

# PRML 第一次作业

## 参数法与非参数法分类器

学生： 槐 泽 鹏

导师：

学号:XXXXXXXXXXXXXXXXXX

联系方式: XXXXXXXXXXXXX

2020 年 11 月 23 日

# 目 录

第 1 节 参数法	1
1.1 贝叶斯决策及最小化风险决策 . . . . .	1
1.2 判别函数 . . . . .	2
1.3 高斯分布下的判别函数 . . . . .	2
1.4 LDF+QDF+RDA+MQDF . . . . .	3
第 2 节 非参数法	4
2.1 非参数法简介 . . . . .	4
2.2 Parzen window . . . . .	5
第 3 节 实验	7
3.1 数据集 . . . . .	7
3.2 结果 . . . . .	7
3.3 分析 . . . . .	8
第 4 节 小结	8
第 5 节 参考文献	8
第 6 节 代码	9

## 第 1 节 参数法

### 1.1 贝叶斯决策及最小化风险决策

贝叶斯决策论 (Bayesian decision theory) 假设模式分类的决策可由概率形式描述, 并假设问题的概率结构已知。

有以下规定:

1.  $c$  为样本类别数量
2.  $\omega_1, \omega_2, \dots, \omega_c$  为样本类别
3.  $P(\omega_i)$  为第  $i$  类样本的先验概率, 且  $\sum_{i=1}^c P(\omega_i) = 1$
4. 样本  $\mathbf{x} \in \mathbb{R}^d$
5. 类别  $\omega_i$  对样本条件概率密度为  $p(\mathbf{x}|\omega_i)$ , 即似然 (likelihood)
6.  $P(\omega_i|\mathbf{x})$  为已知样本  $\mathbf{x}$ , 其属于类别  $\omega_i$  的后验概率 (posterior)

则后验概率  $P(\omega_i|\mathbf{x})$  可以用贝叶斯公式来描述:

$$P(\omega_i|\mathbf{x}) = \frac{P(\mathbf{x}|\omega_i)P(\omega_i)}{P(\mathbf{x})} = \frac{P(\mathbf{x}|\omega_i)P(\omega_i)}{\sum_{j=1}^c P(\mathbf{x}|\omega_j)P(\omega_j)} \quad (1)$$

式 1 中分母被称为证据因子 (evidence)。后验概率当然也满足和为 1,  $\sum_{j=1}^c P(\omega_j|\mathbf{x}) = 1$ 。

所以, 当类条件概率密度、先验概率已知时, 可以用最大后验概率决策 (Maximum a posteriori decision), 将样本的类别判为后验概率最大的那一类。决策规则为:

$$\arg \max_i P(\omega_i|\mathbf{x}) \quad (2)$$

也就是说, 如果样本  $\mathbf{x}$  属于类别  $\omega_i$  的后验概率  $P(\omega_i|\mathbf{x})$  大于其它任一类别的后验概率  $P(\omega_j|\mathbf{x}) (j \in \{1, \dots, c\} \setminus \{i\})$ , 则将该样本分类为类别  $\omega_i$ 。

式 1 还有另一种解释, 就是最小化风险决策, 如下所示。从平均错误率 (平均误差概率)  $P(error)$  最小的角度出发, 讨论模型如何来对样本的类别进行决策。平均错误率的表达式为:

$$P(error) = \int P(error)dx = \int P(error|\mathbf{x})P(\mathbf{x})d\mathbf{x} \quad (3)$$

可以看出, 如果对于每个样本  $x$ , 保证  $P(error|\mathbf{x})$  尽可能小, 那么平均错误率就可以最小。 $P(error|\mathbf{x})$  的表达式为:

$$P(error|\mathbf{x}) = 1 - P(\omega_i|\mathbf{x}), \text{ if we decide } \omega_i \quad (4)$$

