1. **Least squares:**

The least squares method is based on the criterion of minimizing mean squared error. Let’s consider the 2-class case:

Let C0 and C1 denote the two classes. Thus, if y(i) = 0 then Xi ∈ C0 and if y(i) = 1 then Xi ∈ C1

Let n0 and n1 denote the number of examples(features) of each class. (n = n0 + n1)

For any W, y are the one-dimensional data that we get after projection.

We now determine the parameter matrix , by minimizing a sum-of-squares error function:

Setting the derivative with respect to , to zero, and rearranging, we then obtain the  
solution for , in the form:

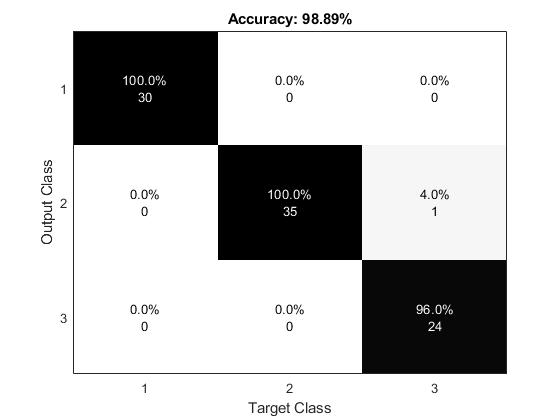
=

where is the pseudo-inverse of the matrix x. We then obtain the discriminant function in the form.

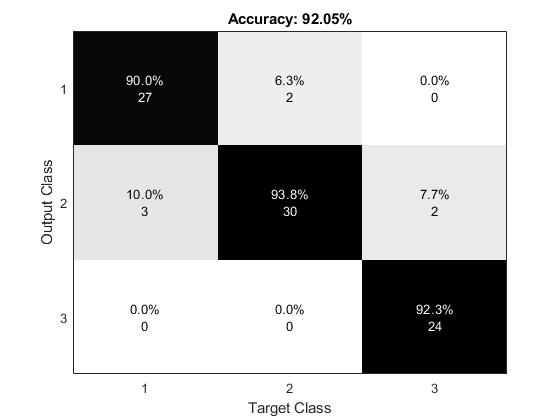
**Following are the Confusion matrices for different datasets for both training**

