深圳大学研究生课程：模式识别理论与方法

课程作业实验报告

实验名称：Parzen 窗估计、k 近邻估计

实验编号：Proj04-01

签　　名：

姓　　名：夏荣杰

学　　号：2170269107

截止提交日期：2018 年 5 月 11 日

摘要： Parzen 窗估计和 k 最近邻估计是两种经典的非参数化估计法，它们都能从样本中估计出总体的概率密度分布，而不必事先考虑概率密度的参数形式。Parzen 窗估计法通过调整 Parzen 窗体积的范围来逼近真实的概率密度值，Parzen 窗估计法的实践依赖于窗函数和窗宽度的选择。而 k 近邻估计通过找出 k 个最近邻样本划分得到的空间进行概率密度估计，可直接通过求后验概率值将测试数据分类为与它最接近的 k 个近邻中出现最多的类别。实验一通过分别设计基于 Parzen 窗的贝叶斯分类器和概率神经网络 PNN 分类器对测试数据进行分类，并通过改变窗宽度 h 进行重复实验。实验结果表明，使用贝叶斯分类器和使用 PNN 分类器对于相同测试样本数据的分类结果是不同的；而它们在窗宽度 h 取不同值时的分类结果是相同的。实验二采用欧氏距离度量分别对一维数据、二维数据实现对测试点的 k 近邻概率估计，还设计了 k 近邻后验概率密度估计分类器实现对测试点的分类。实验过程中，对不同的 k 值进行实验。此外，通过改变距离度量方法对 k 近邻分类实验进行重复实验。实验结果表明，k 的取值会影响 k 近邻估计结果；使用不同距离度量对 k 近邻分类结果是有影响的。

一、 背景技术 或 基本原理

1. 非参数概率密度估计

非参数概率密度估计的基本思路是：特征空间中的点（矢量）**x** 落在区域 R 中的概率为

$$P = \int_R p(\mathbf{x}')d\mathbf{x}' \tag{1-1}$$

其中，P 是密度函数 p(**x**)的某种平均，因此可以通过计算 P 来估计概率密度函数 p(**x**)。

若已知样本数为 n，且每个样本都是独立同分布抽取的，则其中 k 个样本落在区域 R 中的概率服从二项式分布:

$$P_k = \binom{n}{k} P^k (1-P)^{n-k} \tag{1-2}$$

由二项式分布可知，k 的期望值为:

$$k_n = E(k) = nP \tag{1-3}$$

如果假设 p(x)是连续的，并且区域 R 足够小，以至于在这个区间内 p 几乎没有变化，则

$$P = \int_R p(\mathbf{x}')d\mathbf{x}' \approx p(\mathbf{x})V \tag{1-4}$$

其中 **x** 为一个点，而 V_n 则为区域 R 所包含的体积。

所以，

$$p(\mathbf{x}) \cong \frac{k_n / n}{V_n} \tag{1-5}$$

假如直接使用 $T = \frac{k_n / n}{V_n}$ 来估计概率密度是十分粗糙的，但是假如不断调整 V(使其趋近 0）或 k 的值使得 p(**x**) 趋近一个稳定值便可以用 T 来估计 p(**x**) 。根据调整方法的不同可以分为 Parzen 窗口法和 K 最近邻估计法。

2. Parzen 窗估计

Parzen 窗估计的核心思想是按一定规律不断缩小 V 使得 $\frac{k_n / n}{V_n}$ 趋近于真实的 p(**x**)。Parzen 窗泛化后可以使用其他核函数来取代事件的计数频率值。

在 Parzen 窗方法中，首先假设体积 V_n 是 d-维空间的一个超立方体，若 h 是该超立方体的边长，则体积为

$$V_n = h_n^d \tag{2-1}$$

概率密度估计公式为

$$p(\mathbf{x}) = \frac{P}{V_n} = \frac{k_n / n}{V_n} \tag{2-2}$$

在 Parzen 窗方法中，样本数 n 确定以后，体积 V_n 就确定了，但是落在超立方体中

的样本数未知。

通过定义窗函数，能够解析地得到落在窗中的样本个数 k_n 的表达式：

$$\varphi(\mathbf{u})=\begin{cases}1, |u_j|\le\frac{1}{2}, j=1,2,\cdots,d\\ 0, otherwise\end{cases} \tag{2-3}$$

$\varphi(\mathbf{u})$表示一个中心在原点的单位超立方体。若 $\mathbf{x}_i$ 落在超立方体 V_n 内部，则归一化函数 $\varphi\left((\mathbf{x}-\mathbf{x}_i)/h_n\right)=1$，否则便为 0。

因此，落入超立方体内部的样本数为

$$k_n=\sum_{i=1}^{n}\varphi\left(\frac{(\mathbf{x}-\mathbf{x}_i)}{h_n}\right) \tag{2-4}$$

代入公式得

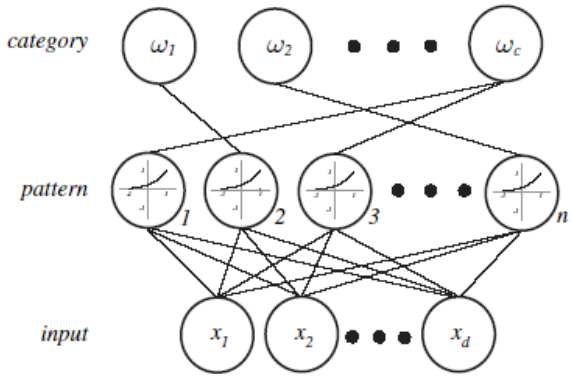
$$p_n(\mathbf{x})=\frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}\varphi\left(\frac{(\mathbf{x}-\mathbf{x}_i)}{h_n}\right) \tag{2-5}$$