从这个表达式可以知道, 最小错误率决策 (Minimum error rate decision) 等价于最大后验概率决策。

## 1.2 判别函数

判别函数  $g_i(x)$  是对分类器的一种描述。分类规则可以描述为：

$$\arg \max_i g_i(\mathbf{x}) \quad (5)$$

也就是说，选择如果样本  $\mathbf{x}$  使得类别  $\omega_i$  的判别函数  $g_i(x)$  的值最大，大于其他任一类别的判别函数值  $g_j(\mathbf{x}) (j \in \{1, \dots, c\} \setminus \{i\})$ ，则将它判为  $\omega_i$  类，这个类别的决策区域记为  $R_i$ 。

特别地，考虑二类分类问题，可得到两个类别的决策面（判别边界）方程为： $g_1(\mathbf{x}) = g_2(\mathbf{x})$ 。

对于贝叶斯分类器来说，判别函数的形式可以有多种选择：最小风险决策对应于  $g_i(x) = -R(\omega_i|\mathbf{x})$ ；最小错误率决策对应于  $g_i(x) = -R(\omega_i|\mathbf{x})$ ，可简化为  $P(\mathbf{x}|\omega_i)P(\omega_i)$ ，或者取对数后成为：

$$g_i(\mathbf{x}) = \ln P(\mathbf{x}|\omega_i) + \ln P(\omega_i) \quad (6)$$

## 1.3 高斯分布下的判别函数

机器学习中生成模型 (Generative model)，本质在于学习数据内部的规律分布，即在得到所有不同类样本的分布之后再进行模式识别/分类，其解决问题的逻辑是  $\mathbf{x} \rightarrow p(\mathbf{x}|\omega_i) \rightarrow g_i(\mathbf{x})$ 。这其中的关键就是如何得到  $P(\mathbf{x}|\omega_i)$ 。

两种方法：参数法和非参数法。本小节介绍参数法，下节介绍非参数法。

参数方法将类条件概率密度  $P(\mathbf{x}|\omega_i)$  指定为一个显式表示的函数，如多元高斯密度；然后将估计条件概率密度转化为了估计该显式函数的参数（比如多元高斯分布的均值和协方差矩阵）。

对于生成式模型来说，由于是不同的类别  $\omega_i$  的样本是分开学习的，每个类别都有自己的判别函数  $g_i(\mathbf{x})$ ，不关心彼此之间的区别（而判别模型则是所有类别的样本放在一起学习，直接从样本标签学习判别函数，直接区分不同类别样本），所以下面的讨论中将省略类别信息  $\omega_i$  中的下标  $i$ ，因为每个类别都是一样的过程。

假设样本  $\mathbf{x} \in \mathbb{R}^d$  的条件概率密度服从多元高斯分布  $p(x) \sim N(\mu, \Sigma)$ ，即：

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1}(\mathbf{x} - \mu)\right) \quad (7)$$

其中， $\mu$  和  $\Sigma$  分别为均值向量和协方差矩阵。

我们对式 6 乘以 2 以消除小数系数，并将式 7 代入，得：

$$g_i(\mathbf{x}) = 2 \log P(\omega_i) - (x - \mu)^T \Sigma^{-1} (x - \mu) - d \log 2\pi - \log |\Sigma_i| \quad (8)$$

后面我们针对式 8 进行讨论。

#### 1.4 LDF+QDF+RDA+MQDF

对式 8 进行不同的近似和假设可以得到不同的高斯分类器。

(1) LDF<sup>[4]</sup>

当假设不同类别样本等协方差时，即假设条件为：

$$\Sigma_i = \Sigma \quad (9)$$

此时判别函数为：

$$g_i(\mathbf{x}) = 2 \log P(\omega_i) - (x - \mu)^T \Sigma^{-1} (x - \mu) \quad (10.1)$$

$$= f(\mathbf{x}) + \mathbf{W}^T \mathbf{x} + \mathbf{b}_i \quad (10.2)$$

式 10 即为 LDF (Linear discriminant function) 高斯分类器，其中 10.2 表明这是一个线性分类器。

(2) QDF<sup>[3]</sup>

当假设不同类别样本等先验概率时，即假设条件为：

$$P(\omega_i) = P_0 \quad (11)$$

此时判别函数为：

$$g_i(\mathbf{x}) = -(x - \mu)^T \Sigma^{-1} (x - \mu) - \log |\Sigma_i| \quad (12)$$

