co-variance matrix and is given by,

$$var_W = \sum_{n \epsilon c_1} (x_n - m_1)(x_n - m_1)^T + \sum_{n \epsilon c_2} (x_n - m_2)(x_n - m_2)^T \tag{19}$$

Now,

$$J(w) = \frac{[W^T var_B W]}{[W^T var_W W]} \tag{20}$$

q Differentiating w.r.t. w, we find that $J(w)$ is maximized when,

$$\begin{aligned} \frac{\partial}{\partial w} J(w) &= 0 \\ \frac{\partial}{\partial w} (W^T var_B W)(W^T var_w W) &= 0 \\ \frac{(W^T var_W W)\, \partial (W^T var_B W)}{\partial w} - \frac{(W^T var_B W)\, \partial (W^T var_W W)}{\partial w} &= 0 \\ (W^T var_W W)(2\ var_B W) - (W^T var_B W)(2\ var_W W) &= 0 \\ (W^T var_W W)\ var_B W &= (W^T var_B W)\ var_W W \end{aligned} \tag{21}$$

We see that $var_B W$ is always in the direction of $(m_2 - m_1)$. Furthermore, we do not care about the magnitude of w, only its direction, and so we can drop the scalar factors$(W^T var_B W)$ and $(W^T var_W W)$ and then multiplying both sides by inverse of var_W gives the proportionality,

$$W\ \alpha\ \ var_W^{-1}(m_2 - m_1) \tag{22}$$

The result obtained from equation 22 is known as Fishers linear discriminant. For a generalized case, that is when K > 2, the generalization of the within-class co-variance matrix to the case of K classes is as follows:

$$var_W = \sum_{k=1}^{K} var_k$$

where,

$$\begin{aligned} var_k &= \sum_{n \epsilon c_k} (x_n - m_k)(x_n - m_k)^T \\ m_k &= \frac{1}{N_k} \sum_{n \epsilon c_k} x_n \end{aligned} \tag{23}$$

N_k is number of patterns in class k denoted as c_k and var_B is the between class variance and is given by,

$$var_B = \sum_{k=1}^{K} N_k (m_k - m)(m_k - m)^T \tag{24}$$

where m is the mean of the total data set.

$$\begin{aligned} m &= \frac{1}{N} \sum_{n=1}^{N} x_n \\ &= \frac{1}{N} \sum_{k=1}^{K} N_k m_k \end{aligned} \tag{25}$$

The co-variance matrices in projected space can be written in similar form as,

$$s_W = \sum_{k=1}^{K} \sum_{n \epsilon c_k} (y_n - \mu_k)(y_n - \mu_k)^T \tag{26}$$

$$s_B = \sum_{k=1}^{K} N_k (\mu_k - \mu)(\mu_k - \mu)^T \tag{27}$$

Here μ is the global mean and μ_k is the per-class mean.

$$\begin{aligned} \mu_k &= \frac{1}{N_k} \sum_{n \epsilon c_k} y_n \\ \mu &= \frac{1}{N} \sum_{k=1}^{K} N_k \mu_k \end{aligned} \tag{28}$$

The explicit n explicit function of the projection matrix w in the form,

$$J(w) = Tr[(W^T var_W W)^{-1} (W^T var_B W)] \tag{29}$$

And on a similar derivation as above, we can get optimized w which is $w*$ by the Eigen vectors of $var_W^{-1} var_B$ which corresponds to the D' largest eigenvalues. Where

$$\begin{aligned} w^* &= [w_1^* | w_2^* \; w_3^* \; \ldots \; w_{c-1}^*] \\ w^* &= argmax(\frac{[W^T var_B W]}{[W^T var_W W]}) \\ (var_B \; \lambda_i \; var_W) w_i^* &= 0 \end{aligned} \tag{30}$$

2.2.1 K-Nearest Neighbour Classifier

The K-nearest neighbors (KNN) is a non-parametric method used for classification. The input consists of the K closest training examples in the feature space. The output in the KNN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its K nearest neighbors. The algorithm for the KNN can be given as follows Training phase: A model is constructed from the training data. Classification algorithm finds relationships between predictors and targets, and relationships are summarized in a model Classifying phase: Use the model for classification on test data whose class labels are unknown. The K nearest neighbors are found out by computing the distances between the test data and the training data. The most popular measure of the distance is the Euclidean distance. The Euclidean distance between two points x and y is:

$$d(x, y) = \sqrt{\sum_{i=1}^{K}(x_i - y_i)}$$

Where K is number of nearest neighbours.
Then the distances are sorted in the ascending order and the K nearest distances are considered and the corresponding K nearest neighbors are taken. The vote of the most number of classes is in the K nearest neighbors is assigned as the class label of the test data.

