Bayes

https://github.com/Huafeng-XU

1.Significance of Bayes

Bayes is extremely useful in many aspects. In this note, I will like to show you how to use Bayes to classify the data.

2.Formula description

In this part, I will show you the Formula description of Bayes from my teacher showed for me. Actually, it is uneasy to understand its principle.

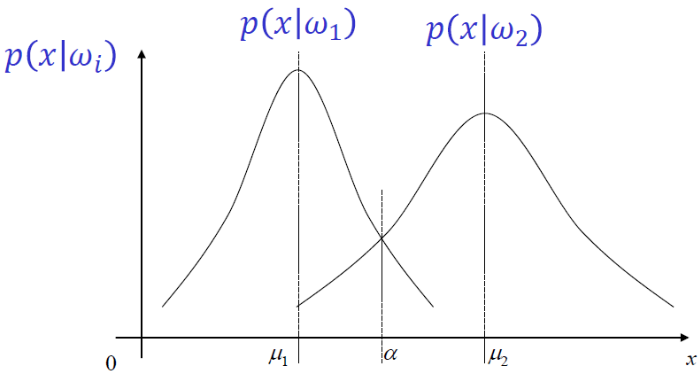
Feeling it difficult? Me too. If you just want to know how Bayes work on matlab, you can **skip this part** and read part 3.

Bayes classifier for Gaussian pattern classes:

- Consider a 1-D problem (n = 1) involving two pattern classes (M = 2) governed by Gaussian densities, with means μ_1 and μ_2 and standard deviations σ_1 and σ_2, respectively

$$g_i(x) = p(x|\omega_i)P(\omega_i)$$

$$= \frac{1}{\sqrt{2\pi}\sigma_i}\exp\left\{-\frac{1}{2}\left(\frac{x-\mu_i}{\sigma_i}\right)^2\right\}P(\omega_i), j = 1, 2$$



- For a multidimensional Gaussian distribution

$$p(\boldsymbol{x}|\omega_i) = (2\pi)^{-\frac{n}{2}}|\boldsymbol{\Sigma}_i|^{-\frac{1}{2}}\exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right\}$$

- The class dependence is obtained parametrically via *class specific mean vectors*, $\boldsymbol{\mu}_i$, and *covariance matrices*, $\boldsymbol{\Sigma}_i$, which are defined as

$$\boldsymbol{\mu}_i = E_i(\boldsymbol{x})$$

and $$\boldsymbol{\Sigma}_i = E_i\{(\boldsymbol{x}-\boldsymbol{\mu}_i)\,(\boldsymbol{x}-\boldsymbol{\mu}_i)^T\}$$

- An estimate of the mean vector and covariance matrix

$$\boldsymbol{\mu}_i = \frac{1}{n_i}\sum_{\boldsymbol{x}\in\omega_i}\boldsymbol{x}$$

and $$\boldsymbol{\Sigma}_i = \frac{1}{n_i}\sum_{\boldsymbol{x}\in\omega_i}\boldsymbol{x}\boldsymbol{x}^T - \boldsymbol{\mu}_i\boldsymbol{\mu}_i^T$$

Discriminant Functions:

- the discriminant function for the i^{th} class

$$d_i(\boldsymbol{x}) = p(\boldsymbol{x}|\omega_i)P(\omega_i)$$

- Any monotonically increasing function of $d_i(\boldsymbol{x})$ is also a valid discriminant function

$$\begin{aligned}
g_i(\boldsymbol{x}) &= \log\{p(\boldsymbol{x}|\omega_i)P(\omega_i)\} \\
&= \log p(\boldsymbol{x}|\omega_i) + \log P(\omega_i) \\
&= -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i) - \frac{n}{2}\log 2\pi \\
&\quad -\frac{1}{2}\log|\boldsymbol{\Sigma}_i| + \log P(\omega_i)
\end{aligned}$$

1. *Equal* $P(\omega_i)$ *and covariance matrices, and* $\boldsymbol{\Sigma} = \mathbf{I}$

$$\begin{aligned}
g_i &= -(\boldsymbol{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i) \\
&= \|\boldsymbol{x}-\boldsymbol{\mu}_i\|^2 \\
&= \boldsymbol{x}^T\boldsymbol{x} + 2\boldsymbol{\mu}_i^T\boldsymbol{x} - \boldsymbol{\mu}_i^T\boldsymbol{\mu}_i
\end{aligned}$$

- An Euclidean distance norm results
- Compared to a *linear discriminant function* or *correlation detector* of the form:

$$g_i(\boldsymbol{x}) = \boldsymbol{w}_i^T\boldsymbol{x} + w_{i0}$$

where $w_{i0} = -\frac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\mu}_i$, and $\boldsymbol{w}_i = \boldsymbol{\mu}_i$