式 12 即为 QDF (Quadratic discriminant function) 高斯分类器<sup>[5]</sup>。

(3) RDA

QDF 在训练样本少时存在协方差矩阵奇异的问题，为解决此问题，RDA (Regularized discriminant analysis)<sup>[1,2]</sup> 通过平滑协方差克服奇异，同时提高了泛华性能。对协方差矩阵的训练公式为：

$$\hat{\Sigma}_i = (1 - \gamma)[(1 - \beta)\Sigma_i + \beta\Sigma_0] + \gamma\sigma_i^2 I \quad (13)$$

$$\Sigma_0 = \sum_{i=1}^c P(\omega_i) \Sigma_i$$

其中  $\beta$   $\gamma$  均为超参。

#### (4) MQDF

MQDF(Modified quadratic discriminant function)<sup>[5]</sup> 对原始样本进行 K-L 变换, 从而使得协方差矩阵可以被正交分解, 即  $\Sigma_i = \Phi_i \Lambda \Phi_i^T$ , 代入 12, 并对特征值进行降序和截断, 只保留较大的前几阶特征值, 其余特征值使用  $\delta_i$  代替, 则判别函数化简为:

$$g_i(\mathbf{x}) = - \sum_{j=1}^k \frac{1}{\lambda_{ij}} [(x - \mu_i)^T \Phi_{ij}]^2 - \frac{1}{\delta_i} \varepsilon_i(x) - \sum_{j=1}^k \log \lambda_{ij} - (d - k) \log \delta_i \quad (14)$$

其中  $\varepsilon_i(x)$  代表投影距离, 表达式如下:

$$\varepsilon_i(x) = \|x - \mu\|^2 - \sum_{j=1}^k [(x - \mu_i)^T \Phi_{ij}]^2 \quad (15)$$

## 第 2 节 非参数法

### 2.1 非参数法简介

对于生成模型来说, 关键在于样本对于指定类别的条件概率密度  $p(\mathbf{x}|\omega_i)$ 。上一节介绍的参数方法, 假定样本是一个按照类别的固定概率分布形式, 然后估计这个显式分布函数中的未知参数。但这里存在两个问题: 首先, 假定的形式可能是不准确的, 实际数据并不符合这个假设; 其次, 经典的密度函数的参数形式都是单模的 (单个局部极大值), 实际上常常会有多模的情况。

非参数方法的做法是, 不假定密度的形式, 直接从样本来估计类条件概率密度  $p(\mathbf{x}|\omega_i)$ 。考虑一个点  $\mathbf{x}_i$  落在区域  $\mathcal{R}$  中的概率:

$$P = \int_{\mathcal{R}} p(\mathbf{x}) d\mathbf{x} \quad (16)$$

其中  $p(\mathbf{x})$  是概率密度函数。设有  $n$  个服从  $p(\mathbf{x})$  的样本  $\mathbf{x}_1, \dots, \mathbf{x}_n$ , 并且有  $k$  个样本落在区域  $\mathcal{R}$ 。根据二项分布的均值我们可以知道,  $k$  的期望值是  $nP$ 。

现在假设这个区域足够小 (体积  $V$ 、样本数  $k$ ), 使得该区域内部是等概率密度的, 那么可以有

$$P = \int_{\mathcal{R}} p(\mathbf{x}) d\mathbf{x} \approx p(\mathbf{x})V \quad (17)$$