3 Results

The above mentioned methods are performed on three different datasets namely: Wine (with 13 feautres for a data point), Wallpapers (with 500 feautres for a data point) and Taiji (with 64 feautres for a data point).The results are analyzed using the confusion matrices, classification matrices as well as accuracy rate and standard deviations of the data predicted by training the model for the linear discriminant.
Confusion Matrix: A confusion matrix, also known as an error matrix, is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one. Each row of the matrix represents the instances in a predicted class while each column represents the instances in an actual class. The name stems from the fact that it makes it easy to see if the system is confusing two classes.
Classification Matrix: A classification matrix sorts all cases from the model into categories, by determining whether the predicted value matched the actual value. All the cases in each category are then counted, and the totals are displayed in the matrix

3.1 Linear Discriminant using Least Squares

The least squares method is utilized to model a linear discriminant using one vs all classifier for predicting the classes and their accuracy with the visualization of plots for classification along with confusion matrices.

3.1.1 WINE Dataset

WINE dataset has 13 features for a data point and the classifier used is 'one-vs-all'. There are results based on plots for features 1,7 and also the classification matrix and confusion matrix based on all features. From the figure 1, we can say that there are outliers for class 2, and class 1 has minimal outliers.
And from table ?? we can show that the confusion matrix after test is performed on model is decently indented diagonally.

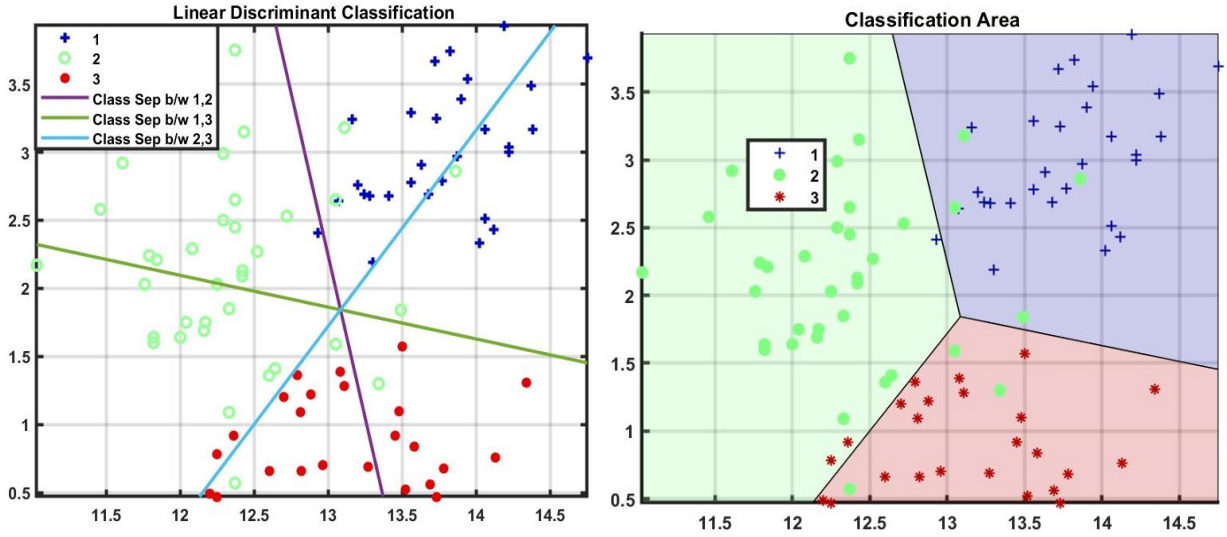

Figure 1: Plots for the Linear Discriminant classification using least square method for WINE Dataset taking feature 1 and feature7. Lines denote the decision boundaries. Blue, green and red points denote the class 1, class 2 and class 3 respectively.