即对 p(x)的估计 $P_n(x)$是关于 x 和样本 x_i 的窗函数的平均，每一个样本根据其到 x 的距离对函数值有不同的贡献。

上式中φ(μ)可以视为核函数，是具有对称衰减特性的径向基函数，这样其便能使用其他核函数进行替代，例如使用高斯核函数。

- Parzen 窗分类器的网络实现：概率神经网络 PNN

概率神经网络(probabilistic neural network, PNN)是由 d-维的输入单元、n 个模式单元，以及 c 个分类单元构成的三层网络结构，如下图所示。



每个模式单元的输入是其权值矢量和归一化的模式矢量 $\mathbf{x}$ 的内积 $z=\mathbf{w}^t\mathbf{x}$，其输出为 $\exp\left[(z-1)/\sigma^2\right]$。每一个分类单元的功能是计算由连接的各个模式单元输出值的和。这样，每个分类单元的激励代表了 Parzen 窗密度估计。

第 k 个分类单元的输出为：

$$p_{n_k}(x)=\frac{1}{n_k}\sum_{i=1}^{n_k}\frac{1}{\sqrt{2}\sigma}e^{-\frac{(\mathbf{x}-\mathbf{x}_i)^2}{2\sigma^2}} \tag{2-6}$$

若令$\left|\mathbf{x}\right|=\mathbf{x}^t\mathbf{x}=1$，$\left|\mathbf{w}_k\right|=\mathbf{w}_k{}^t\mathbf{w}_k=1$，则

$$\left(\mathbf{x}-\mathbf{w}_k\right)^t\left(\mathbf{x}-\mathbf{w}_k\right)=\mathbf{x}^t\mathbf{x}+\mathbf{w}_k{}^t\mathbf{w}_k-2\mathbf{x}^t\mathbf{w}_k=2-2\mathbf{x}^t\mathbf{w}_k \tag{2-7}$$

令$net_k=\mathbf{w}_k{}^t\mathbf{x}$，则

$$\left(\mathbf{x}-\mathbf{w}_k\right)^t\left(\mathbf{x}-\mathbf{w}_k\right)=-2\left(net_k-1\right) \tag{2-8}$$

所以，对于中心在某一个训练样本 $\mathbf{w}_k$ 处的高斯密度窗函数，有

$$\varphi\left(\frac{\mathbf{x}-\mathbf{w}_k}{h_n}\right)\propto\exp\left(-\frac{\left(\mathbf{x}-\mathbf{w}_k\right)^t\left(\mathbf{x}-\mathbf{w}_k\right)}{2\sigma^2}\right)=\exp\left[\left(net_k-1\right)/\sigma^2\right] \tag{2-9}$$

这样，每一个模式层单元向与它相连的那个类别层单元贡献了一个信号，这个信号的强度等于以当前训练样本为中心的高斯函数产生的这个测试点样本的概率。

对这些局部估计值求和得到概率密度函数的 Parzen 窗估计结果 $g_i\left(\mathbf{x}\right)$。通过 $\max\limits_{\mathbf{x}} g_i\left(\mathbf{x}\right)$计算得到测试点的期望的类别。

3. k 近邻估计

k 近邻估计基本思想：将 **x** 置于元胞的中心，逐渐增大元胞，直到元胞中包含 k 个样本为止。这里，k 值是样本数 n 的函数。

不论在何种情况下，估计 p(**x**)的方法是一样的：

$$p\left(\mathbf{x}\right)\cong\frac{k_n/n}{V_n} \tag{3-1}$$

其中，k_n 的一种取法可以为：

$$k_n=\sqrt{n} \tag{3-2}$$

在这种情况下，体积$V_n\approx1/\left(\sqrt{n}\cdot p\left(\mathbf{x}\right)\right)$。

以 x 为中心，构造胞形，使其包含 k 个样本。对 $p\left(\omega_j\mid\mathbf{x}\right)$的一个合理的估计是

$$p\left(\omega_j\mid\mathbf{x}\right)=\frac{k_j}{k} \tag{3-3}$$

其中，k_j 是落在胞形内部的 k 个样本中，属于类ω_j 的样本的数量。

k 近邻方法首先需要定义一个距离度量。

d 维空间的距离度量 L_p 范数(p > 0)为

$$L_p\left(\mathbf{x},\mathbf{y}\right)=\left(\sum_{i=1}^{d}\left|x_i-y_i\right|^p\right)^{1/p} \tag{3-4}$$

这样，欧几里得距离（欧式距离）就是 L_2 范数。

二、 实验方法 或 算法流程步骤

实验用到的方法和步骤如下：

1. Parzen 窗估计

- **设计贝叶斯分类器**

1) 采用的窗函数为窗口宽度值h相同的球状高斯函数：$\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)=e^{-\frac{(\mathbf{x}-\mathbf{x}_i)^T(\mathbf{x}-\mathbf{x}_i)}{2h^2}}$；

2) 根据式（2-5）计算类条件概率密度 $p_n(\mathbf{x}\,|\,w)$；

3) 先验概率$P(w_1)= P(w_2)= P(w_3)= 1/3$，根据 $g_j(\mathbf{x})=p(\mathbf{x}\,|\,w_j)\,p(w_j)$ 计算贝叶斯分类器函数g(x)；

4) 根据 $\mathbf{x}\in w_i, if\ \ i=\arg\max_j\{g_j(\mathbf{x})\,|\,j=1,2,...,c\}$ 训练最小错误率贝叶斯分类器；