从式 17, 就可以得到  $p(\mathbf{x})$  的估计为

$$p(\mathbf{x}) \approx \frac{k/n}{V} \quad (18)$$

式 18按照物理学理解, 即 “密度等于概率除以体积”。

为了估计点  $\mathbf{x}$  处的概率密度  $p(\mathbf{x})$ ，采取的做法是构造一系列包含点  $\mathbf{x}$  的区域  $\mathcal{R}_1, \dots, \mathcal{R}_n$ ，进而产生一个估计序列，该序列收敛到的值就是真实的概率密度。那么对  $p(\mathbf{x})$  的第  $n$  次估计可表示为

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n} \quad (19)$$

上式在  $n \rightarrow \infty$  时收敛到  $p(\mathbf{x})$  的条件有三个： $V_n \rightarrow 0$ 、 $k_n \rightarrow \infty$ 、 $k_n/n \rightarrow 0$ 。下面有两种处理方式：

1. 固定体积  $V$ ，改变  $k$ ，即 Parzen 窗估计。换言之，就是当  $n$  给定时，要求  $V_n$  唯一确定，再求此时的  $k_n$ 。
2. 固定样本数  $k$ ， $V$  改变，即  $k$  近邻估计。也就是说，当  $n$  给定时，要求  $k_n$  唯一确定，再求此时的  $V_n$ 。

以上两种非参数法分类如图 2.1 所示。

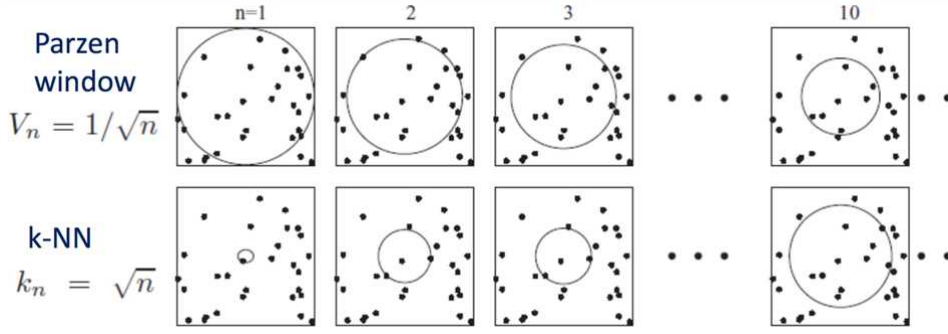


图 2.1 两种非参数法示意图

## 2.2 Parzen window

Parzen window 估计通过改变区域内的样本数来获得估计序列。先举一个简单的例子：设  $\mathcal{R}_n$  是一个窗宽为  $h_n$  的  $d$  维超立方体，则局部区域的体积为  $V_n = h_n^d$ 。定义窗函数  $\phi(\mathbf{u})$ ：

$$\phi(\mathbf{u}) = \begin{cases} 1, & |u_j| \leq \frac{1}{2}; \\ 0, & \text{otherwise.} \end{cases} \quad (20)$$

显然， $\phi(\mathbf{u})$  表示的是一个中心在原点的单位超立方体。在窗函数定义好之后，可以求得以  $\mathbf{x}_n$  为中心、体积为  $V_n = h_n^d$  的局部区域内的样本数：

$$k_n = \sum_{i=1}^n \phi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right) \quad (21)$$

这个式子中， $\mathbf{x}$  是特征空间内任一点， $\mathbf{x}_i$  是  $i$  个样本点。当一个样本  $\mathbf{x}_i$  落在局部区域内时，有  $\phi(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}) = 1$ ，所以累加和可求得局部区域内的样本数。

将式 21 代入式 19，则可估计概率密度函数：

$$p_n(\mathbf{x}) = \frac{1}{nV_n} \sum_{i=1}^n \phi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right) \quad (22)$$