1	0	0
0	1	0
0	0	1

Train Classification Matrix

1	0	0
0.057	0.914	0.029
0	0	1

Test Classification Matrix

30	0	0
0	36	0
0	0	24

Train Confusion Matrix

29	0	0
2	32	1
0	0	24

Test Confusion Matrix

Table 1: Confusion and Classification matrices for the WINE dataset

3.1.2 WALLPAPER Dataset

WALLPAPER dataset has huge set of features say 500 features for a single data point and the classifier used is 'one-vs-all'. There are results based on plots for features 1,7 and also the classification matrix and confusion matrix based on all features. From the figure 2, we can say that there are outliers and due to many features and classes the outliers class and definition are not visualised.
And from table ?? we can show that the confusion matrix before train is decently indented diagonally with proper zeros on the other terms of a diagonal but after the training part is put to test and results in the values of diagonal indentation but there are no valued amounts of zeros in non-diagonal areas, But the values in diagonal region are far more

greater than non-diagonal regional values from matrix.
When we consider the classification matrix of train with test the classification at train stage is more fitting than testing hence chance of over fitting may be seen based on accuracy in the later tables.

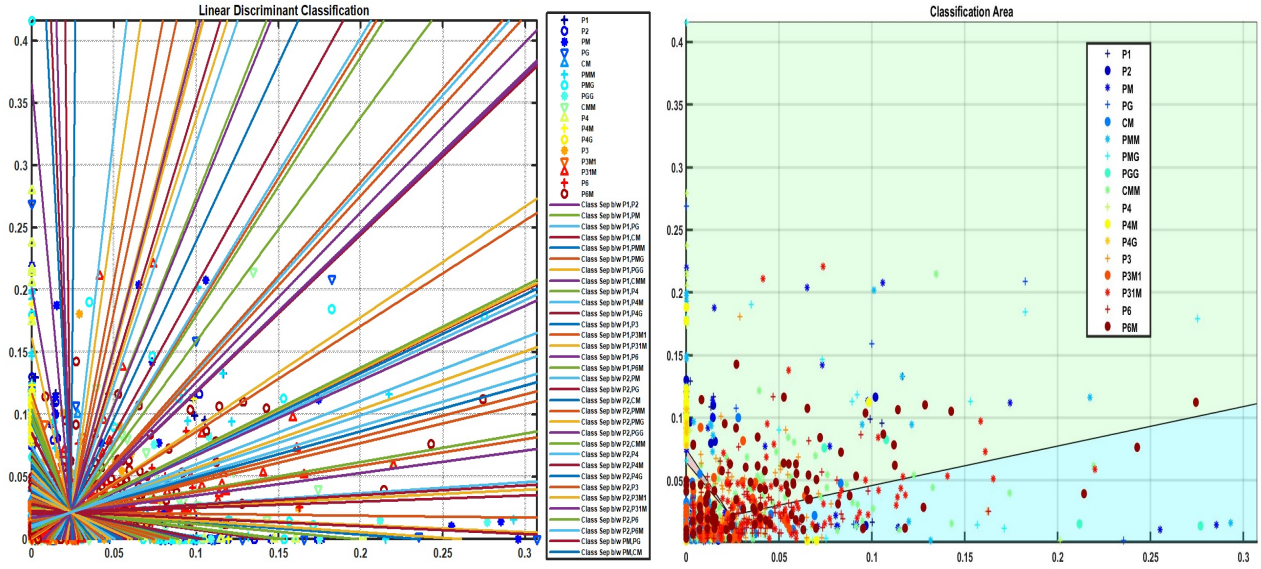

Figure 2: Plots for the Linear Discriminant classification using least square method for WINE Dataset taking feature 1 and feature7. Lines denote the decision boundaries.

0.96	0.01	0	0.02	0	0	0	0	0	0.01	0	0	0	0	0	0	0
0.02	0.9	0.01	0.05	0	0	0	0	0	0	0.01	0	0	0	0.01	0	0
0.01	0	0.97	0	0	0	0.02	0	0	0	0	0	0	0	0	0	0
0.02	0.02	0	0.91	0	0	0.01	0.03	0	0.01	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0.03	0	0	0	0.97	0	0	0	0	0	0	0	0	0	0
0	0	0.01	0.02	0	0	0	0.91	0	0	0	0.06	0	0	0	0	0
0	0	0	0	0.03	0	0	0	0.97	0	0	0	0	0	0	0	0
0.01	0.01	0	0	0	0.01	0	0	0	0.95	0.02	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0.99	0.01	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0.01	0	0	0	0.01	0	0	0	0	0	0	0	0.97	0	0.01	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0.99	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.97	0.03
0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0.99

Table 2: Train Classification Matrix: Showing that the training set if classified with few outliers

