Fisher Discriminant Analysis

Industrial AI Lab.

Dimensionality Reduction with Label

- Dimensionality reduction with label information (when the ultimate goal is classification/regression)
- PCA ignores label information even if it is available
  - Only chooses directions of maximum variance
- Fisher Discriminant Analysis (FDA) takes into account the label information
  - It is also called Linear Discriminant Analysis (LDA)
- FDA/LDA projects data while preserving class separation
  - Examples from same class are put closely together by the projection
  - Examples from different classes are placed far apart by the projection

Projection onto Line ω

- Linear regression projects each data point
  - assume zero mean, otherwise $x \leftarrow x - \bar{x}$
  - $\omega_0 = 0$

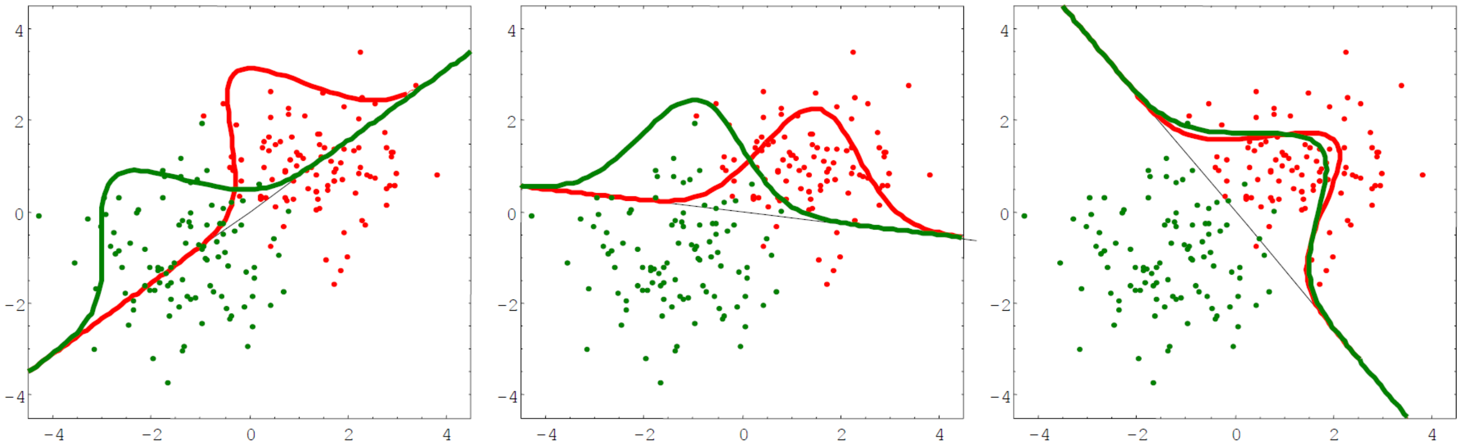
$$\hat{y} = \langle \omega, x \rangle = \omega^T x = \omega_1 x_1 + \omega_2 x_2$$

- Dimension reduction

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \rightarrow \hat{y} \ \text{(scalar)}$$

- Each data point $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ is projected onto ω (projected length on ω direction)
- For a given ω, distribution of the projected points $\{\hat{y}^{(1)}, \cdots, \hat{y}^{(m)}\}$ is specified.
- Question: Which ω is better for classification?

