模式识别—实验二：概率密度估计的非参数方法

实验报告

姓名　　文泓力　　学号　　202111150036　　专业　　人工智能

一、实验目的和要求

1、 理解概率密度估计的非参数方法的基本原理；

2、 熟练掌握非参数估计方法中的K_N近邻估计法和 Parzen 窗法；

3、 要求写出每个实验对应的算法原理并写出对应代码的测试步骤；

二、实验内容

实验：概率密度估计的两种非参数方法的实现

1、 问题描述：

已知四组一维数据集，样本数分别为 16、256、1000 和 2000，请分别使用K_N近邻估计法和 Parzen 窗法估计四组数据集的概率密度函数：

（1） K_N近邻估计法：不同样本下对比；

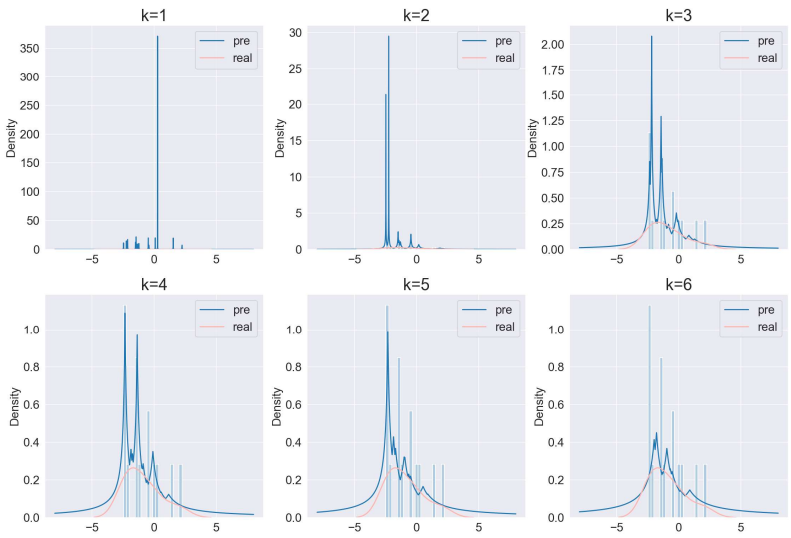


图 1　k 近邻 16 点

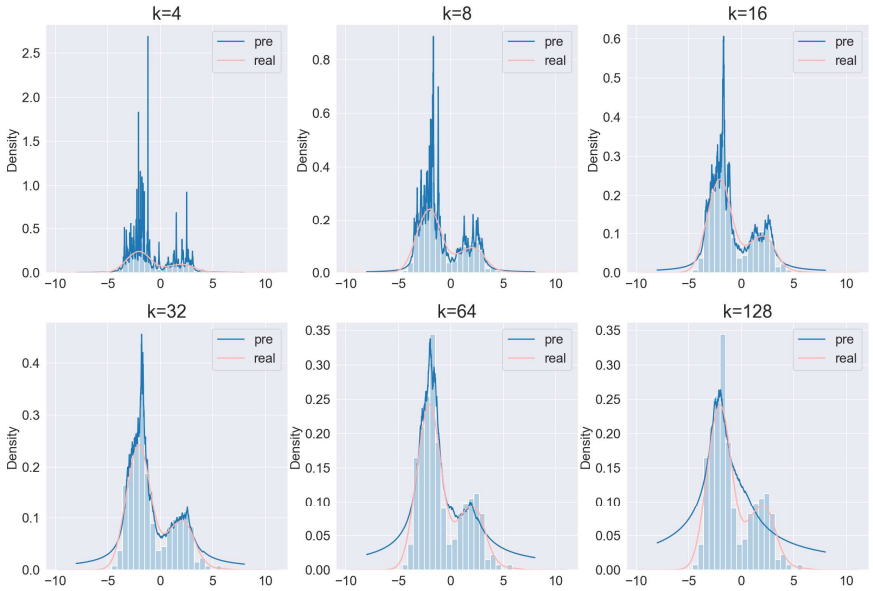


图 2 k 近邻 256 点

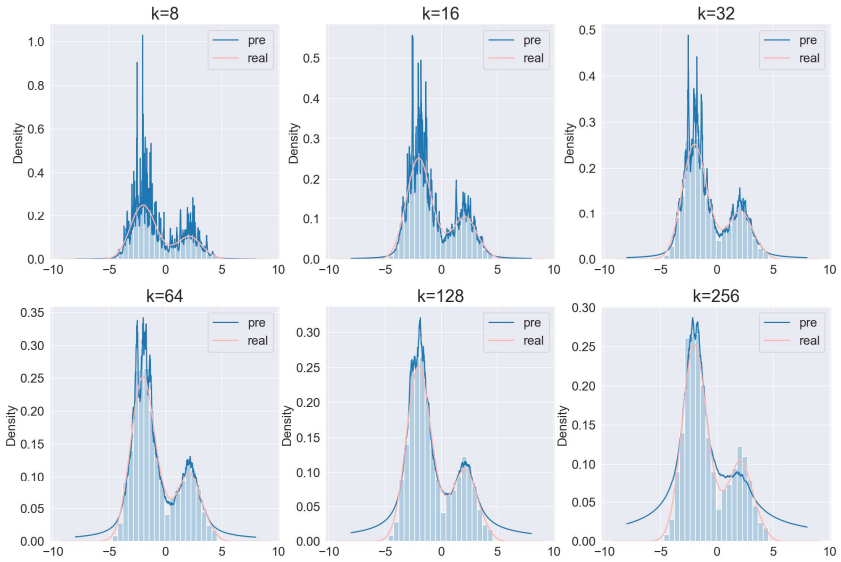


图 3 k 近邻 1000 点

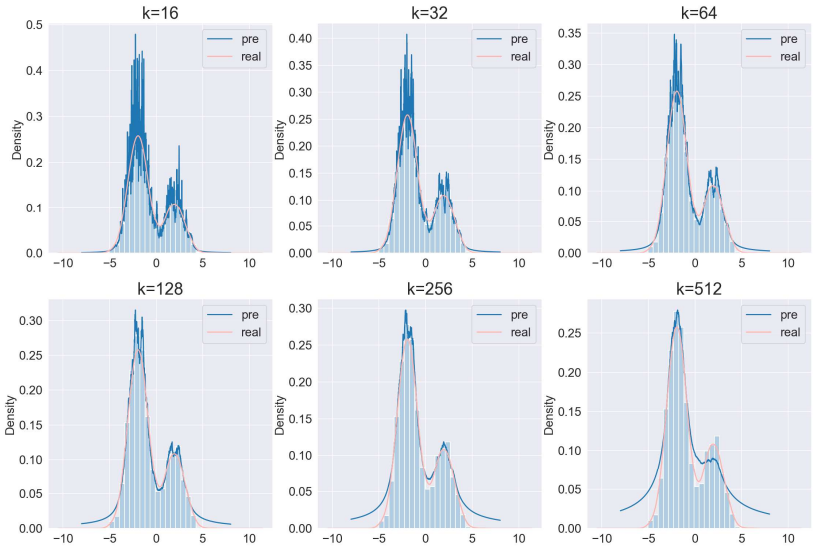


图 4　k 近邻 2000 点