5) 利用步骤4)训练的贝叶斯分类器对样本点$(0.5，1.0，0.0)^T$，$(0.31，1.51，-0.5)^T$，$(-0.3，0.44，-0.1)^T$进行分类。

- **设计 PNN 分类器**

■ 训练PNN网络的参数：

1) 对训练样本集中的每一个样本**x**进行归一化为单位长度；
2) 输入单元与模式层单元的链接权重初始化为$w_k=x_k$，k=1,2,...,n；
3) 模式层上相同类别的单元进行叠加，输出到类别层。

■ 利用PNN网络进行分类：

1) 将一个归一化了的样本点**x**提供给输入层节点；
2) 根据 $net_k=\mathbf{w}_k{}^t\mathbf{x}$，每一个模式层单元都计算内积，得到$net_k$；
3) 每一个类别层单元则把与它相连接的模式层单元的结果进行相加；
4) 根据式（9），计算判别函数g(**x**)，即概率密度函数的Parzen窗估计结果；
5) 通过 $\max_{\mathbf{x}} g_i(\mathbf{x})$ 计算得到测试点的期望的类别。

对贝叶斯分类器和PNN分类器的分类结果进行统计分析。

2. k 近邻估计

- 采用欧氏距离度量，对具有n个训练样本点的一维数据，实现对任给测试点x的k近邻概率估计：

1) 在有效区间生成100个测试点，训练样本点为类别w_3的特征x_1；
2) 根据式（3-4），计算测试点与训练样本点之间的欧式距离；
3) 对每个测试点到每个样本点之间的欧式距离进行从小到大的排序；
4) 取距离最近的k个点，并计算体积V_n，这里的体积为线段长度；
5) 根据式（3-1），计算测试点的概率密度p(**x**)，并分别画出当k=3、5时用程序计算出的概率密度估计的结果图。

- 采用欧氏距离度量，对具有n个训练样本点的二维数据，实现对任给测试点x的k近邻概率估计：
  1) 在有效区间生成100个测试点，训练样本点为类别w_2的特征$(x_1，x_1)^t$；
  2) 根据式（3-4），计算测试点与训练样本点之间的欧式距离；
  3) 对每个测试点到每个样本点之间的欧式距离进行从小到大的排序；
  4) 取距离最近的k个点，并计算体积V_n，这里的体积为面积大小；
  5) 根据式（3-1），计算测试点的概率密度p(x)，并分别画出当k=3、5时用程序计算出的概率密度估计的结果图。
- 采用欧氏距离度量，对已标记的具有三个类的三维训练数据，实现对任给测试点x的k近邻概率估计：
  1) 训练样本点为三个类别w_1、w_2和w_3的数据；
  2) 测试点为：$(-0.41，0.82，0.88)^T$，$(0.14，0.72，4.1)^T$，$(-0.81，0.61，-0.38)^T$；
  3) 根据式（3-4），计算测试点与训练样本点之间的欧式距离；
  4) 对每个测试点到每个样本点之间的欧式距离进行从小到大的排序；
  5) 取距离最近的k个训练样本点，计算属于各个类 w_j 的训练样本的数量；
  6) 取最大数量的类别作为k近邻概率估计的结果；
  7) 分别取k=3、5时，对测试点进行分类。
- **改变距离度量，重复实验：**
  1) 采用L-1距离度量：$L_1(\mathbf{x},\mathbf{y})=\sum_{i=1}^{d}|x_i-y_i|$，对已标记的具有三个类$w_1$、$w_2$和$w_3$的三维训练数据，实现对任给测试点x的k近邻概率估计；
  2) 采用L-∞距离度量：$L_\infty(\mathbf{x},\mathbf{y})=\max\left(|x_i-y_i|,i=1,2,...,d\right)$，对已标记的具有三个类$w_1$、$w_2$和$w_3$的三维训练数据，实现对任给测试点x的k近邻概率估计。

三、 实验结果

1. Parzen 窗估计

本实验设计一个采用上面的 Parzen 窗估计方法计算三维数据类条件概率密度的程序，并设计一个对三个类分类的贝叶斯分类器。

还设计一个可以对三个类分类的概率神经网络 PNN 分类器。

分别对上面中设计的两个分类器进行训练。然后用训练后的分类器对样本点$(0.5，1.0，0.0)^T$，$(0.31，1.51，-0.5)^T$，$(-0.3，0.44，-0.1)^T$进行分类。改变窗宽度 h 的值为 0.1，重复实验。当 h=1 和 h=0.1 时，贝叶斯分类器和 PNN 分类器的分类结果如图 1 所示。

```
--利用贝叶斯分类器对样本点进行分类(h=1.0)--
-------------样本点1：第2类----------------
-------------样本点2：第2类----------------
-------------样本点3：第2类----------------

--利用贝叶斯分类器对样本点进行分类(h=0.1)--
-------------样本点1：第2类----------------
-------------样本点2：第2类----------------
-------------样本点3：第2类----------------

--利用概率神经网络（PNN）分类器对样本点进行分类(h=1.0)--
---------------------样本点1：第3类---------------------
---------------------样本点2：第1类---------------------
---------------------样本点3：第1类---------------------

--利用概率神经网络（PNN）分类器对样本点进行分类(h=0.1)--
---------------------样本点1：第3类---------------------
---------------------样本点2：第1类---------------------
---------------------样本点3：第1类---------------------
```

图 1. h=1 和 h=0.1 时贝叶斯分类器和 PNN 分类器的分类结果

2. k 近邻估计

➢ **欧氏距离度量实验：**

采用欧氏距离度量，对类别 w_3 的特征 x_1，实现对任给测试点 x 的 k 近邻概率估计。分别画出当 k=3、5 时用程序计算出的概率密度估计的结果图，如图 2-1 所示：