**and test sets:**

1. **Wine Data**

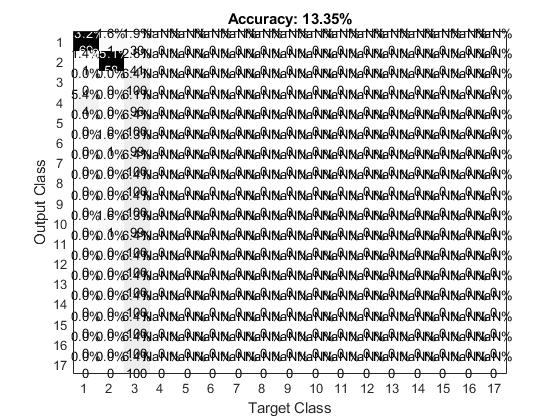


**Training Data Set**

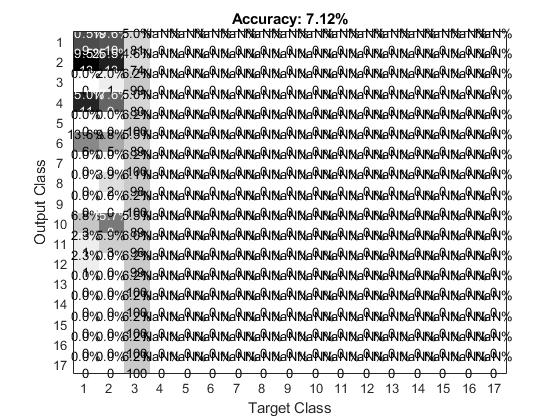


**Test Data Set**

1. **Wallpaper Data**

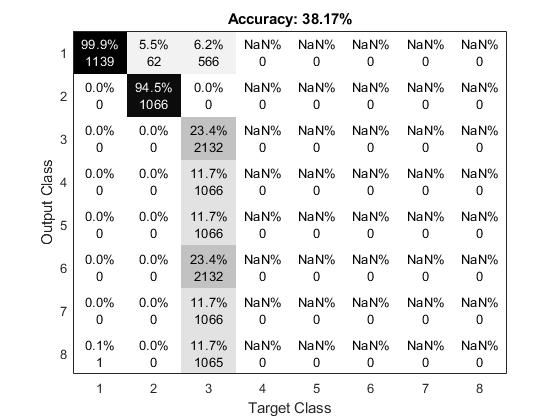


**Training Data Set**

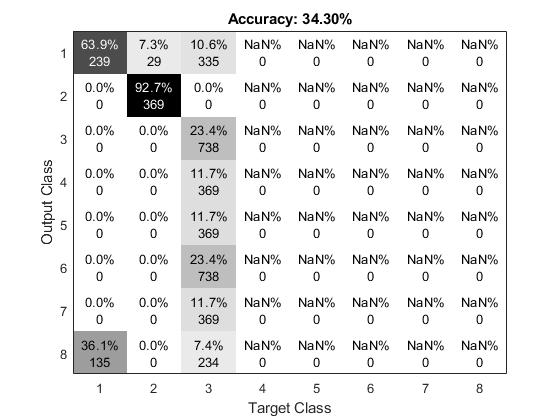


**Test Data Set**

1. **Taiji Data**

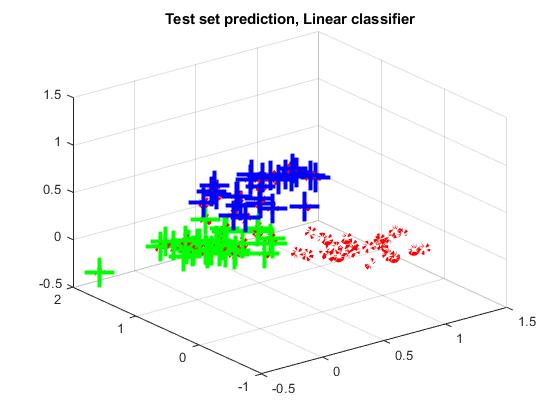


**Training Data Set**



**Test Data Set**

Below is the 3-Dimensional figure for Wine testing:



1. **Fisher LDA:**

The least squares method is based on the criterion of minimizing mean squared error. Let’s consider the 2-class case:

Let C0 and C1 denote the two classes. Thus, if y(i) = 0 then Xi ∈ C0 and if y(i) = 1 then Xi ∈ C1

Let n0 and n1 denote the number of examples(features) of each class. (n = n0 + n1)

For any W, z are the one-dimensional data that we get after projection.

Let and be the means of data from the two classes:

we want a W that maximizes

However, we have to make this scale independent.

Also, the distance between means should be viewed relative to the variances.

Thus, we define:

These give us the variances (upto a factor) of the two classes in the projected data.

We want large separation between and relative to the variances.

Hence, we can take our objective to be to maximize:

=

=

=

=

is a d × d matrix

We can similarly write and also as quadratic forms.

We know

is also d × d matrix and is called within class scatter matrix

Thus,

=

We want to find a W that maximizes :

is not affected by scaling of W.

Maximizing ratio of quadratic forms is a standard optimization problem

Differentiating w.r.t. W and equating to zero, we get:

- \* = 0

This Implies, is in the same direction as

Thus, any maximizer of has to satisfy = λ for some constant λ

This is known as the generalized eigen value problem.

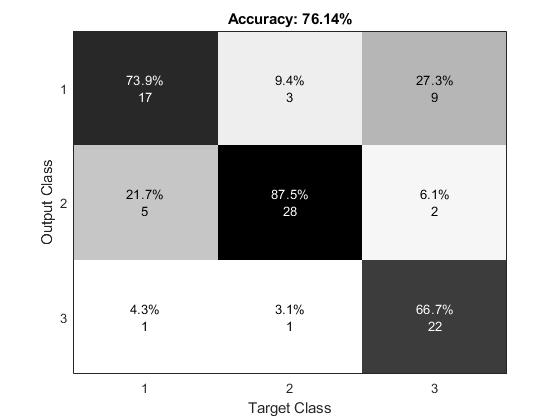
By solving the generalized eigen value problem we can find the best direction W.