实际上，上式给出了一个一般化的估计方式，它规定了窗函数的作用在于：明确样本点  $\mathbf{x}_i$  对于点  $\mathbf{x}$  的密度函数估计所起到的贡献，贡献大小由两个点的距离来决定。因此，窗函数不一定是上面的形式，只要使得估计出来的密度函数  $p_n(\mathbf{x})$  满足概率密度函数应有的性质（例如，积分为 1）即可。因此，要求窗函数满足：

$$\phi(\mathbf{x}) \geq 0, \quad \int P(\mathbf{u})d\mathbf{u} = 1, \quad V_n = h_n \quad (23)$$

Parzen 窗估计中，如果窗函数的形式是规定好的，那么密度估计的效果好坏将取决于窗宽。例如有两个模型，分别规定  $h_n = \frac{1}{\sqrt{n}}$ 、 $h_n = \frac{2}{\sqrt{n}}$ ，这就说明第一个模型的窗宽小于第二个模型（ $n$  相同时，第二个模型的窗宽总是大于第一个模型的）。

下面就讨论窗宽是如何影响估计效果的。

设  $h_n = \frac{h_1}{\sqrt{n}}$ ，那么通过改变参数  $h_1$  就可获得窗宽不同的模型（ $n$  相同时这些模型有着不同的窗宽）。当然了，窗宽序列的获取当然不是只有这一种途径，一般原则是当  $n$  越大时  $h_n$  越小。使用高斯窗函数得到的结果是：在样本数有限的情况下，

- 1 太大的窗宽会是高方差的，欠拟合；窗宽较大时，平滑程度较大。
- 2 太小的窗宽会是低方差的，过拟合。

继续用高斯窗函数的情况下讨论样本数无穷时去估计概率密度（有单模的高斯密度，也有多模的混合密度），得到的结果是：当样本数无穷时，不管窗宽多大，最后总能准确地估计出真实概率密度。

图 2.2 展示了真实密度是多模的情况下，取不同窗宽（每一列代表一个模型，从左到右窗宽依次减小）时，样本数从有限到无限（从上到下样本数依次增大）的估计效果，可以对照上面两个结论来看这张图。能够估计多模的密度，这正是非参数方法可以做到而参数方法所做不到的。



