第六章 判别分析

6.2 Fisher判别

Fisher 判别的基本思想是投影，即将表面上不易分类的数据通过投影到某个方向上，使得投影类与类之间得以分离的一种判别方法。

仅考虑两总体的情况，设两个p维总体为X_1, X_2，且都有二阶矩存在。Fisher 的判别思想是变换多元观测x到一元观测y，使得由总体X_1, X_2产生的y尽可能的分离开来。

第六章 判别分析

1.Fisher判别基本原理

设在p维的情况下，x的线性组合$y=a^T x$，其中a为p维实向量。设X_1, X_2的均值向量分别为μ_1, μ_2（均为p维），且有公共的协方差矩阵Σ（$\Sigma>0$）。那么线性组合$y=a^T x$的均值为

$$\mu_{y_1} = E(y \mid y=a^T x, x \in X_1) = a^T \mu_1,$$

$$\mu_{y_2} = E(y \mid y=a^T x, x \in X_2) = a^T \mu_2,$$

其方差为

$$\sigma_y^2 = \mathrm{Var}(y) = a^T \Sigma a,$$

第六章 判别分析

1.Fisher判别基本原理

考虑比

$$\frac{(\mu_{y_1}-\mu_{y_2})^2}{\sigma_y^2}=\frac{[a^T(\mu_1-\mu_2)]^2}{a^T\Sigma a}=\frac{(a^T\delta)^2}{a^T\Sigma a},$$

其中$\delta=\mu_1-\mu_2$为两总体均值向量差，根据 Fisher 的思想，我们要选择a使得上式达到最大。

第六章 判别分析

1. Fisher判别基本原理

求a，使得$\max \dfrac{(\mu_{y_1}-\mu_{y_2})^2}{\sigma_y^2}=\dfrac{[a^T(\mu_1-\mu_2)]^2}{a^T\Sigma a}=\boxed{\dfrac{(a^T\delta)^2}{a^T\Sigma a}}$

达到最大，其中$\delta=\mu_1-\mu_2$

定理 6.1　x为p维随机变量，设$y=a^Tx$，当选取$a=c\Sigma^{-1}\delta$，$c\neq 0$为常数时，上式达到最大。

特别当$c=1$时，线性函数　$\boxed{a=\Sigma^{-1}\delta}$

$$y=a^Tx=(\mu_1-\mu_2)^T\Sigma^{-1}x$$

称为 Fisher 线性判别函数。令

$$K=\frac{1}{2}(\mu_{y_1}+\mu_{y_2})=\frac{1}{2}(a^T\mu_1+a^T\mu_2)=\frac{1}{2}(\mu_1-\mu_2)^T\Sigma^{-1}(\mu_1+\mu_2).$$

定理 6.2　利用上面的记号，取$a^T=(\mu_1-\mu_2)^T\Sigma^{-1}$，则有

$$K=\frac{1}{2}(\mu_{y_1}+\mu_{y_2})=\frac{1}{2}(a^T\mu_1+a^T\mu_2)=\frac{1}{2}(\mu_1-\mu_2)^T\Sigma^{-1}(\mu_1+\mu_2).$$

$$\mu_{y_1}=a^T\mu_1$$
$$\mu_{y_2}=a^T\mu_2$$

$$\mu_{y_1}-K>0,\quad \mu_{y_2}-K<0.$$

由定理 6.1 得到如下的 Fisher 判别规则

$$\begin{cases} x\in X_1, \text{当}x\text{使得}(\mu_1-\mu_2)^T\Sigma^{-1}x\geq K, \\ x\in X_2, \text{当}x\text{使得}(\mu_1-\mu_2)^T\Sigma^{-1}x<K. \end{cases}$$

定义判别函数

$$W(x)=(\mu_1-\mu_2)^T\Sigma^{-1}x-K=\left[x-\frac{1}{2}(\mu_1+\mu_2)\right]^T\Sigma^{-1}(\mu_1-\mu_2) \quad (10.34)$$

则判别规则可改写成

$$\begin{cases} x\in X_1, \text{当}x\text{使得}W(x)\geq 0, \\ x\in X_2, \text{当}x\text{使得}W(x)<0. \end{cases}$$

第六章 判别分析

当总体的参数未知时，用样本对μ_1,μ_2及Σ进行估计，注意到这里的 Fisher 判别与距离判别一样不需要知道总体的分布类型，但两总体的均值向量必须有显著的差异才行，否则判别无意义。