（2）　Parzen 窗法：核函数选择方窗和高斯窗，窗口大小设置为 0.25，1，4，不同样本下对比。

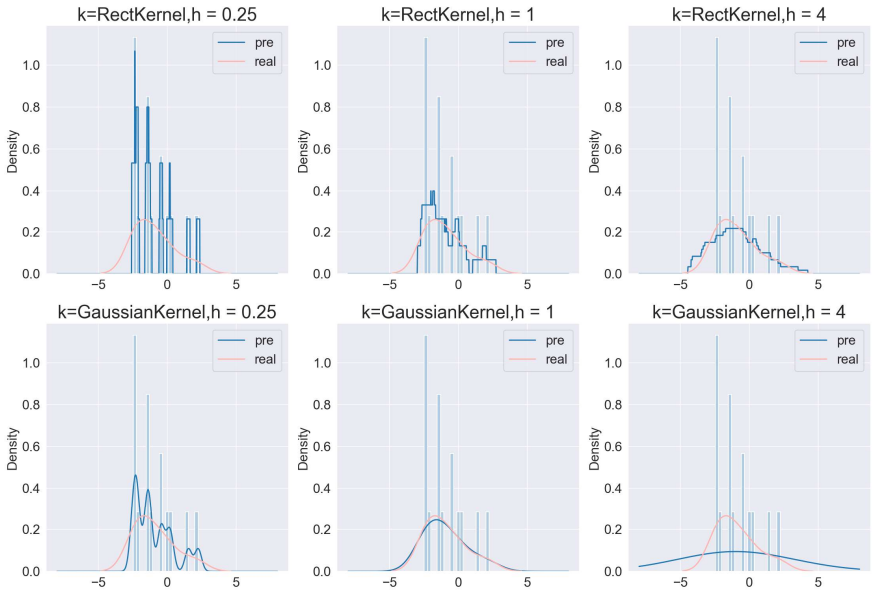


图 5　核函数 16 点

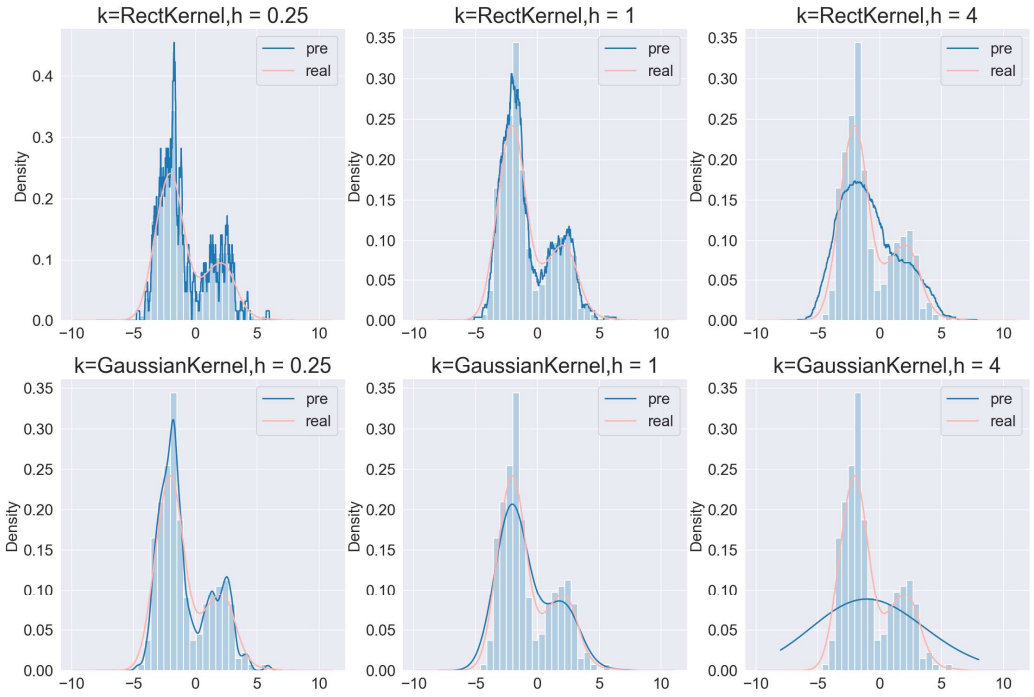

k=RectKernel,h = 0.25
k=RectKernel,h = 1
k=RectKernel,h = 4
k=GaussianKernel,h = 0.25
k=GaussianKernel,h = 1
k=GaussianKernel,h = 4
pre
real
Density

