1. Gaussian Discriminant Analysis

First of all, the Bayes' Formula:

$$p(y=c|x,\theta)=\frac{p(x|y=c;\theta)p(y=c;\theta)}{\sum_{\theta'}p(x|y=c';\theta)p(y=c';\theta)} \tag{0}$$

where c is one of the class.

1.1 class conditional density

We **assume** that the class conditional density is as follows:

$$p(\mathbf{x}|y=c,\theta)=N(\mathbf{x}|\mu_c,\Sigma_c) \tag{1.1}$$

To be more clear, it's a multivariable gaussian distribution and $\mathbf{x}$ is a vector. In the following, I'll use x to replace the bold $\mathbf{x}$, for my own convenience.

So,

$$N(\mathbf{x}|\mu_c,\Sigma_c)=\frac{1}{\sqrt{2\pi\Sigma_c}}e^{-\frac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c)}$$

1.2 Posterior

From, formula 0, we can know that

$$p(y=c|x,\theta)\propto\Pi_cN(x|\mu_c,\Sigma_c) \tag{1.2}$$

where Π_c is the prior probability of class c, $p(y=c)$.

So, the right hand is just the upper part of formula 0. It's because the lower part is on x, which is a fixed parameter.

1.3 Log posterior

Since we're focusing on the difference between those classes, the constant is irrelevant.

$$log\,p(y=c|x;\theta)=log(\Pi_c)-\frac{1}{2}log|2\pi\Sigma_c|-\frac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c)+constant \tag{1.3}$$

1.4 Quadratic decision boundaries

> There's only one higher order of x, and it's $x^T\Sigma_c^{-1}x$.
>
> If each Σ_c is different, it's a quadratic decision boundary.

1.5 Linear decision boundaries

> On the other hand, if Σ_c is independent of c, then there's no more higher order of x. Thus, it's a linear decision boundary.
>
> In fact, there is. But we focus on the difference between those classes. So, it's a linear one.
>
> For some intuition, think about $y=x^3$ and $y=x^3+1$

$$\begin{aligned}log\,p(y=c|x;\theta)&=log(\Pi_c)-\frac{1}{2}log|2\pi\Sigma|-\frac{1}{2}(x-\mu_c)^T\Sigma^{-1}(x-\mu_c)+constant\\&=(log(\Pi_c)-\frac{1}{2}\mu_c^T\Sigma^{-1}\mu_c)\ +\ (\frac{1}{2}x^T\Sigma^{-1}\mu_c+\frac{1}{2}\mu_c^T\Sigma^{-1}x)\ -\ (\frac{1}{2}x^T\Sigma^{-1}x+const)\end{aligned}$$

$$
\begin{aligned}
&= (log(\Pi_c) - \frac{1}{2}\mu_c^T\Sigma^{-1}\mu_c) \ + \ x^T\Sigma^{-1}\mu_c \ - \ (\frac{1}{2}x^T\Sigma^{-1}x + const) \\
&\text{the right most part is independent of class c} \\
&= r_c + x^T\beta_c + k
\end{aligned}
\tag{1.4}
$$

Now, it's more clear that it's a linear function of x.

1.6 LDA VS Logistic regression

The formula 1.4, though we have $p(y=c|x;\theta)$ as its left hand, is actually $p(x|y=c;\theta)p(y=c;\theta)$, the upper part of Bayes Formula, as I mentioned in 1.2 .

We can notice that, the lower part of Bayes Formula is the sum of class c' in the same form as the upper part, class c.

So, we can get that

$$
\begin{gathered}
p(y=c|x;\theta) = \frac{e^{\beta_c^T x + r_c}}{\sum_{c'} e^{\beta_{c'}^T x + r_{c'}}} = \frac{e^{W_c^T[1,x]}}{\sum_{c'} e^{W_{c'}^T[1,x]}} \\
where\ W_c = [r_c, \beta_c]
\end{gathered}
\tag{1.5}
$$

That's exactly what logistic regression will get.