Projection onto Line ω



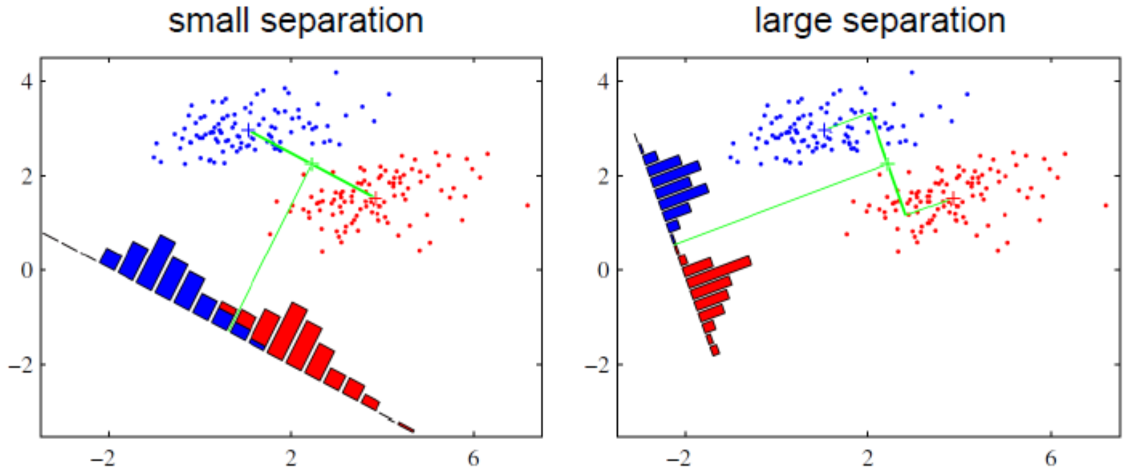
Class C_0	Class C_1
sample mean μ_0 sample variance S_0 $\mu_0 = \frac{1}{n_0} \sum_{x^{(i)} \in C_0}^{n_0} x^{(i)}$ $S_0 = \frac{1}{n_0 - 1} \sum_{x^{(i)} \in C_0}^{n_0} \left(x^{(i)} - \mu_0\right)\left(x^{(i)} - \mu_0\right)^T$	sample mean μ_1 sample variance S_1 $\mu_1 = \frac{1}{n_1} \sum_{x^{(i)} \in C_1}^{n_1} x^{(i)}$ $S_1 = \frac{1}{n_1 - 1} \sum_{x^{(i)} \in C_1}^{n_1} \left(x^{(i)} - \mu_1\right)\left(x^{(i)} - \mu_1\right)^T$
Projected space $E\left[\hat{y} \mid x \in C_0\right] = \mu_0^T \omega$ $\mathrm{var}\left[\hat{y} \mid x \in C_0\right] = \omega^T S_0 \omega$	Projected space $E\left[\hat{y} \mid x \in C_1\right] = \mu_1^T \omega$ $\mathrm{var}\left[\hat{y} \mid x \in C_1\right] = \omega^T S_1 \omega$

Fisher Discriminant Analysis

- Find ω so that when projected onto ω,
  - the classes are maximally separated (maximize distance between classes)
  - Each class is tight (minimize variance of each class)

$$\max_{\omega} \frac{(\text{seperation of projected means})^2}{\text{sum of within class variances}}$$

$$\Longrightarrow \max_{\omega} \frac{\left(\mu_0^T\omega - \mu_1^T\omega\right)^2}{n_0\omega^T S_0\omega + n_1\omega^T S_1\omega}$$



Fisher Discriminant Analysis

$$\omega = \arg\max_{\omega} \left\{ \frac{\left((\mu_0^T - \mu_1^T)\omega\right)^2}{n_0 \omega^T S_0 \omega + n_1 \omega^T S_1 \omega} \right\}$$

$$J(\omega) = \frac{\left((\mu_0^T - \mu_1^T)\omega\right)^2}{\omega^T (n_0 S_0 + n_1 S_1)\omega} = \frac{(m^T \omega)^2}{\omega^T \Sigma \omega}$$

$m \equiv \mu_0 - \mu_1$

$\Sigma \equiv n_0 S_0 + n_1 S_1 = R^T R$ We can always write Σ like this, where R is a "square root" matrix

$u \equiv R\omega \rightarrow \omega = R^{-1} u$ Using R, change the coordinate systems from ω to u

Fisher Discriminant Analysis

$$J(u) = \frac{\left(m^T R^{-1} u\right)^2}{\omega^T R^T R \omega} = \frac{\left(\left(R^{-T} m\right)^T u\right)^2}{u^T u} = \left(\left(R^{-T} m\right)^T \frac{u}{\|u\|}\right)^2$$

$$J(u) = \left(\left(R^{(-T)} m\right)^T \frac{u}{\|u\|}\right)^2 \text{ is maximum when } u = a\, R^{-T} m$$