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
0.12	0.19	0.02	0.16	0.01	0.1	o.o7	0.09	0	0.1	0.03	0.02	0.03	0.01	0.01	0.02	0.02
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
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
0	0	0	0	0.9	0	0	0	0.02	0	0	0	0.04	0.02	0	0.02	0
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
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
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
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
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
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
0	0	0	0	0	0	0	0.007	0	0	0	0.93	0	0	0	0	0
0	0	0	0	0	0	0	0	0.01	0	0	0	0.72	0.11	0.15	0.01	0
0	0	0	0	0	0	0	0	0	0	0	0	0.02	0.98	0	0	0
0	0.01	0	0	0	0	0	0	0	0.01	0	0	0.12	0.01	0.82	0.03	0
0.01	0.01	0.01	0	0	0	0	0	0	0	0	0	0.02	0	0.02	0.86	0.007
0	0	0.04	0	0	0	0	0	0.04	0.01	0	0	0	0	0	0.09	0.82

Table 3: Test Classification Matrix: Showing that the tested set if classified with few outliers

The Confusion matrix obtained from classifiers and based on tables 4, 5 below there is a clear glance that there could be a chance of over-fitting as the data points are fitting very well in training phase but on the test data the model doesn't predict properly the values which is clear from diagonal spread of table 5.

96	1	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0
2	90	1	5	0	0	0	0	0	0	1	0	0	0	1	0	0
1	0	97	0	0	0	2	0	0	0	0	0	0	0	0	0	0
2	2	0	91	0	0	1	3	0	1	0	0	0	0	0	0	0
0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0
0	0	3	0	0	0	97	0	0	0	0	0	0	0	0	0	0
0	0	1	2	0	0	0	91	0	0	0	6	0	0	0	0	0
0	0	0	0	3	0	0	0	97	0	0	0	0	0	0	0	0
1	1	0	0	0	1	0	0	0	95	2	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	99	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0	97	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	99	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	97	3
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	99

Table 4: Train Confusion Matrix: Showing that the training set if indented properly in a diagonal

13	19	4	23	1	5	2	4	3	13	2	2	1	4	4	0	0
12	19	2	16	1	10	7	9	0	10	3	2	3	1	1	2	2
2	2	62	0	0	3	26	0	1	1	0	1	1	0	1	0	0
8	11	2	37	0	4	6	13	0	5	3	4	1	3	3	0	0
0	0	0	0	90	0	0	0	2	0	0	0	4	2	0	2	0
9	6	7	3	3	36	9	3	3	8	11	0	0	0	0	0	2
0	2	14	2	0	2	71	1	0	0	3	2	0	0	0	1	2
1	4	0	6	1	0	4	46	0	0	0	37	0	0	0	1	0
0	3	2	0	16	1	0	0	64	0	0	0	1	0	4	5	4
5	15	0	10	0	3	1	3	1	28	20	5	5	1	0	1	2
0	2	0	2	0	11	0	0	1	10	70	3	0	0	0	0	1
0	0	0	0	0	0	0	7	0	0	0	93	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	72	11	15	1	0
0	0	0	0	0	0	0	0	0	0	0	0	2	98	0	0	0
0	1	0	0	0	0	0	0	0	1	0	0	12	1	82	3	0
1	1	1	0	0	0	0	0	0	0	0	0	2	0	2	86	7
0	0	4	0	0	0	0	0	4	1	0	0	0	0	0	9	82

Table 5: Test Confusion Matrix: Showing that the tested set if indented properly in a diagonal

3.1.3 TAIJI Dataset

TAIJI dataset has huge set of features say 500 features for a single data point and the classifier used is 'one-vs-all'. There are results based on plots for features 1,7 and also the classification matrix and confusion matrix based on all features. From the figure 3, we can say that there are very less outliers and due to many features and classes the outliers class and definition are not visualised.
And from table **??** we can show that the confusion matrix before train is decently indented diagonally with proper zeros on the other terms of a diagonal after the training part is also put to test and results in the values of diagonal indentation and even here there are valued amounts of zeros in non-diagonal areas, But the values in diagonal region are far more greater than non-diagonal regional values from matrix.
When we consider the classification matrix of train with test the classification at train stage is equally fitting alongside of testing hence chance of over fitting should not be seen based on accuracy in the later tables.
As the features differs than wine and wallpaper, which means this dataset has more features than wine but less features than wallpaper,
in this are compared to less than the wallpaper and more than wine the over-fitting is less than the wallpaper whereas the outliers are lesser compared to the wine dataset.

Here, we can see that the diagonal orientation of the matrices here are well arranged and the outliers away from the diagonal are also not that prominent as the wallpaper.

