Two Classes

***Abstract*—This document contains theory involved in curve fitting.**

1 Objective

The objective is to use discriminant function to classify.

2 Generate Dataset

Create a function of the form

$$y = \mathbf{w}^T\mathbf{x} + w_0 \tag{2.0.1}$$

This set consists of N samples of input data i.e. x expressed as shown below

$$x = \begin{pmatrix} x_1, x_2, .., x_N \end{pmatrix}^T \tag{2.0.2}$$

which give the corresponding values of y denoted as

$$y = \begin{pmatrix} y_1, y_2, .., y_N \end{pmatrix}^T \tag{2.0.3}$$

The corresponding values of y are generated from the Eq (2.0.1). The generated input matrix would look like

$$\mathbf{F} = \begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^{N-1} \\ 1 & x_1 & x_1^2 & \dots & x_1^{N-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{N-1} \\ \vdots & & \vdots & & \vdots \\ 1 & \dots & \dots & \dots & x_N^{N-1} \end{pmatrix} \tag{2.0.4}$$

3 Linear Discriminant Function (2 Classes)

The simplest representation of linear discriminant function is

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 \tag{3.0.1}$$

where w_0 is the bias. The negative of bias is sometimes referred to as threshold.

An input vector $\mathbf{x}$ is assigned to class C_1 if $y(\mathbf{x}) >= 0$ and to class C_2 otherwise.

$y(\mathbf{x}) = 0$ defines the decision boundary, which corresponds to $(D-1)$ dimensional hyperplane within the D dimensional input space.

The normal distance from origin to the decision surface is

$$\frac{\mathbf{w}^T\mathbf{x}}{||\mathbf{w}||} = \frac{w_0}{||\mathbf{w}||} \tag{3.0.2}$$

The value of $y(\mathbf{x})$ gives us the measure of the perpendicular distance r from the point $\mathbf{x}$ to the decision surface.

Consider an arbitrary point $\mathbf{x}$ and $\mathbf{x}_\perp$ is its orthogonal projection onto the decision surface

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{||\mathbf{w}||} \tag{3.0.3}$$

we have

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 \tag{3.0.4}$$

$$y(\mathbf{x}_\perp) = \mathbf{w}^T\mathbf{x}_\perp + w_0 \tag{3.0.5}$$

Using Eq (3.0.3), Eq (3.0.4) and Eq (3.0.5),

$$r = \frac{y(\mathbf{x})}{||\mathbf{w}||} \tag{3.0.6}$$

For a compact notation, we introduce dummy input value $x_0 = 1$ and then define

$$\overline{\mathbf{w}} = (w_0, \mathbf{w}) \tag{3.0.7}$$

$$\overline{\mathbf{x}} = (x_0, \mathbf{x}) \tag{3.0.8}$$

So that,

$$y(\mathbf{x}) = \overline{\mathbf{w}}^T\overline{\mathbf{x}} \tag{3.0.9}$$

In order to find a good projection vector, we need to define a measure of separation between the projections.

Consider there are N_1 points of class C_1 and N_2 points of class C_2

The mean vector of the two classes are

$$\mu_1 = \frac{1}{N_1}\sum_{\mathbf{x}\epsilon C_1}\mathbf{x} \tag{3.0.10}$$

$$\mu_2 = \frac{1}{N_2}\sum_{\mathbf{x}\epsilon C_2}\mathbf{x} \tag{3.0.11}$$

The mean vector in y feature space is

$$\overline{\mu_1} = \frac{1}{N_1}\sum_{y\epsilon C_1} y = \frac{1}{N_1}\sum_{\mathbf{x}\epsilon C_1} \mathbf{w}^T\mathbf{x} = \mathbf{w}^T\mu_1 \quad (3.0.12)$$

$$\overline{\mu_2} = \frac{1}{N_2}\sum_{y\epsilon C_2} y = \frac{1}{N_2}\sum_{\mathbf{x}\epsilon C_2} \mathbf{w}^T\mathbf{x} = \mathbf{w}^T\mu_2 \quad (3.0.13)$$

We could choose the objective function as the distance between the projected means

$$J(\mathbf{w}) = |\overline{\mu_1} - \overline{\mu_2}| = \left|\mathbf{w}^T(\mu_1 - \mu_2)\right| \quad (3.0.14)$$

The distance between the projected means is not very good as it doesn't involve standard deviations with in the classes.

For each class we define the scatter, an equivalent of the variance, as

$$\overline{s_1}^2 = \sum_{y\epsilon C_1}(y - \overline{\mu_1})^2 \quad (3.0.15)$$

$$\overline{s_2}^2 = \sum_{y\epsilon C_2}(y - \overline{\mu_2})^2 \quad (3.0.16)$$

The quantity $(\overline{s_1}^2 + \overline{s_1}^2)$, represents the within-class scatter of the projected inputs.