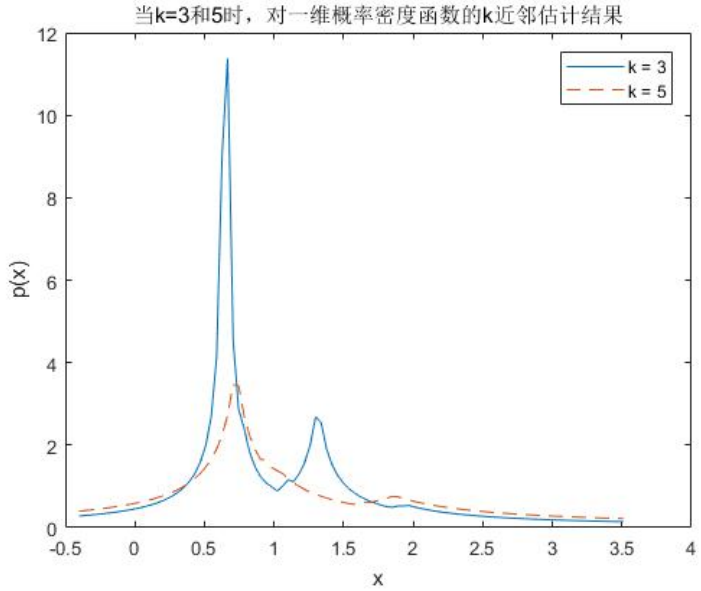


图 2-1. 当 k=3 和 5 时，对一维概率密度函数的 k 近邻估计结果

采用欧氏距离度量，对类别 w_2 的特征 $(x_1, x_1)^t$，实现对任给测试点 x 的 k 近邻概率估计。分别画出当 k=3、5 时用程序计算出的概率密度估计的结果图，如图 2-2 和图 2-3 所示：

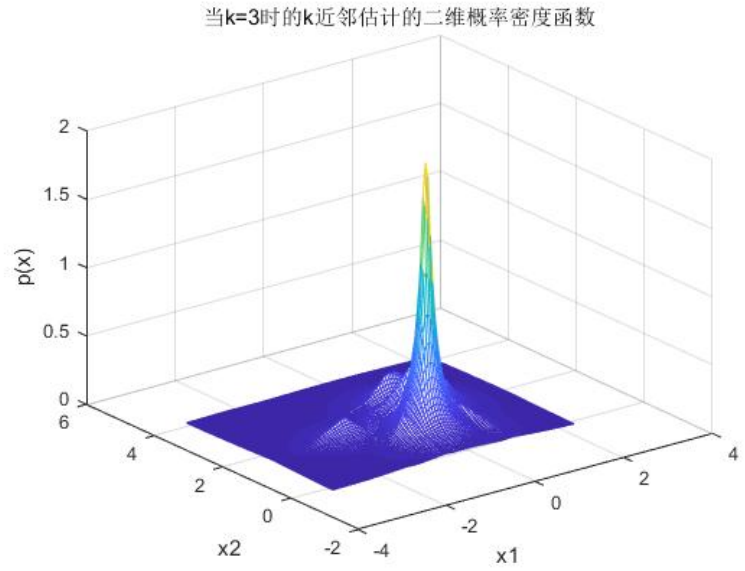


图 2-2. 当 k=3 时的 k 近邻估计的二维概率密度函数

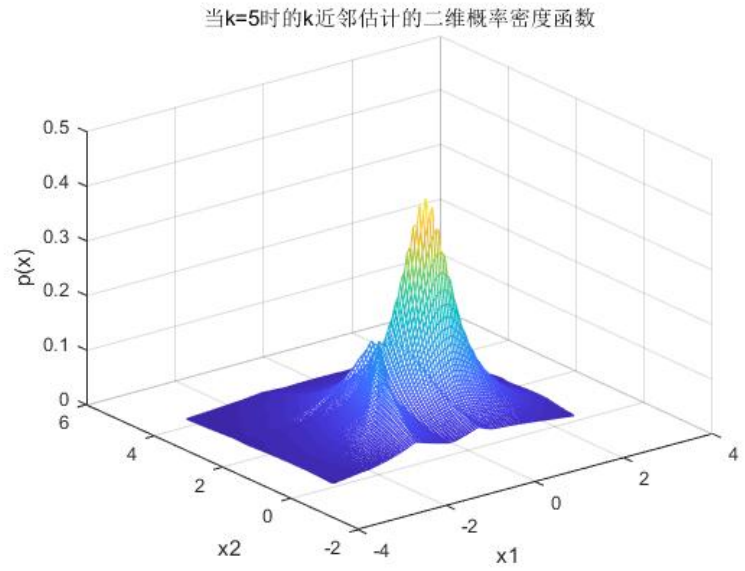


图 2-3. 当 k=5 时的 k 近邻估计的二维概率密度函数

采用欧氏距离度量，利用三个类别 w_1、w_2 和 w_3 的数据作为训练数据，在分别取 k=3、5 时，对测试点进行分类：$(-0.41，0.82，0.88)^T$，$(0.14，0.72，4.1)^T$，$(-0.81，0.61，-0.38)^T$。分类结果如图 2-4 所示：