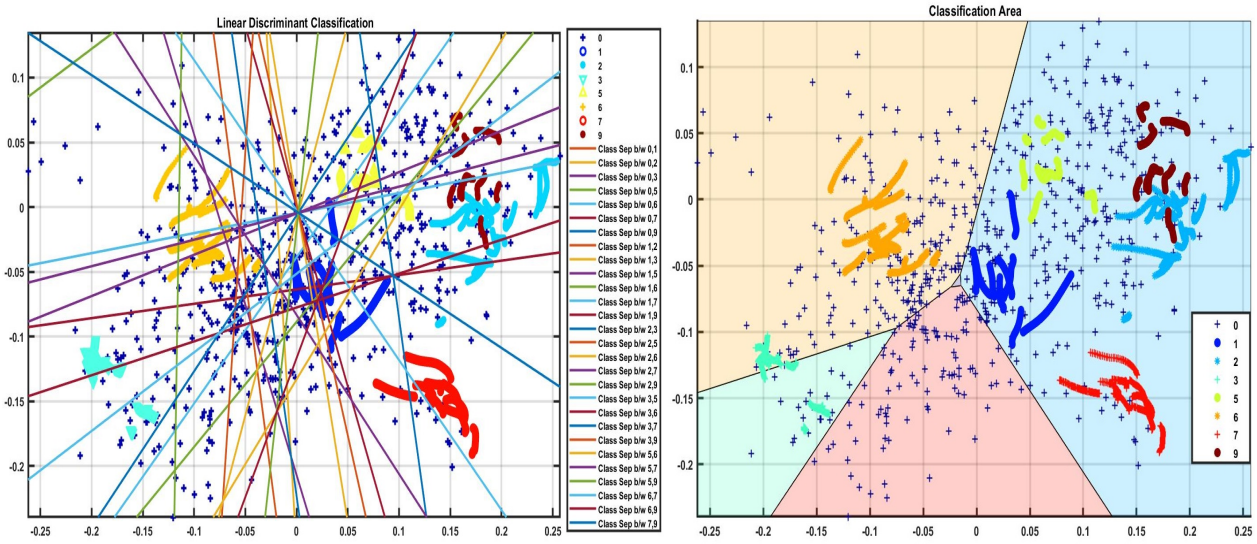

Figure 3: Plots for the Linear Discriminant classification using least squares for TAIJI for features 1, 7. The lines defines the classification boundaries and each color of point is for each class, towards the right each classification area is of each class.

0.69	0.035	0.061	0.033	0.037	0.05	0.03	0.058
0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1

Table 6: Train Classification Matrix: Showing that the training set if classified with rare outlier occurence

0.430	0.053	0.083	0.045	0.176	0.071	0.035	0.108
0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1

Table 7: Test Classification Matrix: Showing that the training set if classified with rare outlier occurence

1220	63	109	59	66	92	54	104
0	1066	0	0	0	0	0	0
0	0	2132	0	0	0	0	0
0	0	0	1066	0	0	0	0
0	0	0	0	1066	0	0	0
0	0	0	0	0	2132	0	0
0	0	0	0	0	0	1066	0
0	0	0	0	0	0	0	1066

Table 8: Train Confusion Matrix: Showing that the tested set if indented properly in a diagonal

259	32	50	27	106	43	21	65
0	369	0	0	0	0	0	0
0	0	738	0	0	0	0	0
0	0	0	369	0	0	0	0
0	0	0	0	369	0	0	0
0	0	0	0	0	738	0	0
0	0	0	0	0	0	369	0
0	0	0	0	0	0	0	369

Table 9: Test Confusion Matrix: Showing that the tested set if indented properly in a diagonal

3.1.4 Comparision for three Datasets

When the three datasets are compared the test accuracy of Wine and Taiji are better and are almost equal to the train accuracy which states that the data is not overfitting, it shows that the data fits properly in model and hence results with same outcome of predictions. Which states that these two datsets doesn't have a chance of overfitting.
However, from the dataset's confusion and classification matrices for each dataset it was already clear that the Wallpaper dataset would result for a risk of overfiting, hence from this accuracy rates it is clear that the data is overfitting for train and is not as accurate as fitted model for prediction while testing.

Dataset	Train Acc	Train Std	Test Acc	Test Std
Wine	1	0	0.972	0.050
Wallpaper	0.967	0.0328	0.617	0.269
Taiji	0.961	0.109	0.928	0.201

Table 10: Train and Test accuracy and standard deviations for WINE, Wallpaper and Taiji datasets using Least Squares method for features 1, 7.