图 6　核函数 256 点

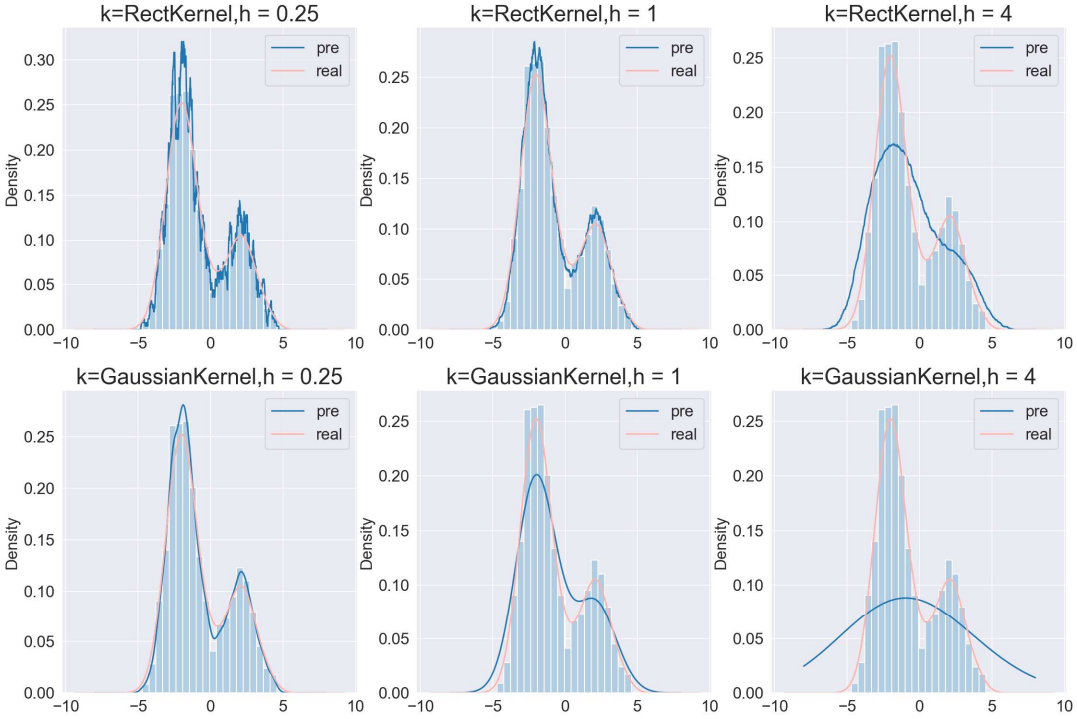

k=RectKernel,h = 0.25
k=RectKernel,h = 1
k=RectKernel,h = 4
k=GaussianKernel,h = 0.25
k=GaussianKernel,h = 1
k=GaussianKernel,h = 4
pre
real
Density

