判别分析

一、距离判别法

>> Mahalanobis距离（简称马氏距离）

设$\boldsymbol{x}$、$\boldsymbol{y}$是从均值为$\boldsymbol{\mu}$、协方差为$\boldsymbol{\Sigma}$的总体$\boldsymbol{A}$中抽取出来的样本，则

- 总体$\boldsymbol{A}$内两点$\boldsymbol{x}$与$\boldsymbol{y}$的马氏距离定义为

$$d(\boldsymbol{x},\boldsymbol{y})=\sqrt{(\boldsymbol{x}-\boldsymbol{y})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{y})}$$

- 样本$\boldsymbol{x}$与总体$\boldsymbol{A}$的马氏距离为

$$d(\boldsymbol{x},\boldsymbol{A})=\sqrt{(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$$

```python
import numpy as np
import pandas as pd
def calMahalanobis(x: np.ndarray, A: pd.DataFrame):
    """计算样本x到总体A之间的Mahalanobis距离（马氏距离）"""
    mu = A.mean().values
    cov = A.cov().values
    x_ = (x-mu).reshape(-1, 1)
    res = x_.T @ np.linalg.inv(cov) @ x_
    return res[0][0]**0.5
```

>> 判别方法

计算样本x到各总体间的马氏距离，最小距离的那一类即为该样本所属的类

二、Fisher判别法

> Fisher判别又称为线性判别分析（Linear Discriminant Analysis，LDA），
> 其基本思想是投影，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。

>> 算法原理

对于两个总体G_1、G_2，其均值向量分别为μ_1、μ_2，公共的协方差矩阵为Σ。

对于一个样本x，其判别函数为：

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

判别准则为：

$$\begin{cases}x\in G_1, & W(x)\geq 0\\ x\in G_2, & W(x)<0\end{cases}$$

>> Python实现

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
md = LDA().fit(x0, y0)
val = md.predict(x)
```

三、Bayes判别

> Bayes判别和Bayes估计的思想方法是一样的，即假定对研究的对象已经有一定的认识，这种认识常用先验概率来描述。当取得一个样本后，就可以用样本来修正已有的先验概率分布，得出后验概率分布，再通过后验概率分布进行各种统计推断。

>> 算法原理

对于两个总体G_1、G_2，其均值向量分别为μ_1、μ_2，公共的协方差矩阵为Σ。

设两个总体的先验概率分别为p_1、p_2；来自G_2误判为G_1引起的损失为$L(1|2)$，来自G_1误判为G_2引起的损失为$L(2|1)$

对于一个样本x，其判别函数为：

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

判别准则为：

$$\begin{cases}x\in G_1, & W(x)\geq\beta\\ x\in G_2, & W(x)<\beta\end{cases}\quad 其中，\beta=\ln\frac{L(1|2)\cdot p_2}{L(2|1)\cdot p_1}$$

四、判别分析的评价方法

1. 回代误判率

$$\hat{P}=\frac{N_1+N_2}{m+n}$$

其中，N_1、N_2分别为两个总体回代误判的个数，m、n为两个总体的样本数。

2. 交叉误判率

>> 算法原理

交叉误判率估计是每次删除一个样品，利用其余的$m+n-1$个训练样品建立判别准则，再用所建立的准则对删除的样品进行判别。对训练样品中每个样品都做如上分析，以其误判的比例作为误判率。

>> Python实现

```
from sklearn.model_selection import cross_val_score
res = cross_val_score(model, x0, y0,cv=k)  # cv=k表示把已知样本点分为k组
```