3.2 Linear Discriminant using Fisher Projection

The Fisher Projection is utilized to model a linear discriminant using KNN classifier for predicting the classes and their accuracy with the visualization of plots for classification along with confusion matrices.

K Optimization:
I have used a ideal equation for all the real world data oriented algorithms for selecting K for the features given in dataset.
Formulation for K is

$$K = \lfloor \sqrt{count_{features}} \rceil$$

Where $count_features$ is number of features in a dataset.

3.2.1 WINE Dataset

WINE dataset has 13 features for a data point and the classifier used is KNN Classifier with derived rounded k value equal to 4. There are results based on plots for features 1,7 and also the classification matrix and confusion matrix based on all features.
From table 11 we can show that the confusion matrix after test is performed on model is decently indented diagonally and there would be less score for overfitting and outliers than the least squares method.
The optimal K values based on the formula in section 3.2 is:

$$K = \lfloor \sqrt{13} \rceil$$
$$k = 4$$

1	0	0
0	1	0
0	0.292	0.71

Train Classification Matrix

1	0	0
0	0.886	0.114
0	0.25	0.75

Test Classification Matrix

30	0	0
0	36	0
0	7	17

Train Confusion Matrix

29	0	0
0	31	4
0	6	18

Test Confusion Matrix

Table 11: Confusion and Classification matrices for the WINE dataset using KNN for k = 4

3.2.2 WALLPAPER Dataset

WALLPAPER dataset has huge set of features say 500 features for a single data point and the classifier used is 'one-vs-all'. There are results based on plots for features 1,7 and also the classification matrix and confusion matrix based on all features.
And from table 13 we can show that the confusion matrix before train is decently indented diagonally with proper zeros on the other terms of a diagonal and after the training part is put to test and results in the values of diagonal indentation and there is valued amounts of zeros in non-diagonal areas, But the values in diagonal region are far more greater than non-diagonal regional values from matrix.
When we consider the classification matrix of train with test the classification at train stage is equally fitting to testing hence no chance of over fitting may be seen based on accuracy in the later tables.
The calculated K value here is:

$$K = \lfloor \sqrt{500} \rceil$$
$$k = 22$$

0.97	0.02	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0
0.02	0.94	0	0.03	0	0	0	0	0	0.01	0	0	0	0	0	0	0
0.01	0	0.96	0.01	0	0	0.02	0	0	0	0	0	0	0	0	0	0
0.01	0	0	0.96	0	0	0	0.03	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0.01	0	0	0.99	0	0	0	0	0	0	0	0	0	0	0
0	0	0.02	0	0	0	0.98	0	0	0	0	0	0	0	0	0	0
0	0	0	0.02	0	0	0	0.95	0	0	0	0.03	0	0	0	0	0
0	0	0	0	0.01	0	0	0	0.99	0	0	0	0	0	0	0	0
0	0.01	0	0	0	0	0	0.01	0	0.97	0.01	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0.99	0.01	0	0	0	0	0
0	0	0	0	0	0	0	0.01	0	0	0	0.99	0	0	0	0	0
0	0	0	0	0.01	0	0	0	0	0	0	0	0.98	0	0.01	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.98	0.02
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 12: Train Classification Matrix: Shows the minimal outliers and overfitting for KNN classifier for K = 22 for Wallpaper dataset.

0.26	0.25	0.02	0.23	0	0.06	0	0.01	0	0.14	0.03	0	0	0	0	0	0
0.22	0.25	0.02	0.14	0	0.09	0.03	0.04	0	0.19	0.02	0	0	0	0	0	0
0.02	0.03	0.68	0.02	0	0.02	0.23	0	0	0	0	0	0	0	0	0	0
0.13	0.12	0.02	0.46	0	0.02	0.05	0.09	0	0.09	0.01	0.01	0	0	0	0	0
0	0	0	0	0.94	0	0	0	0	0	0	0	0.06	0	0	0	0
0.09	0.1	0.04	0	0	0.56	0.06	0	0	0.08	0.007	0	0	0	0	0	0
0.02	0.02	0.18	0.02	0	0.02	0.69	0.02	0	0.01	0.02	0	0	0	0	0	0
0.01	0.01	0	0.007	0	0	0.03	0.68	0	0	0	0.2	0	0	0	0	0
0	0	0.01	0	0.1	0	0	0	0.81	0	0	0	0	0	0.04	0.03	0.01
0.13	0.21	0	0.05	0	0.01	0	0.06	0	0.35	0.16	0.03	0	0	0	0	0
0.01	0	0	0.02	0	0.007	0	0	0	0.17	0.73	0	0	0	0	0	0
0	0	0	0	0	0	0	0.007	0	0	0	0.93	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0.96	0.02	0.02	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0.05	0	0.93	0.02	0
0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0.01	0.93	0.05
0	0	0	0	0	0	0	0	0.04	0	0	0	0	0	0	0.06	0.9

Table 13: Test Classification Matrix: Shows the minimal outliers and overfitting for KNN classifier for K = 22 for Wallpaper dataset.