The Fisher linear discriminant is defined as the linear function $\mathbf{w}^T\mathbf{x}$ that maximizes the objective function

$$J(\mathbf{w}) = \frac{|\overline{\mu_1} - \overline{\mu_2}|^2}{\overline{s_1}^2 - \overline{s_2}^2} \quad (3.0.17)$$

To find the optimum projection $\mathbf{w}*$, we need to express $J(\mathbf{w})$ as an explicit function of $\mathbf{w}$.

We define a measure of the scatter in multivariate feature space $\mathbf{x}$ which are co-variance matrices

$$S_1 = \sum_{x\epsilon C_1}(\mathbf{x} - \mu_1)(\mathbf{x} - \mu_1)^T \quad (3.0.18)$$

$$S_2 = \sum_{x\epsilon C_2}(\mathbf{x} - \mu_2)(\mathbf{x} - \mu_2)^T \quad (3.0.19)$$

$$S_1 + S_2 = S_w \quad (3.0.20)$$

where S_w is called the **within-class** co-variance matrix

The scatter of the projection y can then be expressed as a function of the scatter matrix in feature space $\mathbf{x}$

$$\overline{s_1}^2 = \sum_{y\epsilon C_1}(y - \overline{\mu_1})^2$$

$$= \sum_{x\epsilon C_1}\left(\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\mu_1\right)^2 = \sum_{x\epsilon C_1}\mathbf{w}^T(\mathbf{x} - \mu_1)(\mathbf{x} - \mu_1)^T\mathbf{w}$$

$$= \mathbf{w}^T S_1\mathbf{w} \quad (3.0.21)$$

Similarily,

$$\overline{s_2}^2 = \mathbf{w}^T S_2\mathbf{w} \quad (3.0.22)$$

$$\overline{s_1}^2 + \overline{s_2}^2 = \mathbf{w}^T S_w\mathbf{w} \quad (3.0.23)$$

Similarly, the difference between the projected means can be expressed in terms of the means in the original feature space

$$(\overline{\mu_1} - \overline{\mu_2})^2 = \left(\mathbf{w}^T\mu_1 - \mathbf{w}^T\mu_2\right)^2$$

$$= \mathbf{w}^T(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T\mathbf{w} = \mathbf{w}^T S_B\mathbf{w} \quad (3.0.24)$$

where, $S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$ is called **between-class** covariance matrix.

since S_B is the outer product of two vectors,its rank is at most one.

The Expression of the fisher's criterion in terms of S_w and S_B is

$$J(\mathbf{w}) = \frac{\mathbf{w}^T S_B\mathbf{w}}{\mathbf{w}^T S_w\mathbf{w}} \quad (3.0.25)$$

To find the maximum of $J(\mathbf{w})$ we find the derivation and equate it to zero

$$\frac{d}{d\mathbf{w}}(J(\mathbf{w})) = \frac{d}{d\mathbf{w}}\left(\frac{\mathbf{w}^T S_B\mathbf{w}}{\mathbf{w}^T S_w\mathbf{w}}\right) = 0$$

$$\implies \left(\mathbf{w}^T S_w\mathbf{w}\right)\frac{d}{d\mathbf{w}}\left(\mathbf{w}^T S_B\mathbf{w}\right)$$

$$-\left(\mathbf{w}^T S_B\mathbf{w}\right)\frac{d}{d\mathbf{w}}\left(\mathbf{w}^T S_w\mathbf{w}\right) = 0$$

$$\implies \left(\mathbf{w}^T S_w\mathbf{w}\right)2S_B\mathbf{w} - \left(\mathbf{w}^T S_B\mathbf{w}\right)2S_w\mathbf{w} = 0 \quad (3.0.26)$$

Divide by $\mathbf{w}^T S_w\mathbf{w}$,

$$\left(\frac{\mathbf{w}^T S_w\mathbf{w}}{\mathbf{w}^T S_w\mathbf{w}}\right)S_B\mathbf{w} - \left(\frac{\mathbf{w}^T S_B\mathbf{w}}{\mathbf{w}^T S_w\mathbf{w}}\right)S_w\mathbf{w} = 0$$

$$\implies S_B\mathbf{w} - JS_w\mathbf{w} = 0$$

$$\implies S_w^{-1}S_B\mathbf{w} - J\mathbf{w} = 0 \quad (3.0.27)$$

Solving Eq (3.0.27) gives us,

$$\mathbf{w}^* = argmax\left(\frac{\mathbf{w}^T S_B \mathbf{w}}{\mathbf{w}^T S_w \mathbf{w}}\right) = S_w^{-1}(\mu_1 - \mu_2) \quad (3.0.28)$$

Eq (3.0.28), is known as the Fisher's Linear Discriminant

4 Implementation

We would perform Linear Discriminant Analysis on iris dataset. The Iris flower data set or Fisher's Iris data set is a multivariate data set.