2. *Uncorrelated components, unequal variances*

i.e. $\mathbf{\Sigma}_i = \mathbf{\Sigma}$ is diagonal with unequal σ_{ii}^2

$$\mathbf{\Sigma}_i = \mathbf{\Sigma} = \begin{bmatrix} \sigma_{00}^2 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_{11}^2 & 0 & \cdots & 0 \\ & \vdots & & & \vdots \\ 0 & 0 & \cdots & 0 & \sigma_{n-1,n-1}^2 \end{bmatrix}$$

$$\therefore \quad \mathbf{\Sigma}_i^{-1} = \begin{bmatrix} \frac{1}{\sigma_{00}^2} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma_{11}^2} & 0 & \cdots & 0 \\ & \vdots & & & \vdots \\ 0 & 0 & \cdots & 0 & \frac{1}{\sigma_{n-1,n-1}^2} \end{bmatrix}$$

$\Rightarrow$ a weighted distance classifier

3. *Equal covariance matrices for all of the classes,* $\mathbf{\Sigma}_i = \mathbf{\Sigma}$, *equal* $P(\omega_i)$

$$\begin{aligned} h_i &= (\boldsymbol{x} - \boldsymbol{\mu}_i)^T \mathbf{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i) \\ &= \left(\boldsymbol{x}^T \mathbf{\Sigma}^{-1} - \boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1}\right)(\boldsymbol{x} - \boldsymbol{\mu}_i) \\ &= \boldsymbol{x}^T \mathbf{\Sigma}^{-1} \boldsymbol{x} - \boldsymbol{x}^T \mathbf{\Sigma}^{-1} \boldsymbol{\mu}_i - \boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{x} + \boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{\mu}_i \\ &= \boldsymbol{x}^T \mathbf{\Sigma}^{-1} \boldsymbol{x} - 2\boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{x} + \boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{\mu}_i \\ &= \boldsymbol{x}^T \mathbf{\Sigma}^{-1} \boldsymbol{x} - 2\left(\boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{x} - \tfrac{1}{2}\boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{\mu}_i\right) \end{aligned}$$

$\boldsymbol{x}^T \mathbf{\Sigma}^{-1} \boldsymbol{x}$ is common to all classes, therefore

$$\begin{aligned} g_i &= \boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{\mu}_i \\ &= (\mathbf{\Sigma}^{-1} \boldsymbol{\mu}_i)^T \boldsymbol{x} - \tfrac{1}{2}\boldsymbol{\mu}_i^T \mathbf{\Sigma}^{-1} \boldsymbol{\mu}_i \end{aligned}$$

4. $\mathbf{\Sigma}_i$ = *arbitrary*

$$\begin{aligned} g_i &= -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \mathbf{\Sigma}_i^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i) - \frac{1}{2}\log|\mathbf{\Sigma}_i| + \log P(\omega_i) \\ &= -\tfrac{1}{2}\boldsymbol{x}^T \mathbf{\Sigma}_i^{-1} \boldsymbol{x} + \boldsymbol{\mu}_i^T \mathbf{\Sigma}_i^{-1} \boldsymbol{x} - \tfrac{1}{2}\boldsymbol{\mu}_i^T \mathbf{\Sigma}_i^{-1} \boldsymbol{\mu}_i \\ &\quad -\tfrac{1}{2}\log|\mathbf{\Sigma}_i| + \log P(\omega_i) \end{aligned}$$

$\Rightarrow$ hyper-quadratic, i.e. can assume any of the general forms:

hyperplanes, pairs of hyperplanes, hyperspheres, hyperellipsoids, ...

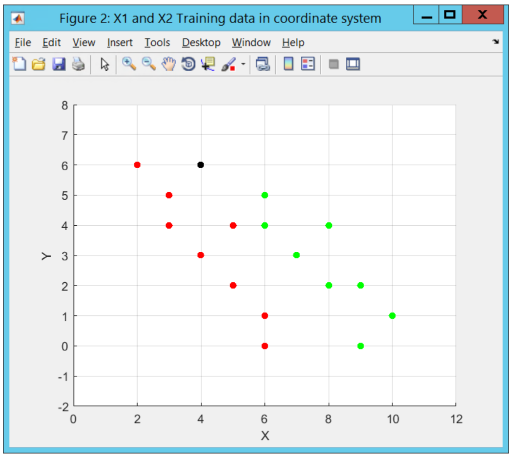
- ## 3.Sample and code on matlab

Consider the following two sets of training data, where the samples are drawn randomly:
$X_1 = \{(2,6),(3,4),(3,5),(4,3),(5,2),(5,4),(6,0),(6,1)\}$
$X_2 = \{(6,4),(6,5),(7,3),(8,2),(8,4),(9,0),(9,2),(10,1)\}$
Assume the $q = (4,6)$ is a query input. You need to find out q belongs to which class.