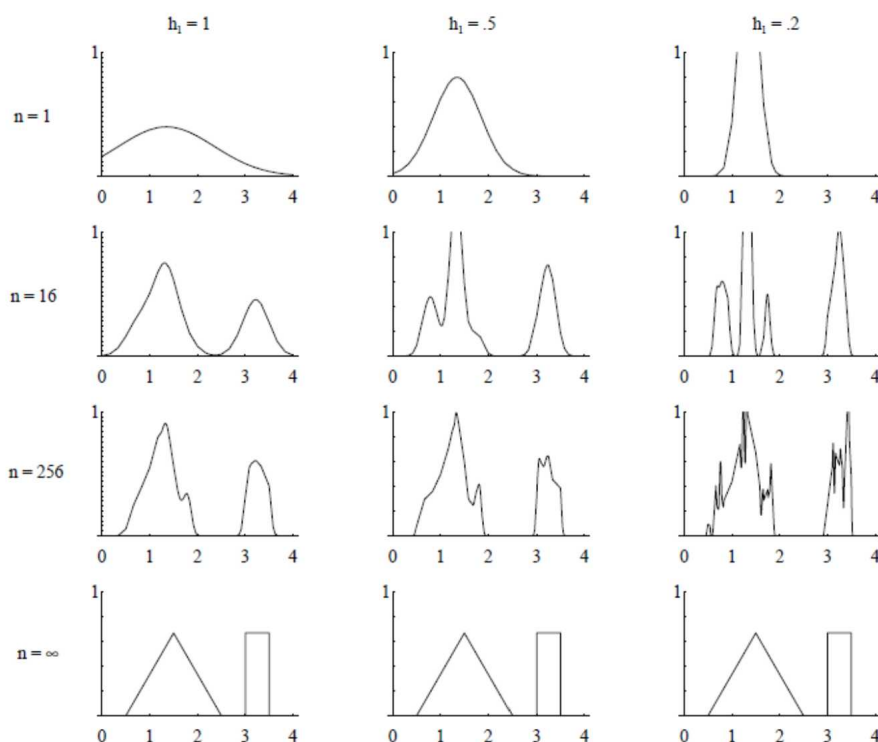


图 2.2 不同窗宽和样本数量的估计效果示意图

## 第 3 节 实验

本节我们在 6 个数据集<sup>1</sup>，测试 4 种参数法分类器和 1 种非参数法分类器的效果。

### 3.1 数据集

一共采用 6 个数据集，他们的静态参数如表 1 所示。

### 3.2 结果

对四种参数法和一种非参数法的分类结果如 2 所示。

<sup>1</sup><http://archive.ics.uci.edu/ml/>

<sup>2</sup><http://archive.ics.uci.edu/ml/machine-learning-databases/00571/>

<sup>3</sup><http://archive.ics.uci.edu/ml/machine-learning-databases/abalone/>

<sup>4</sup><http://archive.ics.uci.edu/ml/datasets/Iris>

<sup>5</sup><http://archive.ics.uci.edu/ml/machine-learning-databases/forest-fires/>

<sup>6</sup><http://archive.ics.uci.edu/ml/machine-learning-databases/wine/>

<sup>7</sup><http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/>

表 1: 数据集参数

数据集 名称	参数			数据本地位置 及 网上下载链接
	所有/代码使用 属性数量	分类名称/数量	样本数量	
HCV	14/10	病症类别/4	615	./data/hcvdat0.csv <sup>2</sup>
Abalone	7/7	性别/3	4177	./data/abalone.data <sup>3</sup>
Iris	4/4	植物类别/3	150	./data/iris.csv <sup>4</sup>
ForestFires	13/8	火灾月份/12	517	./data/forestfires.csv <sup>5</sup>
Wine	13/13	酒类别/3	178	./data/wine.data <sup>6</sup>
BreastCancer	32/9	症状类别/2	569	./data/wdbc.data <sup>7</sup>

表 2: 参数法及非参数法分类结果

数据集	LDF	QDF	MQDF	RDA		Parzen window	
				结果	$\gamma/\beta$	结果	最佳窗宽 $h_{best}$
HCV	0.0081	0.0325	<b>0.8617</b>	<b>0.8617</b>	0.05/0	0.0162	0.1
Abalone	0.5526	0.4940	0.4605	0.5514	0/0.75	<b>0.5586</b>	0.1
Iris	0.9	1	0.9333	1	0.25/0	0.9	0.2
ForestFires	0.1826	0.1634	0.0865	0.1634	0/0	<b>0.4038</b>	0.1
Wine	<b>0.9722</b>	<b>0.9722</b>	0.1666	<b>0.9722</b>	0/0.5	<b>0.9722</b>	0.1
BreastCancer	<b>0.9473</b>	0.9210	0.7017	0.9385	0/0.35	0.9298	0.2
备注: 1 超参数 $\beta$ 和 $\gamma$ 交叉验证的步长设为 0.05 2 这里我们设定 $\beta$ 和 $\gamma$ 可以取到 0 3 非参数法中最佳窗宽 $h_{best}$ 交叉验证的步长设为 0.1, 采用高斯窗形式。							

### 3.3 分析

## 第 4 节 小结

## 第 5 节 参考文献

- [1] Ke Lin Du and M. N. S. Swamy. *Regularized Discriminant Analysis*. 2019.
- [2] Jerome H. Friedman. Regularized discriminant analysis. *Journal of the Amer-*

ican Statal Association, 84(405):165–175, 1989.