图 7　核函数 1000 点

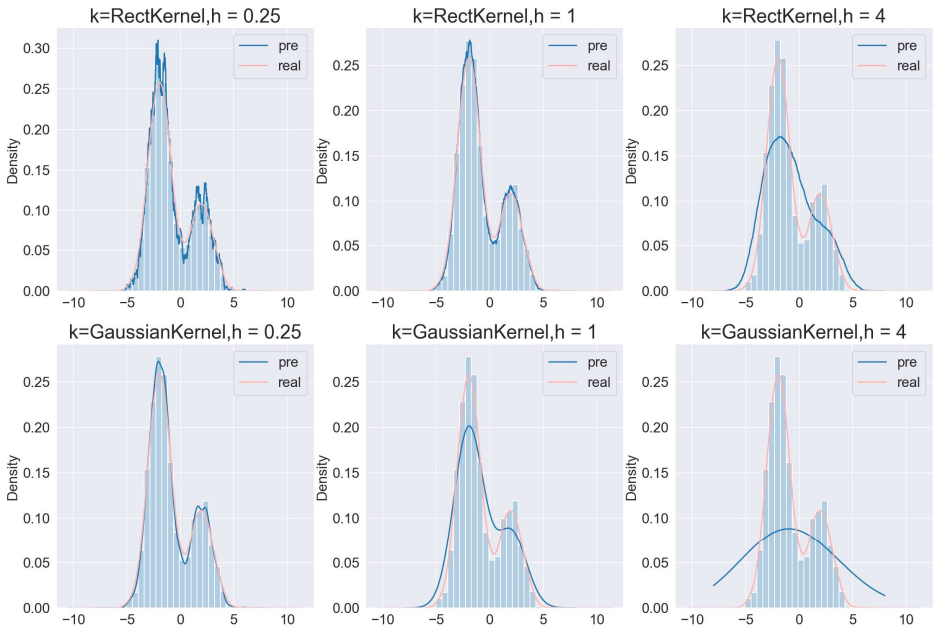


图 8　核函数 2000 点

2、实验内容：

（1）　画出在两种估计法下，不同设置估计得到的概率密度图（如上所示）

（2）　算法原理解释

1．非参数化概率密度的估计

对于未知概率密度函数的估计方法，其核心思想是：一个向量 x 落在区域 R 中的概率可表示为：

$$P = \int_R p(\boldsymbol{x})d\boldsymbol{x}$$

其中，P 是概率密度函数 p(x)的平滑版本，因此可以通过计算 P 来估计概率密度函数 p(x)，假设 n 个样本 x1,x2,⋯,xn，是根据概率密度函数 p(x)独立同分布的抽取得到，这样，有 k 个样本落在区域 R 中的概率服从以下分布：

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

其中 K 的期望满足：

$$E(k) = nP$$

k 的分布在均值附近有着非常显著的波峰，因此若样本个数 n 足够大时，使用 k/n 作为概率 P 的一个估计将非常准确。假设 p(x)是连续的，且区域 R 足够小，则有：

$$\int_R p(\boldsymbol{x})d\boldsymbol{x} \approx p(\boldsymbol{x})V$$

如下图所示，以上公式产生一个特定值的相对概率，当 n 趋近于无穷大时，曲线的形状逼近一个δ函数，该函数即是真实的概率。公式中的 V 是区域 R 所包含的体积。综上所述，可以得到关于概率密度函数 p(x)的估计为：

$$p(\boldsymbol{x}) \approx \frac{k/n}{V}$$

在实际中，为了估计 x 处的概率密度函数，需要构造包含点 x 的区域 R1,R2,⋯,Rn。第一个区域使用 1 个样本，第二个区域使用 2 个样本，以此类推。记 Vn 为 Rn 的体积。kn 为落在区间 Rn 中的样本个数，而 pn (x)表示为对 p(x)的第 n 次估计：

$$p_n(\boldsymbol{x}) = \frac{k_n/n}{V_n}$$

欲满足 pn(x)收敛：pn(x)→p(x)，需要满足以下三个条件：

$$\lim_{n\to\infty} V_n = 0$$

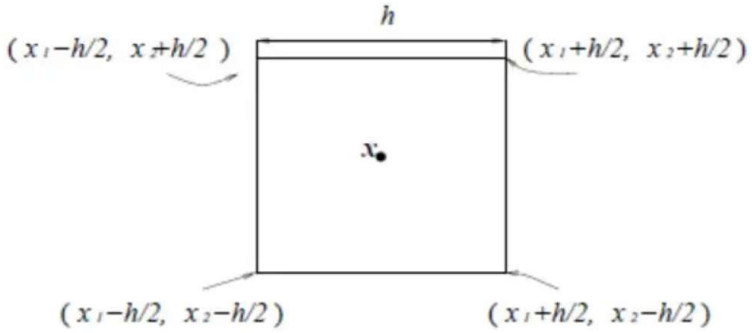
$$\lim_{n\to\infty} k_n = \infty$$

$$\lim_{n\to\infty} k_n/n = 0$$

有两种经常采用的获得这种区域序列的途径，如下图所示。其中“Parzen 窗方法”就是根据某一个确定的体积函数，比如 Vn=1/√n 来逐渐收缩一个给定的初始区间。这就要求随机变量 kn 和 kn/n 能够保证 pn (x)能收敛到 p(x)。第二种“k-近邻法”则是先确定 kn 为 n 的某个函数，如 kn=√n。这样，体积需要逐渐生长，直到最后能包含进 x 的 kn 个相邻点。

2. Parzen 窗估计法

已知测试样本数据 x1,x2,⋯,xn，在不利用有关数据分布的先验知识，对数据分布不附加任何假定的前提下，假设 R 是以 x 为中心的超立方体，h 为这个超立方体的边长，对于二维情况，方形中有面积 V=h^2，在三维情况中立方体体积 V=h^3，如下图所示。



根据以下公式，表示 x 是否落入超立方体区域中：

$$\varphi\left(\frac{x-x_i}{h}\right)=\begin{cases}1 & \frac{|x_{ik}-x_k|}{h}<\frac{1}{2},k=1,2,\ldots\\ 0 & \text{其他}\end{cases}$$

