# Lecture 18

LDA with the assumption that the covariance matrix is a constant.

Thus the decision function is a function of a linear combination of the observations:

$$\mathbf{w}^{\mathbf{T}}\mathbf{x} = \frac{1}{2}\mathbf{w}^{\mathrm{T}}(\mathbf{m_1} + \mathbf{m_2}), \text{ where } \mathbf{w} = \mathbf{C}^{-1}(\mathbf{m_1} - \mathbf{m_2})$$

Consistent with Fisher's linear discriminant with projection:

$$\mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1), \text{ where } \mathbf{S}_{\mathrm{W}} = 2\boldsymbol{C}$$

and $\mathbf{S}_{\mathrm{W}}$ is the total *within-class* covariance matrix, given by

$$\mathrm{S_W} = \sum_{n \in \mathcal{C}_1} (\mathrm{x}_n - \mathrm{m}_1)(\mathrm{x}_n - \mathrm{m}_1)^{\mathrm{T}} + \sum_{n \in \mathcal{C}_2} (\mathrm{x}_n - \mathrm{m}_2)(\mathrm{x}_n - \mathrm{m}_2)^{\mathrm{T}}.$$

## Example

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \mathbf{m_1} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}, \mathbf{m_2} = \begin{bmatrix} 9 \\ 9 \end{bmatrix},$$

$$\boldsymbol{C}_1 = \boldsymbol{C}_2 = \boldsymbol{C} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \qquad \boldsymbol{C}^{-1} = \frac{1}{3}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$$

$$\mathbf{w} = \mathbf{C}^{-1}(\mathbf{m_1} - \mathbf{m_2}) = \frac{1}{3}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}\begin{bmatrix} -6 \\ -6 \end{bmatrix} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

$$\frac{1}{2}\mathbf{w}^{\mathrm{T}}(\mathbf{m_1} + \mathbf{m_2}) = \frac{1}{2}\begin{bmatrix} -2 & -2 \end{bmatrix}\begin{bmatrix} 12 \\ 12 \end{bmatrix} = -24$$

The decision boundary is $\mathbf{w}^{\mathbf{T}}\mathbf{x} = \frac{1}{2}\mathbf{w}^{\mathrm{T}}(\mathbf{m_1} + \mathbf{m_2})$

$$x_1 + x_2 = 12$$

http://www.ece.uah.edu/~dwpan/course/ee610/code/Discriminant%20Analysis/fitcdiscr_demo.m

# fitcdiscr

Fit discriminant analysis classifier

```
>> Model = fitcdiscr(data, label);
Model.DiscrimType

ans =

   'linear'
```

```
% Class 1
m1 = [3, 3]';   % Mean vector
cov1 = [2 1; 1 2]; % Covariance matrix
rng default  % For reproducibility
r1 = mvnrnd(m1,cov1,N);
```

```
% Generate data entries for Class 2
m2 = [9, 9]';
cov2 = [2,1; 1,2];
rng default  % For reproducibility
r2 = mvnrnd(m2,cov2,N);
```

```
>> K = Model.Coeffs(1,2).Const
L = Model.Coeffs(1,2).Linear

K =

  24.2022


L =

  -2.0036
  -2.0380
```

$$-2.0036 x_1 - 2.0380 x_2 + 24.2022 = 0$$

```
>> mean(data_C1)

ans =

   2.9539    3.0221
```

```
>> cov(data_C1)

ans =

   1.9959    0.9819
   0.9819    1.9787
```

```
>> K = Model.Coeffs(2,1).Const

K =

 -24.2022

>> L = Model.Coeffs(2,1).Linear

L =

   2.0036
   2.0380
```

```
>> resubLoss (Model)

ans =

   0.0090
```

- LDA can be viewed as a minimum-distance classifier, with the distance being the Mahalanobis distance (between a point $\mathbf{x}$ and the sample mean of a distribution), instead of the Euclidean distance.

$$(\mathbf{x}-\mathbf{m_1})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}-\mathbf{m_1}) = (\mathbf{x}-\mathbf{m_2})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x}-\mathbf{m_2})$$

QDA

```
% The case with varying covariances
N = 1000;

% Class 1
m1 = [3, 3]';   % Mean vector
cov1 = [2 1; 1 2]; % Covariance matrix
%rng default  % For reproducibility
r1 = mvnrnd(m1,cov1,N);

data_C1 = zeros(N, 2);
data_C1 = r1;
label_C1 = ones(N, 1);

% Generate data entries for Class 2
m2 = [9, 9]';
cov2 = [5,3; 3,5];
%rng default  % For reproducibility
r2 = mvnrnd(m2,cov2,N);

data_C2 = zeros(N, 2);
data_C2 = r2;
label_C2 = 2*ones(N, 1);

% Combine data of two classes
data = vertcat (data_C1, data_C2);
label = vertcat (label_C1, label_C2);
```

```
>> K = Model_QDA.Coeffs(1,2).Const

K =

   8.2375
```

Model_QDA.Coeffs(1, 2)

| Field | Value |
| --- | --- |
| DiscrimType | 'quadratic' |
| Const | 8.2375 |
| Linear | [-0.2651;-0.0500] |
| Quadratic | [-0.1728,0.0778;0.0778,-0.1830] |
| Class1 | 1 |
| Class2 | 2 |

$$K + \begin{bmatrix} x_1 & x_2 \end{bmatrix} L + \begin{bmatrix} x_1 & x_2 \end{bmatrix} Q \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0.$$

http://www.ece.uah.edu/~dwpan/course/ee610/code/Discriminant%20Analysis/lda_demo.py

```
import numpy as np
infile = "lda_data.csv"
dataset = np.loadtxt(infile, delimiter=',')
X = dataset[:, 0:2]

y = dataset[:,2]  # labels

from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA

clf = LDA()
clf.fit(X, y)
clf.intercept_
clf.coef_
clf.score(X,y)


clf.intercept_
Out[6]: array([-24.34960648])

clf.coef_
Out[7]: array([[2.12826041, 1.92595686]])

clf.score(X,y)
Out[8]: 0.994
```

Regression

- Linear models for regression
- Polynomial curve fitting as an illustrative example
- Solution to a least-square problem
- Math review
  - Vector Calculus
  - Linear Algebra
    - Vector Space, QR Decomposition, SVD, Condition Numbers, etc.
- Numerical Stability
- Implementations