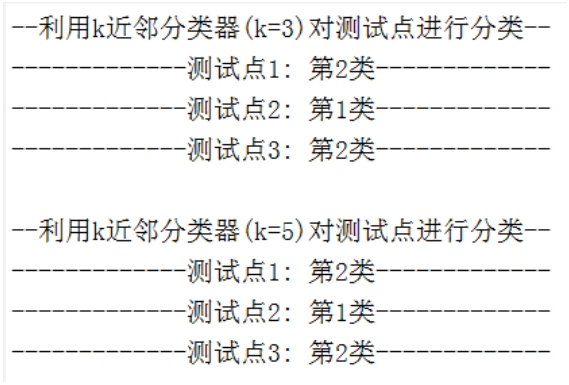


图 2-4. 采用欧氏距离度量，当 k=3 和 5 时，k 近邻分类器的分类结果

- **改变距离度量，重复实验：**

采用 L-1 距离度量：$L_1(\mathbf{x},\mathbf{y})=\sum_{i=1}^{d}|x_i-y_i|$，利用三个类别 w1、w2 和 w3 的数据作为训练数据，在分别取 k=3、5 时，对测试点进行分类：$(-0.41，0.82，0.88)^T$，$(0.14，0.72，4.1)^T$，$(-0.81，0.61，-0.38)^T$。分类结果如图 2-5 所示：

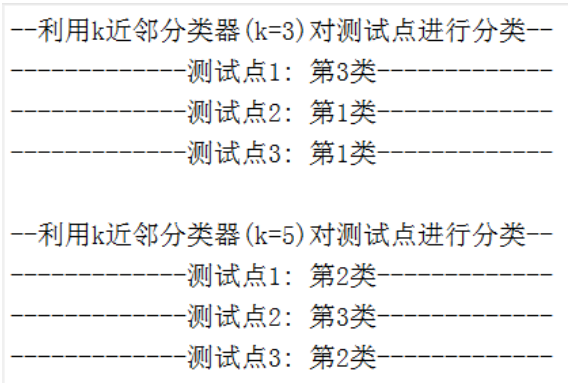


图 2-5. 采用 L-1 距离度量，当 k=3 和 5 时，k 近邻分类器的分类结果

采用 L-∞距离度量：$L_\infty(\mathbf{x},\mathbf{y})=\max\left(|x_i-y_i|,i=1,2,...,d\right)$，利用三个类别 w1、w2 和 w3 的数据作为训练数据，在分别取 k=3、5 时，对测试点进行分类：$(-0.41，0.82，0.88)^T$，$(0.14，0.72，4.1)^T$，$(-0.81，0.61，-0.38)^T$。分类结果如图 2-6 所示：

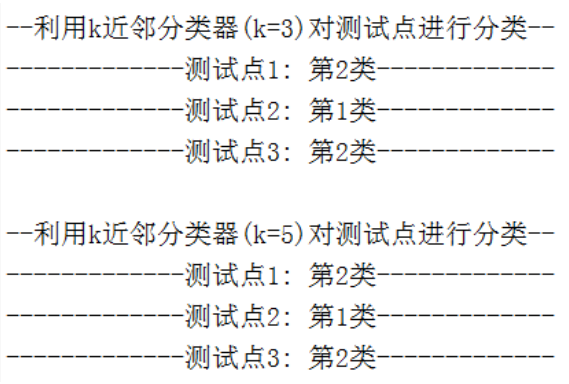


图 2-6. 采用 L-∞距离度量，当 k=3 和 5 时，k 近邻分类器的分类结果

四、 讨论与分析

1. Parzen 窗估计

在窗宽度 h=1 和 h=0.1 时，设计的贝叶斯分类器和 PNN 分类器的分类结果对比如表 1 所示。

表 1. 贝叶斯分类器和 PNN 分类器的分类结果

		样本点 1 $(0.5，1.0，0.0)^T$	样本点 2 $(0.31，1.51，-0.5)^T$	样本点 3 $(-0.3，0.44，-0.1)^T$
h=1	贝叶斯分类器	2	2	2
	PNN 分类器	3	1	1
h=0.1	贝叶斯分类器	2	2	2
	PNN 分类器	3	1	1

从表 1 可以看到，使用贝叶斯分类器和使用 PNN 分类器对于相同测试样本数据的分类结果是不同的。贝叶斯分类器在 h 取不同值时的分类结果是相同的；PNN 分类器在 h 取不同值时的分类结果是相同的。

基于 Parzen 窗的贝叶斯分类器和 PNN 分类器的异同：

- **相同点：**

① 基于 Parzen 窗的贝叶斯分类器计算训练样本的类条件概率；PNN 分类器的模式层单元向与它相连接的那个类别层单元的信号等于以当前训练样本为中心的高斯函数产生这个测试点样本的概率。可由式（2-9）得知，两个概率的形式是一样的，即 PNN 分类器的模式层的激活函数与 Parzen 窗函数相同。

② PNN 分类器的关键问题实际上和 Parzen 窗一样，就是选取体积序列（在这里是选取σ=h），如果过大，会导致平滑效果太过明显，太小会导致变化特别剧烈，分类效果都不会太好。

- **不同点：**

① 基于 Parzen 窗的贝叶斯分类器采用最小错误率准则，首先计算训练样本的类条件概率，再计算测试样本 x 的后验概率，选择具有最大后验概率的类作为该对象所属的类；而 PNN 分类器对模式层输出中相同类别进行求和得到判别函数——概率密度函数的 Parzen 窗估计结果，进而通过最大化判别函数得到测试点的期望的类别。

② 贝叶斯分类器考虑先验概率和似然概率，实现了最小错误率意义上的优化；PNN 分类器的好处之一就是学习速度很快，因为规则简单（$w_k=x_k$），而且新的训练样本容易被加入以前训练好的分类器中。

- **按理说**，改变 h 值，基于 Parzen 窗的贝叶斯分类器和 PNN 分类器的分类结果会发生变化，但可能是由于训练样本点太少，实验的分类结果在不同的 h 值下都是相同的。