Fisher Discriminant Analysis

- Why?
  - Dot product of a unit vector and another vector is maximum when the two have the same direction.

$$u = aR^{-T}m = aR^{-T}(\mu_0 - \mu_1)$$
$$\omega = R^{-1}u = aR^{-1}R^{-T}(\mu_0 - \mu_1) = a\left(R^T R\right)^{-1}(\mu_0 - \mu_1) = a\Sigma^{-1}(\mu_0 - \mu_1)$$

$$\therefore\ \omega = a(n_0 S_0 + n_1 S_1)^{-1}(\mu_0 - \mu_1)$$

Python Code

```
import numpy as np
import matplotlib.pyplot as plt

% matplotlib inline
```

```
#training data set generation

n0 = 200
n1 = 200

mu = [0, 0]
sigma = [[0.9, -0.4],
         [-0.4, 0.3]]

x0 = np.random.multivariate_normal([2.5,2.5], sigma, n0).T        # data in class 0
x1 = np.random.multivariate_normal([1,1], sigma, n1).T            # data in class 0

print(x0.shape)

x0 = np.asmatrix(x0)
x1 = np.asmatrix(x1)
```

```
(2, 200)
```

Fisher Discriminant Analysis

$$u = aR^{-T}m = aR^{-T}(\mu_0 - \mu_1)$$

$$\omega = R^{-1}u = aR^{-1}R^{-T}(\mu_0 - \mu_1) = a\left(R^T R\right)^{-1}(\mu_0 - \mu_1) = a\Sigma^{-1}(\mu_0 - \mu_1)$$

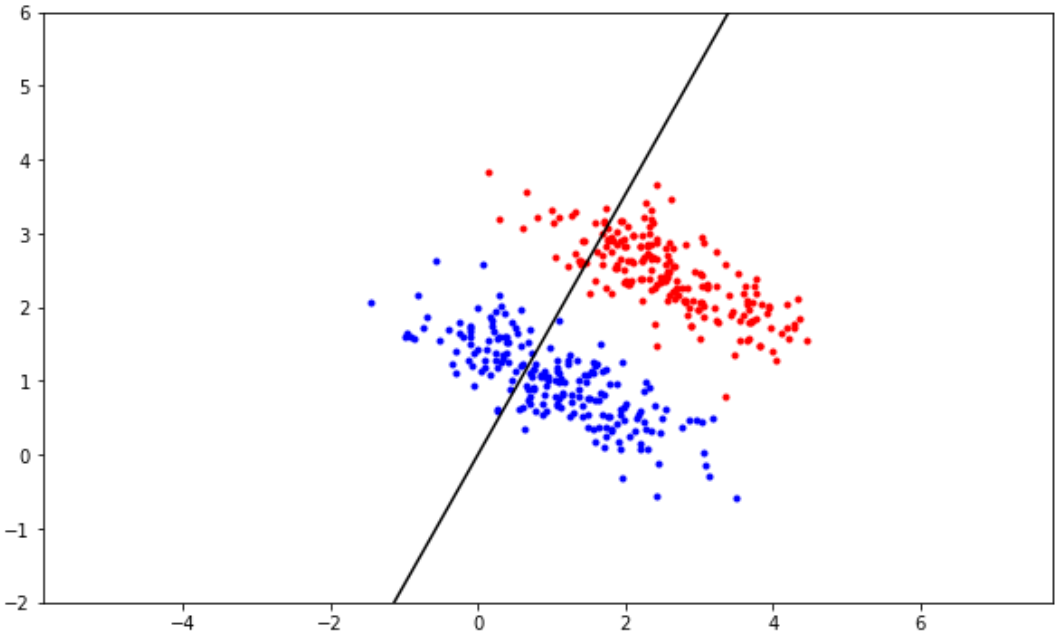
$$\therefore\ \omega = a(n_0 S_0 + n_1 S_1)^{-1}(\mu_0 - \mu_1)$$

```
mu0 = np.mean(x0, axis=1)
mu1 = np.mean(x1, axis=1)

S0 = 1/(n0-1) * (x0-mu0)*(x0-mu0).T
S1 = 1/(n1-1) * (x1-mu1)*(x1-mu1).T

w = (n0*S0 + n1*S1).I * (mu0-mu1)
print(w)
```

```
[[ 0.02432795]
 [ 0.04290126]]
```