估计它的概率分布：

$$p(x)=\frac{1}{nV}\sum_{i=1}^{N}\varphi\left(\frac{x-x_i}{h}\right)$$

其中 n 为样本数量，h 为选择的窗的长度，φ(.)为核函数，通常采用矩形窗和高斯窗。

3．k 最近邻估计

在 Parzen 算法中，窗函数的选择往往是个需要权衡的问题，k-最近邻算法提供了一种解决方法，是一种非常经典的非参数估计法。基本思路是：已知训练样本数据 x1,x2,…,xn 而估计 p(x)，以点 x 为中心，不断扩大体积 Vn，直到区域内包含 k 个样本点为止，其中 k 是关于 n 的某一个特定函数，这些样本被称为点 x 的 k 个最近邻点。

当涉及到邻点时，通常需要计算观测点间的距离或其他的相似性度量，这些度量能够根据自变量得出。这里我们选用最常见的距离度量方法：欧几里德距离。

最简单的情况是当 k=1 的情况，这时我们发现观测点就是最近的（最近邻）。一个显著的事实是：这是简单的、直观的、有力的分类方法，尤其当我们的训练集中观测点的数目 n 很大的时候。可以证明，k 最近邻估计的误分概率不高于当知道每个类的精确概率密度函数时误分概率的两倍。

（3） 解释实验结果。

（实验超参数：1.采样恒为 1600 点采样，即对-8，8 内均匀取 1600 点作为测试样本，估计这些点的概率密度，以获得分布曲线。2.高斯窗采用标准正态分布）

- 从 K 近邻非参数密度估计可以看出，较少的样本点会导致概率密度函数的平滑程度大幅下降，这也和样本与估计点数目差距（16 与 1600）的差距有关。随着采样点数的增加，概率密度函数也趋于真实分布，变得更加光滑。而随着 K 值的增加，概率密度函数先是更加逼近真实的概率分布，然后概率密度的差异性被消去，先是各密度峰衰退、然后逐渐逼近均匀分布密度函数，这是因为采样区域的增加，“盒子”逐渐包括整个样本空间，而在盒子内概率密度相同的假设下，样本空间也是趋于均匀分布。
- 从核函数（矩形窗）可以看出较少的样本点会导致概率密度函数的平滑程度大幅下降，显示出窗函数原本的形状（矩形），这也和样本与估计点数目差距（16 与 1600）的差距有关。随着采样点数的增加，概率密度函数也趋于真实分布，变得更加光滑。随着代表窗口宽度的 h 值的增加，概率密度函数先是更加逼近真实的概率分布，然后概率密度的差异性被消去，先是各密度峰衰退、然后逐渐逼近均匀分布密度函数，这是因为采样区域的增加，“超球体”逐渐包括整个样本空间，而在矩形窗内部核函数恒为 1 的假设下，样本空间也是趋于均匀分布。
- 从核函数（高斯窗）可以看出较少的样本点会导致概率密度函数的平滑程度大幅下降，但是其光滑程度是三种方法中最好的，这是因为其显示出窗函数原本的形状（高斯曲线）本身就是光滑的，这也和样本与估计点数目差距（16 与 1600）的差距有关。随着采样点数的增加，概率密度函数也趋于真实分布，变得更加光滑。随着代表窗口宽度的 h 值的增加，概率密度函数先是更加

逼近真实的概率分布，然后概率密度的差异性被消去，先是各密度峰衰退、然后逐渐逼近样本的均值为期望的高斯函数，这是因为采样区域的增加，“超球体”逐渐包括整个样本空间，而在窗口为高斯分布的假设下，样本空间也是趋于高斯分布。

（4） 代码测试步骤：

- 代码结构：

```
Code
|  dataset.py
|  estimate.py
|  main.py
|  plot.py
|  utils.py
|  requirements.txt
|
└──data
      data_1000.npy
      data_16.npy
      data_2000.npy
      data_256.npy
```

- 相关函数详细注释在 main.py 中
- 测试环境配置：`conda create -n venv python=3.11 && conda activate venv &&  pip install -r requirements.txt`