1.7 Binary case

To get more intuition with the relationship of LDA and Logistic regression, let's take a binary case as an example.

$$
\begin{aligned}
p(y=1|x;\theta) &= \frac{e^{\beta_1^T x + r_1}}{e^{\beta_1^T x + r_1} + e^{\beta_0^T x + r_0}} \\
&= \frac{1}{1 + e^{(\beta_0 - \beta_1)^T x + (r_0 - r_1)}} \\
&= \sigma((\beta_1 - \beta_0)^T x + (r_1 - r_0))
\end{aligned}
$$

To get the form like $\sigma(W^T(x - x_0))$, we can set

$$
W = \beta_1 - \beta_0 = \Sigma^{-1}(\mu_1 - \mu_0) \tag{1.6}
$$

What we need to do is find a x_0, which meet $W^T x_0 = -(r_1 - r_0)$,

Then, we can easily get that

$$
x_0 = \frac{1}{2}(\mu_1 + \mu_0) - (\mu_1 - \mu_0)\frac{log(\frac{\pi_1}{\pi_0})}{(\mu_1 - \mu_0)^T\Sigma^{-1}(\mu_1 - \mu_0)} \tag{1.7}
$$

With this W and x_0, we have

$$
p(y=1|x;\theta) = \sigma(W^T(x - x_0)) \tag{1.8}
$$

And it's equal to

$$
\hat{y}(x) = 1 \ \ iff \ \ W^T x > c \text{ ,where c=}W^T x_0
$$

GEO intuition

We knew that $W = \Sigma^{-1}(\mu_1 - \mu_0)$,

if $\Sigma = \sigma^2 I$,

> then $W = \sigma^2(\mu_1 - \mu_0)$, is parallel to a line joining to the two centroids of these two classes.

> And the product of x and W is to project x to this line, and find whether it's close to class 1 or class 0.

1.8 Model fitting

We are here focusing on mathematical solutions, not gradient descent.

The likelihood of the dataset is

$$p(D|theta) = \Pi_{n=1}^{N} M(y_n|\Pi)\ \ \Pi_{c=1}^{C} N(x_n|\mu_c, \Sigma_c)^{I(y_n=c)}$$

The log likelihood of the dataset would be

$$log\, p(D|\theta) = [\sum_{n=1}^{N}\sum_{c=1}^{C} I(y_{n=c}) log \Pi_c] + \sum_{c=1}^{C}[\sum_{n:y_n=c} log N(x_n|\mu_c, \Sigma_c)] \tag{1.9}$$

So, we can optimize Π and (μ_c, Σ_c) seperately.

The result is as follows:

$$\begin{aligned}
&\hat{\Pi}_c = \frac{N_c}{N} \\
&\hat{\mu_c} = \frac{1}{N_c}\sum_{n:y_n=c} x_n \\
&\hat{\Sigma}_c = \frac{1}{N_c}\sum_{n:y_n=c}(x_n - \hat{\mu_c})(x_n - \hat{\mu_c})^T \\
\\
&\text{If } \Sigma_c = \Sigma \text{, then} \\
&\hat{\Sigma} = \frac{1}{N}\sum_{c=1}^{C}\sum_{n:y_n=c}(x_n - \hat{\mu_c})(x_n - \hat{\mu_c})^T
\end{aligned} \tag{1.10}$$

The deduction is in Chapter 2.

2. Deduction of Log likelihood

2.1 Optimization of $\hat{\Pi}_c$

When focusing on Π_c, the LL can be written as

$$LL = \sum_{c=1}^{C} N_c log(\Pi_c)$$

And the constraint is

$$\sum_{c=1}^{C} \Pi_c = 1$$

Then, the Lagrangian is as follows:

$$L := \sum_{c=1}^{C} N_c log(\Pi_c) - \lambda(\sum_{c=1}^{C}\Pi_c - 1) \tag{2.1}$$