97	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
2	94	0	3	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	96	1	0	0	2	0	0	0	0	0	0	0	0	0	0
1	0	0	96	0	0	0	3	0	0	0	0	0	0	0	0	0
0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	99	0	0	0	0	0	0	0	0	0	0	0
0	0	2	0	0	0	98	0	0	0	0	0	0	0	0	0	0
0	0	0	2	0	0	0	95	0	0	0	3	0	0	0	0	0
0	0	0	0	1	0	0	0	99	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	1	0	97	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	99	1	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	99	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	98	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	98	2
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100

Table 14: Train Confusion Matrix: Shows the minimal outliers and overfitting for KNN classifier for K = 22 for Wallpaper dataset.

26	25	2	23	0	6	0	1	0	14	3	0	0	0	0	0	0
22	25	2	14	0	9	3	4	0	19	2	0	0	0	0	0	0
2	3	68	2	0	2	23	0	0	0	0	0	0	0	0	0	0
13	12	2	46	0	2	5	9	0	9	1	1	0	0	0	0	0
0	0	0	0	94	0	0	0	0	0	0	0	6	0	0	0	0
9	10	4	0	0	56	6	0	0	8	7	0	0	0	0	0	0
2	2	18	2	0	2	69	2	0	1	2	0	0	0	0	0	0
1	1	0	7	0	0	3	68	0	0	0	20	0	0	0	0	0
0	0	1	0	10	0	0	0	81	0	0	0	0	0	4	3	1
13	21	0	5	0	1	0	6	0	35	16	3	0	0	0	0	0
1	0	0	2	0	7	0	0	0	17	73	0	0	0	0	0	0
0	0	0	0	0	0	0	7	0	0	0	93	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	96	2	2	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	5	0	93	2	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	93	5
0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	6	90

Table 15: Test Confusion Matrix: Shows the minimal outliers and overfitting for KNN classifier for K = 22 for Wallpaper dataset.

3.2.3 TAIJI Dataset

Taiji dataset has huge set of features say 64 features for a single data point and the classifier used is KNN. There are results based on plots for features 1,7 and also the classification matrix and confusion matrix based on all features.
And from table 19 we can show that the confusion matrix before train is decently indented diagonally with proper zeros on the other terms of a diagonal after the training part is also put to test and results in the values of diagonal indentation and even here there are valued amounts of zeros in non-diagonal areas, But the values in diagonal region are far more greater than non-diagonal regional values from matrix.

When we consider the classification matrix of train with test the classification at train stage is equally fitting alongside of testing hence chance of over fitting is not seen based on accuracy in the later tables. The optimal K values based on the formula in section 3.2 is:

$$K = \lfloor \sqrt{64} \rceil$$
$$k = 8$$

0.848	0.002	0.0036	0.0017	0.0018	0.0025	0.0016	0.0016
0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1

Table 16: Train Classification Matrix: Shows the minimal outliers and overfitting for KNN classifier for K = 8 for Wallpaper dataset.

0.789	0.00182	0.003	0.0023	0.0063	0.0036	0.0018	0.0022
0.2033	0.797	0	0	0	0	0	0
0.0037	0	0.963	0	0	0	0	0
0	0	0	1	0	0	0	0
0.111	0	0	0	0.8895	0	0	0
0.0067	0	0	0	0	0.993	0	0
0.0065	0	0	0	0	0	0.935	0
0	0	0	0	0	0	0	1

Table 17: Test Classification Matrix: Shows the minimal outliers and overfitting for KNN classifier for K = 8 for Wallpaper dataset.

1499	39	65	31	32	44	29	28
0	1066	0	0	0	0	0	0
0	0	2132	0	0	0	0	0
0	0	0	1066	0	0	0	0
0	0	0	0	1066	0	0	0
0	0	0	0	0	2132	0	0
0	0	0	0	0	0	1066	0
0	0	0	0	0	0	0	1066

Table 18: Train Confusion Matrix: Shows the minimal outliers and overfitting for KNN classifier for K = 8 for Wallpaper dataset.