- [3] Cheng Lin Liu, H. Sako, and H. Fujisawa. *Discriminative learning quadratic discriminant function for handwriting recognition*. IEEE Press, 2004.
- [4] M.N Murty and Rashmi Raghava. *Linear Discriminant Function*. Springer International Publishing, 2016.
- [5] M. Sakai, M. Yoneda, and H. Hase. A new robust quadratic discriminant function. In *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, 1998.

## 第 6 节 代码

本大作业是基于 GitHub *luyi-pamela* 的代码进行的修改，详见 <https://github.com/pamela0530/bayes>。

尽管本次作业基于以上代码编写，但是原来代码有一些原理上的错误，同时本人针对一些可改善的地方进行了修改，共有 3 处修改和错误，如下所示。

最后本人代码也已经上传至本人 GitHub 仓库，详见 [https://github.com/huaizepeng2020/PRML\\_HW1.git](https://github.com/huaizepeng2020/PRML_HW1.git)。

1. 计算判别函数时少加了  $-\log |\Sigma_i|$ ，源代码和改后代码如图 6.1 所示。

```
def discriminant_function(self, x, covMatrix, mean_i, p_i):
    #covMatrix = self.data.cov().as_matrix(columns=None)
    # print covMatrix
    cov_inverse = np.linalg.pinv(covMatrix)
    a = x-mean_i
    b = np.dot(a.T, cov_inverse)
    c = -np.dot(b, a)
    pp = -np.linalg.det(covMatrix) #改后代码——槐泽鹏

    discriminant_function_value = c + 2 * math.log(p_i)+pp #改后代码——槐泽鹏
    #print "result:", surface_fuction_value
    return discriminant_function_value
```

图 6.1 源代码 bug1

2. 在 MQDF 算法中，计算修正后的协方差矩阵时，源代码输出的直接是原协方差矩阵，并且特征值截断方式和较大特征值保留算法都很混乱。因此本人重新编写了此块代码。源代码和改后代码分别如图 6.2和图 6.3所示。

```
def modify_covariance(self, cov):
    a, b = np.linalg.eig(cov)
    a_modeified = [0 for i in range(len(a))]
    sum_a = sum(a)
    c = 0
    for i in range(len(a)):
        index_i = np.where(a==sorted(a,reverse=True)[i])[0][0]
        a_modeified[index_i] = a[index_i]
        c += a[index_i]
        cc = float(c)/sum_a
        delta = float(sum_a-c)/float(len(a) - index_i + 1)
        while 0.99 <= cc:
            a_modeified[index_i] = delta
            index_i +=1
            if len(a) == index_i:
                break
    cov_modeified = np.dot(np.dot(np.linalg.inv(b), mat(diag(a))),b)
    return cov_modeified
```

图 6.2 源代码 bug2

```
def modify_covariance(self, cov): self: <__main__.GaussianClassifiers object at 0x7f9e799eb
# 改后代码——槐泽鹏
a, b = np.linalg.eig(cov) a: [1.36372150e+05 1.50237042e+01 1.12973703e+01 6.92248397e
a_modeified = [0 for i in range(len(a))] a_modeified: [136372.14992853804, 3.299046448
a_modeified[-1]=a[-1]
sum_a = sum(a) sum_a: 136398.54230012503
c = 0 c: 136372.14992853804
for i in range(len(a)):
    index_i = i index_i: 0
    a_modeified[index_i] = a[index_i]
    c += a[index_i]
    cc = float(c)/sum_a cc: 0.9998065054718186
    if index_i<len(a)-1:
        delta = float(sum_a-c)/float(len(a) - index_i - 1) delta: 3.2990464483737014
        if 0.99 <= cc:
            for j in range(index_i+1, len(a)): j: 8
                a_modeified[j] = delta
            break
    cov_modeified = np.dot(np.dot(np.linalg.inv(b), mat(diag(a_modeified))),b) cov_modeified
return cov_modeified
```

图 6.3 源代码 bug2 改后代码

3. 原来代码中加载数据是采用在线加载的方式，我将数据集统一下载至 data 文件夹中并改为本地读取。