Taking derivatives with respect to λ , we get

$$\frac{\partial L}{\partial \lambda} = \sum_{c=1}^{C}\Pi_c - 1 = 0$$

And the derivation with respect to Π_c is:

$$\frac{\partial L}{\partial \Pi_c} = \frac{N_c}{\Pi_c} - \lambda = 0$$
$$\Pi_c = \frac{N_c}{\lambda}$$

To get rid of λ, we can make use of the sum of Π_c:

$$\sum_c \Pi_c = \sum_c \frac{N_c}{\lambda} = \frac{N}{\lambda}$$
$$\text{We knew that } \sum_c \Pi_c = 1, \text{ so } \frac{N}{\lambda} = 1, N = \lambda$$

As a result,

$$\Pi_c = \frac{N_c}{N}$$

2.2 Optimization of μ_c

First of all, the derivation of μ_c will remove all the parameters that's not of class c.

The log likelihood with respect to class c is:

$$\begin{aligned} LL_c &= \sum_{n=1}^{N_c} logN(x_n|\mu_c, \Sigma) \\ &= \frac{N_c}{2} log|\Lambda| - \frac{1}{2}\sum_{n=1}^{N_c}(x_n - \mu_c)^T\Lambda(x_n - \mu_c), \text{ignore the constant and } \Lambda = \lambda^{-1} \end{aligned} \tag{2.2}$$

Let $z_n = x_n - \mu_c$, then

$$\begin{aligned} \frac{\partial LL_c}{\partial \mu_c} &= -\frac{1}{2}\sum_{n=1}^{N_c} \frac{\partial z_n^T \Lambda z_n}{\partial z_n} \frac{\partial z_n}{\partial \mu_c} \\ &= -\frac{1}{2}\sum_{n=1}^{N_c}(\Lambda + \Lambda^T) z_n(-I) \\ &\Lambda \text{ is a diagnal matrix, } \Lambda^T = \Lambda \\ &= \sum_{n=1}^{N_c} \Lambda z_n = 0 \\ \text{So we get,} & \\ &\sum_{n=1}^{N_c} \Lambda z_n = 0 \\ &\sum_{n=1}^{N_c} (x_n - \mu_c) = 0 \\ &\mu_c = \overline{x_n} \end{aligned} \tag{2.3}$$

2.3 Optimization of Σ

Here we need

$$\begin{aligned} &tr(scalar) = scalar \\ &tr(ABC) = tr(BCA) = tr(CAB) \\ &\frac{\partial tr(AB)}{\partial B} = A^T \\ &\frac{\partial log|A|}{\partial A} = (A^{-1})^T \end{aligned}$$

Reference: [derivation for det]()

Then, we're good to go,

$$
\begin{aligned}
\frac{LL_c}{\Lambda} &= \frac{\partial \frac{N_c}{2} log|\Lambda| - \frac{1}{2}\sum_{n=1}^{N_c}(x_n-\mu_c)^T\Lambda(x_n-\mu_c)}{\partial\Lambda} \\
&= \frac{\partial \frac{N_c}{2} log|\Lambda| - \frac{1}{2}\sum_{n=1}^{N_c} tr((x_n-\mu_c)^T\Lambda(x_n-\mu_c))}{\partial\Lambda} \\
&= \frac{\partial \frac{N_c}{2} log|\Lambda| - \frac{1}{2}\sum_{n=1}^{N_c} tr((x_n-\mu_c)(x_n-\mu_c)^T\Lambda)}{\partial\Lambda} \\
&= \frac{N_c}{2}\Lambda^{-1} - \frac{1}{2}\sum_{n=1}^{N_c}(x_n-\mu_c)(x_n-\mu_c)^T = 0
\end{aligned}
$$

So, we can get

$$
\begin{aligned}
\Lambda^{-1} &= \Sigma_c \\
&= \frac{1}{N_c}\sum_{n=1}^{N_c}(x_n-\mu)(x_n-\mu)^T
\end{aligned}
$$

Most part are based on **Murphy**'s book, **Probabilistic Machine Learning**.

And part 2.3 is based on this blog.