476	11	18	14	38	22	11	13
75	294	0	0	0	0	0	0
27	0	711	0	0	0	0	0
0	0	0	369	0	0	0	0
41	0	0	0	328	0	0	0
5	0	0	0	0	733	0	0
24	0	0	0	0	0	345	0
0	0	0	0	0	0	0	369

Table 19: Test Confusion Matrix: Shows the minimal outliers and overfitting for KNN classifier for K = 8 for Wallpaper dataset.

3.2.4 Comparison for three Datasets

When the three datasets are compared the test accuracy of Wine and Taiji are better and are almost equal to the train accuracy which states that the data is not over-fitting, it shows that the data fits properly in model and hence results with same outcome of predictions. Which states that these two datasets doesn't have a chance of over-fitting. However, from the dataset's confusion and classification matrices for each dataset it was already clear that the Wallpaper dataset would result for a risk of over-fitting, hence from this accuracy rates it is clear that the data is over-fitting for train and is not as accurate as fitted model for prediction while testing.

Dataset	Train Acc	Train Std	Test Acc	Test Std
Wine	0.903	0.168	0.879	0.125
Wallpaper	0.979	0.019	0.710	0.253
Taiji	0.981	0.054	0.921	0.087

Table 20: Train and Test accuracy and standard deviations for WINE, Wallpaper and Taiji datasets using Fisher Method for optimal values of K to be 4, 22, 8 respectively.

4 Relation between Fisher and Least Squares Discriminants

The sum-of-squares error function can be written as shown below,

$$E = \frac{1}{2}\sum_{n=1}^{N}(w^T x_n + w_0 - t_n)^2 \tag{31}$$

Taking the derivative of E w.r.t. to w_0 and w and equating it to zero, we obtain the below formulation.

$$\sum_{n=1}^{N}(w^T x_n + w_0 - t_n) = 0 \tag{32}$$

$$\sum_{n=1}^{N}(w^T x_n + w_0 - t_n)x_n = 0 \tag{33}$$

from equation (32) and by using the target coding scheme for t_n, we can derive the bias as shown below,

$$w_0 = -w^T m \tag{34}$$

where

$$\sum_{n=1}^{N} t_n = N_1 \frac{N}{N_1} - N_2 \frac{N}{N_2} = 0 \tag{35}$$

m is the mean of total dataset given by the below formula,

$$m = \frac{1}{N} \sum_{n=1}^{N} x_n = \frac{1}{N}(N_1 m_1 + N_2 m_2) \tag{36}$$

Again by using the choice of t_n, the equation (33) becomes,

$$(S_w + \frac{N_1 N_2}{N} S_B) w = N(m_1 - m_2) \tag{37}$$

S_W and S_B are given by equations (26), and (27). $S_B w$ is always in the direction of $(m_2 - m_1$, thus by ignoring irrelevant scale factors we can write

$$w \propto S_W^{-1}(m_2 - m_1) \tag{38}$$

From equation 38, we can observe that weight vector coincides with that of Fisher criterion, the new vector x should be classified as belonging to
class C_1 if

$$y(x) = w^T(x - m) > 0$$

class C_2 otherwise
Observation: From the experiment from the code I have observed that the accuracy for the Wine dataset for both least squares and Fisher projections as follows:

Classification	Train Acc	Train Std	Test Acc	Test Std
Least Squares	0.76	0.0205	0.812	0.051
Fisher Projection	0.967	0.023	0.634	0.36

Table 21: Train and Test accuracy and standard deviations for WINE using Least Squares and Fisher Method for two classes

5 Conclusion

- While performing least squares with one vs all classifiers to model a linear discriminant, it was observed that the method worked fine for datasets with relatively small number of classes (as in the case of Wine and Taiji). However, when the number of classes increased, the model incurred the problem of overfitting as seen in the case of the Wallpaper dataset.

- The failure of the least squares method can be explained due to assumption of the least square to Gaussian distribution whereas the binary target vectors are not Gaussian.
- The Fischer projection for linear discriminant model with KNN classifier gave better results than the least squares method. Hence, it can be concluded that by reducing the dimensions the accuracy rate of prediction can be increased.
- Finally, comparison on Wine dataset for two classes, it was observed that Fischer criterion is indeed a special case of least squares method.

References

[1] C. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006. 4