Question: By using Bayes method to classify the query input q belongs to which class.

Method:

- **Step 1**: Plot X1 and X2 Training data in coordinate system using Matlab. X_1 marked at red, X_2 marked at green and q marked at black.



You can easily use the code below at Matlab to achieve this:

```
clear all;
%row 1 represent x in coordinate system.
%row 2 represent y in coordinate system.
X1=[2 3 3 4 5 5 6 6
   6 4 5 3 2 4 0 1];
X1_x = X1(1,:);
X1_y = X1(2,:);

X2=[6 6 7 8 8 9 9 10
   4 5 3 2 4 0 2 1];
X2_x = X2(1,:);
X2_y = X2(2,:);

%Plot X1 and X2 Training data in coordinate system.
figure('Name','X1 and X2 Training data in coordinate system')
scatter(X1_x,X1_y,'filled','red')
hold on
scatter(X2_x,X2_y,'filled','green')
scatter(4,6,'filled','black')
grid on
axis([0 12 -2 8])
xlabel('X');
ylabel('Y');
```

Hoping that you can type the code yourself. But if you think this may waste your time. You can just download all the code from my **GitHub**: https://github.com/Huafeng-XU. It is in the ***Bayes/****Code*. Actually, step 1 is not necessary. However, it is good for us to observe the training data firstly.

- **Step 2**: Assume that the probability density function of each class is Gaussian, and the prior probability of each class is estimated by the corresponding number of samples in the class. Thus:

$$P(w1) = P(w2) = \frac{1}{2}$$

The code of Matlab is below:

```
Pw1 = 1/2;
Pw2 = 1/2;
```

- **Step 3**: Calculate the mean:

$$m = \frac{1}{n}\sum_{i=1}^{n} x_i$$

The code of Matlab is below:

```
%Calculate the mean value of X1,X2
meanX1 = (1/8 * sum(X1'))';
meanX2 = (1/8 * sum(X2'))';
```

- **Step 4**: Calculate the covariance matrices Σi. Here, I prefer use $\boldsymbol{Si}$ to **replace** $\boldsymbol{\Sigma i}$:

$$Si = \sum_{i=1}^{n} (xi - mi) \cdot (xi - mi)^T$$

The code of Matlab is below:

```
SX1 = (X1 - meanX1) * (X1 - meanX1)';
SX2 = (X2 - meanX2) * (X2 - meanX2)';
```

- **Step 5**: Calculate the Σi^{-1}. Here, I prefer use $\boldsymbol{SiP}$ to **replace** $\boldsymbol{\Sigma i^{-1}}$:

$$SiP = Si^{-1}$$

The code of Matlab is below:

```
%Calculate the Matrix inverse of Si
SX1P = inv(SX1);
SX2P = inv(SX2);
```

- **Step 6**: For decision function:

$$gi(x) = -\frac{1}{2}X^T\Sigma_i^{-1}X + m_i^T\Sigma_i^{-1}X - \frac{1}{2}m_i^T\Sigma_i^{-1}m_i - \frac{1}{2}\log|\Sigma_i|$$

Here, our query input is q. Thus:

$$gi(x) = -\frac{1}{2}q^T\Sigma_i^{-1}q + m_i^T\Sigma_i^{-1}q - \frac{1}{2}m_i^T\Sigma_i^{-1}m_i - \frac{1}{2}\log|\Sigma_i|$$

The code of Matlab is below:

```
%For decision function
%Find out the query input q=(4,6) belongs to
which class
q = [4 6]';
gX1 = -0.5 * q' * SX1P * q + meanX1' * SX1P * q
-0.5 * meanX1' * SX1P * meanX1 - 0.5 *
log(abs(det(SX1)));

gX2 = -0.5 * q' * SX2P * q + meanX2' * SX2P * q
-0.5 * meanX2' * SX2P * meanX2 - 0.5 *
log(abs(det(SX2)));

if gX1 > gX2
    class =  1;
else
    class = 2;
end
class
```

- The ***classs = 1*** means that q belongs to class1 and ***classs = 2*** means that q belongs to class2.

  I hope that my work can give you a hand to help you understand LDA better. Thank you for read.

Reference:

[1]. Pattern Recognition:Theory & Application Prof.Kennetth K.M.Lam.
[2]. https://en.wikipedia.org/wiki/Bayes%27_theorem