**Following are the Confusion matrices for different datasets for both training and test sets:**

1. **Wine dataset:**

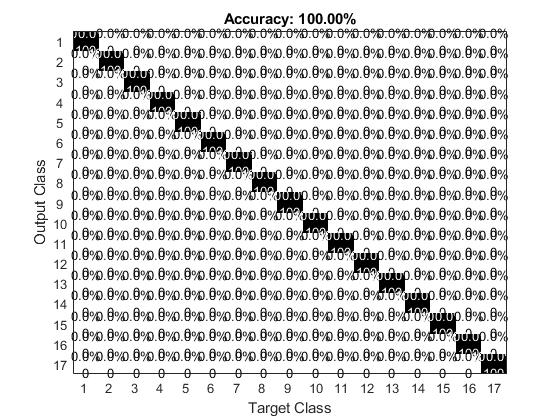


**Training Data Set**

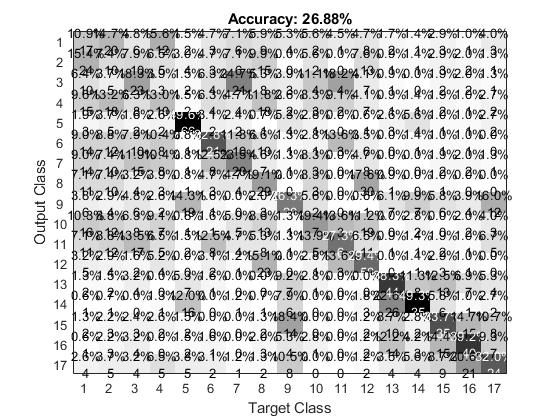


**Test Data Set**

**b. Wallpaper Data Set:**

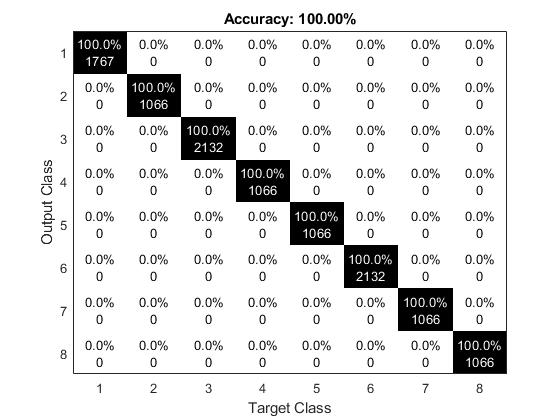


**Training Data Set**

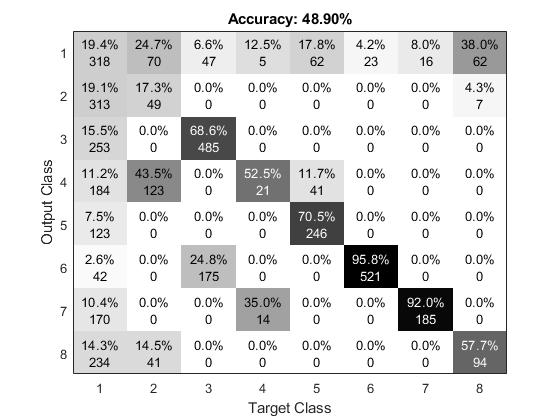


**Test Data Set**

**c. Taiji Data Set:**



**Training Data Set**



**Test Data Set**

1. **Fisher LDA as special case of Linear Regression(for 2 classes):**

Let’s say the targets for class C1 be N/N1, where N1 is the number of patterns in class C1, and N is the total number of patterns.

For class *C*2, we shall take the targets to be *-N/N*2, where *N*2 is the number of patterns in class *C*2

The sum-of-squares error function can be written:

Setting the derivatives of *E* with respect to *w*0 and **w** to zero, we obtain respectively  
  
 = 0

making use of our choice of target coding scheme for the , we obtain an expression for the bias in the form.

where we have used

and where **m** is the mean of the total data set and is given by:

Thius we can write:

**Below are the confusion matrices for wine data after removing class 3:**



**Least Square Test Data Set**



**Fisher LDA Test Data Set**