Based on Fisher's linear discriminant model, this data set became a typical test case for many statistical classification techniques in machine learning.

First, we define the LDA class

```
class LDA:
    def __init__(self):
        pass

    def fit(self, X, y):
        pass
```

Now, we define a function fit() and firstly find the mean of for each class and store them in mean_vectors list

```
def fit(self, X, y):
        target_classes = np.unique(y)
        mean_vectors = []

        for cls in target_classes:
            mean_vectors.append(np.mean(X[y
                == cls], axis=0))
```

Since we use the 2 classes method,then simply calculate the B matrix as per the Eq (3.0.12),

```
mu1_mu2 = (mean_vectors[0] - mean_vectors
    [1]).reshape(1, X.shape[1])
B = np.dot(mu1_mu2.T, mu1_mu2)
```

Now, we define an empty array s_matrix to store the co-variance matrix for each class

```
s_matrix = []

for cls, mean in enumerate(mean_vectors):
        Si = np.zeros((X.shape[1], X.shape[1]))
        for row in X[y == cls]:
            t = (row - mean).reshape(1, X.
                shape[1])
            Si += np.dot(t.T, t)
        s_matrix.append(Si)
```

Create the S matrix and loop through the s_matrix list to append all the values.

```
S = np.zeros((X.shape[1], X.shape[1]))
for s_i in s_matrix:
        S += s_i
```

Now, we calculate S^{-1} and $S^{-1}B$. Further we get the eigen values and eigen vectors as follows

```
S_inv = np.linalg.inv(S)

S_inv_B = S_inv.dot(B)

eig_vals, eig_vecs = np.linalg.eig(S_inv_B)
```

Now, we define a function to load the iris dataset for running the experiment.

```
def load_data(cols, load_all=False, head=False):
    iris = sns.load_dataset("iris")

    if not load_all:
        if head:
            iris = iris.head(100)
        else:
            iris = iris.tail(100)

    le = preprocessing.LabelEncoder()
    y = le.fit_transform(iris["species"])

    X = iris.drop(["species"], axis=1)

    if len(cols) > 0:
        X = X[cols]

    return X.values, y
```

Since we need to solve the 2 classes problem, we use only 2 classes from the dataset.

```
cols = ["petal_length", "petal_width"]
X, y = load_data(cols, load_all=False, head=
    True)
print(X.shape)
```

Invoke fit() by passing the X and y, which will return all the eigen vectors.

```
lda = LDA()
W = lda.fit(X, y)
```

The following block plots the projected space.

```
colors = ['red', 'green', 'blue']
```

```
fig, ax = plt.subplots(figsize=(10, 8))
for point, pred in zip(X, y):
    ax.scatter(point[0], point[1], color=colors[pred
        ], alpha=0.3)
    proj = (np.dot(point, W) * W) / np.dot(W.T,
        W)

    ax.scatter(proj[0], proj[1], color=colors[pred],
        alpha=0.3)

plt.show()
```

The plot would look like Fig

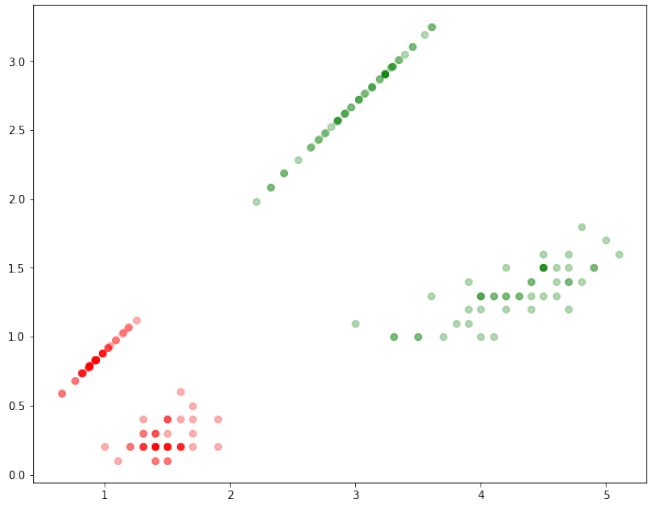

Fig. 0

In speech dataset, since the input array is of higher dimension, we would use mfccs(Mel Frequency Cepstral Coefficients) as these are proved to be more efficient in feature extraction and classification.

Python code:

https://github.com/Hrithikraj2/EE4015_IDP/blob/main/Assignment_8/Assignment_8.ipynb