2.Fisher判别的python计算

例 6.1 1989 年国际大学生数学建模竞赛 A 题：朦虫分类）朦虫是一种昆虫，分为很多类型，其中有一种名为 Af，是能传播花粉的益虫；另一种名为 Apf，是会传播疾病的害虫。这两种类型的朦虫在形态上十分相似，很难区别。现测得 9 只 Af 和 6 只 Apf 朦虫的触角长度和翅膀长度数据。

Af：(1.24,1.27)，(1.36,1.74)，(1.38,1.64)，(1.38,1.82)，(1.38,1.90)，(1.40,1.70)，(1.48,1.82)，(1.54,1.82)，(1.56,2.08)；

Apf：(1.14,1.78)，(1.18,1.96)，(1.20,1.86)，(1.26,2.00)，(1.28,2.00)，(1.30,1.96)。

若两类朦虫协方差矩阵相等，试判别(1.24,1.80)，(1.28,1.84)与(1.40,2.04)三只朦虫属于哪一类。

```
#程序文件 Pex6_3.py
import numpy as np
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
x0=np.array([[1.24,1.27], [1.36,1.74], [1.38,1.64], [1.38,1.82], [1.38,1.90],
      [1.40,1.70],[1.48,1.82], [1.54,1.82], [1.56,2.08], [1.14,1.78], [1.18,1.96],
      [1.20,1.86], [1.26,2.00], [1.28,2.00], [1.30,1.96]])  #  输入已知样本数据
x=np.array([[1.24,1.80], [1.28,1.84], [1.40,2.04]]) # 输入待判样本点数据
y0=np.hstack([np.ones(9),2*np.ones(6)]) #y0  为已知样本数据的类别
```

```
clf = LDA()
clf.fit(x0, y0)
print("判别结果为： ",clf.predict(x))
print("已知样本的误判率为： ",1-clf.score(x0,y0))
```

程序运行结果如下：

判别结果为： [2. 2. 2.]

已知样本的误判率为： 0.0

例 6.2　从健康人群、硬化症患者和冠心病患者中分别随机选取 10 人、6 人和 4 人，考察了他们各自心电图的 5 个不同指标（记作 x_1, x_2, x_3, x_4, x_5）如表 11.2 所示，试对两个待判样品作出判断。

序号	x_1	x_2	x_3	x_4	x_5	类型
1	8.11	261.01	13.23	5.46	7.36	1
2	9.36	185.39	9.02	5.66	5.99	1
3	9.85	249.58	15.61	6.06	6.11	1
4	2.55	137.13	9.21	6.11	4.35	1
5	6.01	231.34	14.27	5.21	8.79	1
6	9.46	231.38	13.03	4.88	8.53	1
7	4.11	260.25	14.72	5.36	10.02	1
8	8.90	259.51	14.16	4.91	9.79	1
9	7.71	273.84	16.01	5.15	8.79	1
10	7.51	303.59	19.14	5.7	8.53	1
11	6.8	308.9	15.11	5.52	8.49	2
12	8.68	258.69	14.02	4.79	7.16	2
13	5.67	355.54	15.13	4.97	9.43	2
14	8.1	476.69	7.38	5.32	11.32	2
15	3.71	316.12	17.12	6.04	8.17	2
16	5.37	274.57	16.75	4.98	9.67	2
17	5.22	330.34	18.19	4.96	9.61	3
18	4.71	331.47	21.26	4.3	13.72	3
19	4.71	352.5	20.79	5.07	11	3
20	3.36	347.31	17.9	4.65	11.19	3
21	8.06	231.03	14.41	5.72	6.15	待判
22	9.89	409.42	19.47	5.19	10.49	待判

```
#程序文件 Pex6_4.py
import pandas as pd
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
a=pd.read_excel("Pdata11_2.xlsx",header=None)
b=a.values
x0=b[:-2,1:-1].astype(float) # 提取已知样本点的观测值
y0=b[:-2,-1].astype(int)
x=b[-2:,1:-1] # 提取待判样本点的观察值
clf = LDA()
clf.fit(x0, y0)
print("判别结果为： ",clf.predict(x))
print("已知样本的误判率为： ",1-clf.score(x0,y0))
```

程序运行结果如下：

判别结果为： [1 2]

已知样本的误判率为： 0.0

从上面例子可以看出，Fisher 线性判别法的效果比距离判别法的效果要好。