2. k 近邻估计

由图 2-1 可知，当 k=3 和 5 时，对一维概率密度函数的 k 近邻估计结果，由于 n=100，为有限值，估计出的斜率是不连续的；

由图 2-2 和图 2-3 可知，当 k=3 和 5 时，对二维概率密度函数的 k 近邻估计结果，由于 n=100，为有限值，估计出的密度相当“崎岖”，存在斜率上的不连续的；随着 k 的增大，概率密度峰值减小，且双峰分布较为明显。这是因为，当 k 取不同的值时，得

到的体积 V_n 会不同，根据式（2-1）可知，概率密度估计值 $p_n(x)$会不同。另外，在一维情况下，V_n 计算的是两点间的距离，而二维的情况下，V_n 计算的是两点间的距离为半径的圆的面积。在三维的情况下，V_n 计算的是两点之间的距离为半径的圆的体积。

采用欧氏距离度量，L-1 距离度量和 L-∞距离度量，利用三个类别 w_1、w_2 和 w_3 的数据作为训练数据，在分别取 k=3、5 时，对测试点进行分类：$(-0.41，0.82，0.88)^T$，$(0.14，0.72，4.1)^T$，$(-0.81，0.61，-0.38)^T$。分类结果对比如表 2 所示：

表 2. 欧氏距离度量，L-1 距离度量和 L-∞距离度量的 k 近邻分类结果

距离度量	k	测试点 1 $(-0.41，0.82，0.88)^T$	测试点 2 $(0.14，0.72，4.1)^T$	测试点 3 $(-0.81，0.61，-0.38)^T$
L-2（欧氏距离）	3	2	1	2
	5	2	1	2
L-1	3	3	1	1
	5	2	3	2
L-∞	3	2	1	2
	5	2	1	2

- **K 的取值对 k 近邻估计的影响：**

从表 2 可以看到，使用欧氏距离度量，k=3 和 k=5 时 k 近邻分类结果相同；使用 L-1 距离度量，k=3 和 k=5 时 k 近邻分类结果不相同；使用 L-∞距离度量，k=3 和 k=5 时 k 近邻分类结果相同。由此可见，**k 的取值会影响 k 近邻估计结果。**这是因为，当 k 取不同的值时，得到的体积 V_n 会不同，根据式（2-1）可知，概率密度估计值 $p_n(x)$会不同。

个人猜测，由于训练样本点较少，使用欧氏距离度量和 L-∞距离度量，k=3 和 k=5 时 k 近邻分类结果相同。

- **距离度量的选取对 k 近邻估计的影响：**

纵观表 2 三种距离度量的分类结果可知，使用欧氏距离度量和使用 L-∞距离度量的分类结果是相同的，而与使用 L-1 距离度量的分类结果不同。因此可以得出结论：**使用不同距离度量对 k 近邻分类结果是有影响的。**这是因为，度量距离的计算方法不同，导致计算得到的与测试样本点最接近 k 的近邻不同，测试样本点分类为这 k 个近邻中出现最多的那个类别的结果便会不同，即 k 近邻分类结果不同。

个人猜测，由于训练样本点较少，导致使用欧氏距离度量和 L-∞距离度量的 k 近邻分类结果相同。

3. Parzen 窗估计和 k 近邻估计的比较与分析

由上面的 Parzen 窗估计和 k 近邻估计的实验结果分析可以知道：

Parzen 窗估法和 k 近邻估计都能从样本中估计出总体的概率密度分布。Parzen 窗估计法通过调整 V_n 的范围来逼近真实的概率密度值，Parzen 窗估计法的实践依赖于窗函数 φ 和窗宽度 h 的选择。而 k 近邻估计通过找出 k 个最近邻样本划分得到的空间进行概率密度估计。k 近邻分类的实现可以绕过概率密度估计而直接求后验概率值，如式（3-3）。相比于 Parzen 窗估计法，k 近邻估计不需要设定窗宽度值 h，只需设定好距离度量 L 和 k 值就可以进行分类，并给出较好的分类结果。

附录.