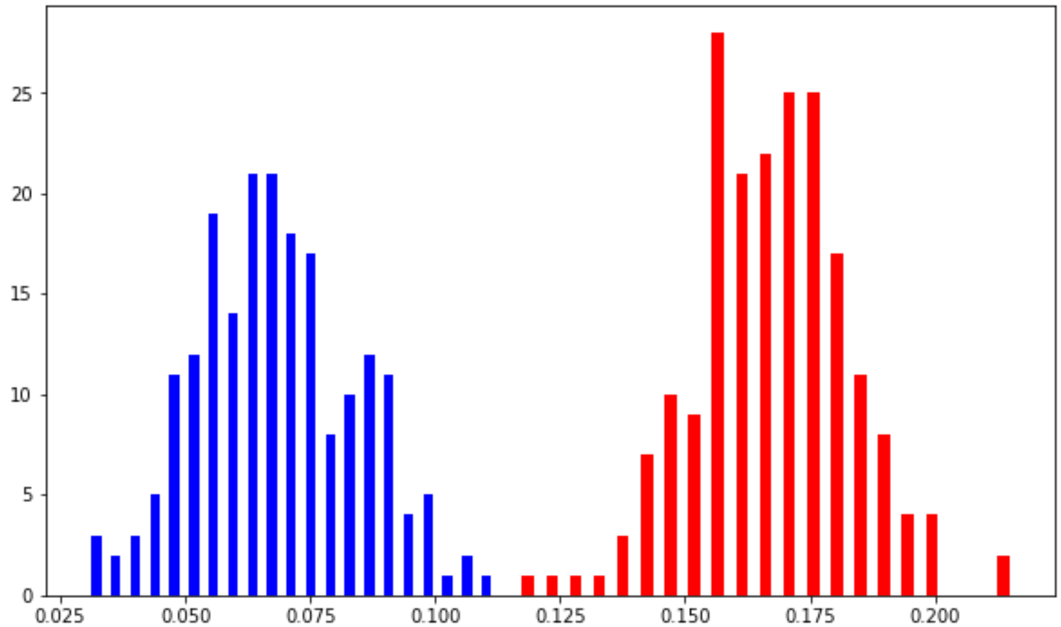
Projection Line

```
plt.figure(figsize = (10, 6))
plt.plot(x0[0,:],x0[1,:],'r.')
plt.plot(x1[0,:],x1[1,:],'b.')

xp = np.arange(-4, 6, 0.1)
yp = w[1,0]/w[0,0]  * xp
plt.plot(xp, yp, 'k')
plt.axis('equal')
plt.ylim([-2, 6])
plt.show()
```



Histogram

```
y1 = x0.T*w
y2 = x1.T*w

plt.figure(figsize = (10, 6))
plt.hist(y1, 21, color='r', rwidth=0.5)
plt.hist(y2, 21, color='b', rwidth=0.5)
plt.show()
```



Different ω (y axis)

```
w = np.array([[0],[1]])
w = np.asmatrix(w)

plt.figure(figsize = (10, 6))
plt.plot(x0[0,:],x0[1,:],'r.')
plt.plot(x1[0,:],x1[1,:],'b.')
plt.axvline(0, color='k')
plt.axis('equal')
plt.ylim([-2, 6])
plt.show()

y1 = x0.T*w
y2 = x1.T*w

plt.figure(figsize = (10, 6))
plt.hist(y1, 21, color='r', rwidth=0.5)
plt.hist(y2, 21, color='b', rwidth=0.5)
plt.show()
```