1. Parzen 窗估计

```
%%----------------Proj04-01：Parzen 窗估计、k 近邻估计--------------%%
%%-----------------Proj04-01-exp1：Parzen 窗估计-------------------%%
clear; clc;
N = 10;%每类样本数量
c = 3;%类别数目
%%第一类
w1 = [0.28 1.31 -6.2; 0.07 0.58 -0.78; 1.54 2.01 -1.63; -0.44 1.18 -4.32; -0.81
0.21 5.73;
    1.52 3.16 2.77; 2.20 2.42 -0.19; 0.91 1.94 6.21; 0.65 1.93 4.38; -0.26 0.82
-0.96];
%%第二类
w2 = [0.011 1.03 -0.21; 1.27 1.28 0.08; 0.13 3.12 0.16; -0.21 1.23 -0.11; -2.18
1.39 -0.19;
    0.34 1.96 -0.16; -1.38 0.94 0.45; -0.12 0.82 0.17; -1.44 2.31 0.14; 0.26 1.94
0.08];
%%第三类
w3 = [1.36 2.17 0.14; 1.41 1.45 -0.38; 1.22 0.99 0.69; 2.46 2.19 1.31; 0.68 0.79
0.87;
    2.51 3.22 1.35; 0.60 2.44 0.92; 0.64 0.13 0.97; 0.85 0.58 0.99; 0.66 0.51 0.88];
%%样本点
sample1 = [0.5, 1.0, 0.0]'; sample2 = [0.31, 1.51, -0.5]'; sample3 = [-0.3, 0.44,
-0.1]';

%% 利用设计的基本 Parzen 窗估计分类器进行分类
p_w1 = 1/3; p_w2 = 1/3; p_w3 = 1/3;%%贝叶斯分类器的先验概率
for h = [1, 0.1]%parzen 窗宽度
    [f_max1, pre_b1] = Bayes_classifier(sample1, w1, w2, w3, N, h, p_w1, p_w2,
p_w3);%%贝叶斯分类器对样本点进行分类
    [f_max2, pre_b2] = Bayes_classifier(sample2, w1, w2, w3, N, h, p_w1, p_w2,
p_w3);
    [f_max3, pre_b3] = Bayes_classifier(sample3, w1, w2, w3, N, h, p_w1, p_w2,
p_w3);
    fprintf('--利用贝叶斯分类器对样本点进行分类(h=%.1f)--\n', h);
    fprintf('-------------样本点 1: 第%d 类----------------\n', pre_b1);
    fprintf('-------------样本点 2: 第%d 类----------------\n', pre_b2);
    fprintf('-------------样本点 3: 第%d 类----------------\n\n', pre_b3);
end
%% 利用设计的概率神经网络（PNN）分类器进行分类
for h = [1, 0.1]%parzen 窗宽度
    sigma = h;%方差参数
    [weight, a] = PNN_train(w1, w2, w3);%%PNN 训练
```

```
    class1 = PNN_classifier(sample1, weight, a, sigma);%%PNN 分类器对样本点进行分
类
    class2 = PNN_classifier(sample2, weight, a, sigma);
    class3 = PNN_classifier(sample3, weight, a, sigma);
    fprintf('--利用概率神经网络（PNN）分类器对样本点进行分类(h=%.1f)--\n', h);
    fprintf('----------------------样本点 1: 第%d 类----------------------\n',
class1);
    fprintf('----------------------样本点 2: 第%d 类----------------------\n',
class2);
    fprintf('----------------------样本点 3: 第%d 类----------------------\n\n',
class3);
end

%% 子函数
function p_x_w = class_pdf(sample, w, N, h)
%%这个函数用于计算类条件概率密度
%%输入：sample 为样本点，w 为训练数据，N 为训练数据数量，h 为 parzen 窗宽度
%%输出：p_x_w 为类条件概率密度
p_x_w = 0;
for i = 1: N
    parzen = exp((-(w(i, :)' - sample)' * (w(i, :)' - sample))/(2 * h^2));%窗函
数，标量
    p_x_w = p_x_w + parzen;%%类条件概率密度
end
p_x_w = (1 / N) * p_x_w;
end

function [f_max, pre_b] = Bayes_classifier(sample, w1, w2, w3, N, h, p_w1, p_w2,
p_w3)
%%这个函数用于设计基本 Parzen 窗估计分类器：一个对三个类分类的贝叶斯分类器
%%输入：sample 为样本点，w1、w2、w3 为训练数据，N 为训练数据数量，h 为 parzen 窗宽度，p_w1、
p_w2、p_w3 为贝叶斯分类器先验概率
%%输出：f_max 为最大判别函数值， pre_b 为 f_max 对应的分类结果
p1 = class_pdf(sample, w1, N, h); %分别计算类条件概率
p2 = class_pdf(sample, w2, N, h);
p3 = class_pdf(sample, w3, N, h);
g1 = p1 * p_w1; %分类器函数
g2 = p2 * p_w2;
g3 = p3 * p_w3;
[f_max, pre_b] = max([g1; g2; g3]);
end

function [weight, a] = PNN_train(w1, w2, w3)
%%这个函数用于训练 PNN 网络的参数
```

```
%%输入：w1、w2、w3 为训练数据
%%输出：w 为训练得到的权重参数，a 为索引
w1 = normr(w1); w2 = normr(w2); w3 = normr(w3);%归一化
weight1 = w1; weight2 = w2; weight3 = w3;%学习规则
weight = [weight1; weight2; weight3];
a = zeros(size(weight));
for i = 1: size(weight, 1)
    if i <= size(w1, 1)
        a(i, 1) = 1;
    end
    if i > size(w1, 1) && i < (size(w1, 1) + size(w2, 1))
        a(i, 2) = 1;
    end
    if i >= (size(w1, 1) + size(w2, 1))
        a(i, 3) = 1;
    end
end
end

function class = PNN_classifier(sample, weight, a, sigma)
%%这个函数用于 PNN 网络的分类
%%输入：sample 为测试样本点，weight 为训练权重参数，a 为索引，sigma 为方差
%%输出：class 为分类结果
sample = normr(sample);%测试点归一化
net = zeros(size(weight, 1), 1);
g = zeros(size(weight));
for k = 1: size(net, 1)
    net(k, :) = weight(k, :) * sample;
    if a(k, 1) == 1
        g(k, 1) = g(k, 1) + exp((net(k) - 1) / (sigma^2));
    end
    if a(k, 2) == 1
        g(k, 2) = g(k, 1) + exp((net(k) - 1) / (sigma^2));
    end
    if a(k, 3) == 1
        g(k, 3) = g(k, 3) + exp((net(k) - 1) / (sigma^2));
    end
end
[~, class] = max(max(g));
end
```

2. k 近邻估计

```
%%----------------Proj04-01：Parzen 窗估计、k 近邻估计--------------%%
%%------------------Proj04-01-exp1：k 近邻估计---------------------%%
clear; clc;
N = 10;%每类样本数量
c = 3;%类别数目
%%第一类
w1 = [0.28 1.31 -6.2; 0.07 0.58 -0.78; 1.54 2.01 -1.63; -0.44 1.18 -4.32; -0.81
0.21 5.73;
    1.52 3.16 2.77; 2.20 2.42 -0.19; 0.91 1.94 6.21; 0.65 1.93 4.38; -0.26 0.82
-0.96];
%%第二类
w2 = [0.011 1.03 -0.21; 1.27 1.28 0.08; 0.13 3.12 0.16; -0.21 1.23 -0.11; -2.18
1.39 -0.19;
    0.34 1.96 -0.16; -1.38 0.94 0.45; -0.12 0.82 0.17; -1.44 2.31 0.14; 0.26 1.94
0.08];
%%第三类
w3 = [1.36 2.17 0.14; 1.41 1.45 -0.38; 1.22 0.99 0.69; 2.46 2.19 1.31; 0.68 0.79
0.87;
    2.51 3.22 1.35; 0.60 2.44 0.92; 0.64 0.13 0.97; 0.85 0.58 0.99; 0.66 0.51 0.88];
%% 采用欧氏距离度量，对具有 n 个训练样本点的一维数据，实现对任给测试点 x 的 k 近邻概率估计。
sample1 = w3(:, 1);
for k = [3, 5]%k 参数
    n = 100;%测试点数量
    L = zeros(n, N);%欧式距离
    test = linspace(min(sample1) - 1, max(sample1) + 1, n);%测试点
    for i = 1: n
        for j = 1: N
            L(i, j) = sqrt((test(i) - sample1(j))^2);
        end
    end
    [a, b] = sort(L, 2);%对每个测试点到每个样本点之间的欧式距离进行排序，a 为重新排好序，
b 为索引
    v1 = a(:, k);%%取前 k 列，即距离最近的 k 个，可视为体积 V
    p = (k / N) ./ v1;
    if k == 3
        l1 = plot(test, p, '-');
    else
        l2 = plot(test, p, '--');
    end
    hold on;
end
legend([l1, l2], 'k = 3', 'k = 5');
xlabel('x'); ylabel('p(x)');
```

```
title('当 k=3 和 5 时，对一维概率密度函数的 k 近邻估计结果');

%% 采用欧氏距离度量，对具有 n 个训练样本点的二维数据，实现对任给测试点 x 的 k 近邻概率估计。
sample2 = w2(:, 1: 2);
for k = [3, 5]%k 参数
    n = 100;%测试点数量
    L = zeros(n, N);%欧式距离
    test_x = linspace(min(sample2(:, 1)) - 1, max(sample2(:, 1)) + 1, n);%测试点
    test_y = linspace(min(sample2(:, 2)) - 1, max(sample2(:, 2)) + 1, n);
    [X, Y] = meshgrid(test_x, test_y);
    for i = 1: size(X(:), 1)
        for j = 1: N
            L(i, j) = sqrt((X(i) - sample2(j, 1))^2 + (Y(i) - sample2(j, 2))^2);%欧式距离
        end
    end
    [a, b] = sort(L, 2);%对每个测试点到每个样本点之间的欧式距离进行排序，a 为重新排好序，b 为索引
    v2 = pi * (a(:, k).^2);%%取前 k 列，即距离最近的 k 个，计算体积 V
    p = (k / N) ./ v2;
    p = reshape(p, size(X));%将行向量转换为矩阵向量
    if k == 3
        figure; mesh(X, Y, p); title('当 k=3 时的 k 近邻估计的二维概率密度函数');
    else
        figure; mesh(X, Y, p); title('当 k=5 时的 k 近邻估计的二维概率密度函数');
    end
    xlabel('x1'); ylabel('x2'); zlabel('p(x)');
end

%% 采用欧氏距离度量，对已标记的具有三个类的三维训练数据，实现对任给测试点 x 的 k 近邻概率估计。
test1 = [-0.41, 0.82, 0.88]'; test2 = [0.14, 0.72, 4.1]'; test3 = [-0.81, 0.61, -0.38]';%测试点
w = cat(3, w1, w2, w3);
for k = [3, 5]%k 参数
    class1 = KNN_classifier(w, test1, k);
    class2 = KNN_classifier(w, test2, k);
    class3 = KNN_classifier(w, test3, k);
    fprintf('--利用 k 近邻分类器(k=%d)对测试点进行分类--\n', k);
    fprintf('-------------测试点 1: 第%d 类-------------\n', class1);
    fprintf('-------------测试点 2: 第%d 类-------------\n', class2);
    fprintf('-------------测试点 3: 第%d 类-------------\n\n', class3);
end
```

```
%% 子函数
function class = KNN_classifier(w, test, k)
%%该函数用于 k 近邻算法分类
%%输入：w 为训练样本集合，test 为测试样本，k 为 k 近邻参数
%%输出：class 为 k 个近邻中出现最多的那个类别，即分类结果
    N = size(w, 1); c = size(w, 3);
    L = zeros(N, c);%欧式距离
    for i = 1: c
        for j = 1: N
            L(j, i) = sqrt((test(1) - w(j, 1, i))^2 + (test(2) - w(j, 2, i))^2 +
(test(3) - w(j, 3, i))^2);%欧式距离
%           L(j, i) = abs(test(1) - w(j, 1, i)) + abs(test(2) - w(j, 2, i)) +
abs(test(3) - w(j, 3, i));%L1 距离
%           L(j, i) = max(max(abs(test(1) - w(j, 1, i)), abs(test(2) - w(j, 2,
i))), abs(test(3) - w(j, 3, i)));%L 无穷距离
        end
    end
    t = sort(L(:));%对每个测试点到每个样本点之间的欧式距离进行排序，a 为重新排好序，b 为索
引
    [~, n] = find(L <= t(k), k);
    index = 1: max(n);
    h = histc(n, index);
    [~, max_index] = max(h);
    class = index(max_index);
end
```