实验环境

PyCharm-162.1237.1

TextMate ， python 2,7

一． 实验题目

<1>.考虑对于上述表格中的数据进行 Parzen 窗估计和设计分类器。窗函数为一个球形的高斯函数，如下所示:

$$\Phi((x - \quad)/h) \propto \exp\ [-(\quad - \quad)(x - \quad)/(2h\ 2)]$$

编写程序，使用 Parzon 窗估计方法对一个任意的测试样本点 x，进行分类。对分类器的训练则使用表格中的三维数据。同时另 h=1，分类样本点为:(0.5,1.0,0.0），(0.31,1.51, - 0.50），(- 0.3,0.44, - 0.1)

现在我们另 h=0.1，重复 a。

<2>.考虑不同维数空间中，使用 k-紧邻概率密度估计方法的效果

编写程序，对于一维的情况，当有 n 个数据样本点时，进行 k-近似概率密度估计。对表格中的类别 3 中的特征 1，用程序画出当 k=1,3,5 时的概率密度估计结果编写程序，对于二维的情况，当有 n 个数据样本点时，进行 k-近似概率密度估计。对表格中的类别 2 中的特征(1, 2），用程序画出当 k=1,3,5 时的概率密度估计结果对表格中的 3 个类别的三维特征，使用 k-紧邻概率密度估计方法。并且对下列点出的概率密度进行估计:(- 0.41,0.82,0.88），(0.14,0.72,4.1），(- 0.81,0.61, - 0.38)

二． 实验数据

	W1			W2			W3		
样本	X1	X2	X3	X1	X2	X3	X1	X2	X3
1	0.28	1.31	-6.2	0.011	1.03	-0.21	1.36	2.17	0.14
2	0.07	0.58	-0.78	1.27	1.28	0.08	1.41	1.45	-0.38
3	1.54	2.01	-1.63	0.13	3.12	0.16	1.22	0.99	0.69
4	-0.44	1.18	-4.32	-0.21	1.23	-0.11	2.46	2.19	1.31
5	-0.81	0.21	5.73	-2.18	1.39	-0.19	0.68	0.79	0.87
6	1.52	3.16	2.77	0.34	1.96	-0.16	2.51	3.22	1.35
7	2.20	2.42	-0.19	-1.38	0.94	0.45	0.60	2.44	0.92
8	0.91	1.94	6.21	-0.12	0.82	0.17	0.64	0.13	0.97
9	0.65	1.93	4.38	-1.44	2.31	0.14	0.85	0.58	0.99
10	-0.26	0.82	-0.96	0.26	1.94	0.08	0.66	0.51	0.88

三. 实验过程

（1） 第一个问题，需要用到的是 Parzon 窗估计以及设计分类器，题中给出了窗函数，我们把这个函数代入书上的公式，对得到的 p 进行比较可以获得分类结果
公式如下：

$$p_n(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h_n}\varphi(\frac{x-x_i}{h_n})$$

当 h=1 时得出结果：

```
slf:py_code slf$ python machine_4_2.py
[[ 0.12591898  0.4710785   0.39801295]
 [ 0.15343855  0.48279633  0.22602246]
 [ 0.13990812  0.37825433  0.18231173]]
```

这个结果的每一行分别是代表这三个测试点相对于$\omega 1$,□$\omega 2$ 以及$\omega 3$ 所计 算出来的$pn(x)$的结果。所以我们可以很明显的发现，这三个测试点到最后都应该被划分为第二类。

当 h=2 时得出结果：

```
slf:py_code slf$ python machine_4_2.py
[[  8.77469531e-20   6.76901953e-05   8.00018885e-17]
 [  2.87111635e-20   1.20018382e-05   2.15932897e-25]
 [  3.63712276e-12   3.78309404e-04   1.06402608e-39]]
```

这个结果的每一行分别是代表这三个测试点相对于$\omega 1$,□$\omega 2$ 以及$\omega 3$ 所计 算出来的$pn(x)$的结果。所以我们同样可任意发现，这三个测试点到最后都应该被划分为第二类。

（2）第二题，我们需要使用 k-近邻估计方法，进行分类，并且绘制出概率密度估计结果。首先，对于 k-近邻估计方法，其基本公式如下:

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

其中，kn就是我们 k-近邻估计中的 k，n 为我们的样本集的大小，而Vn为对应的面积或者体积。具体的算法过程如下:

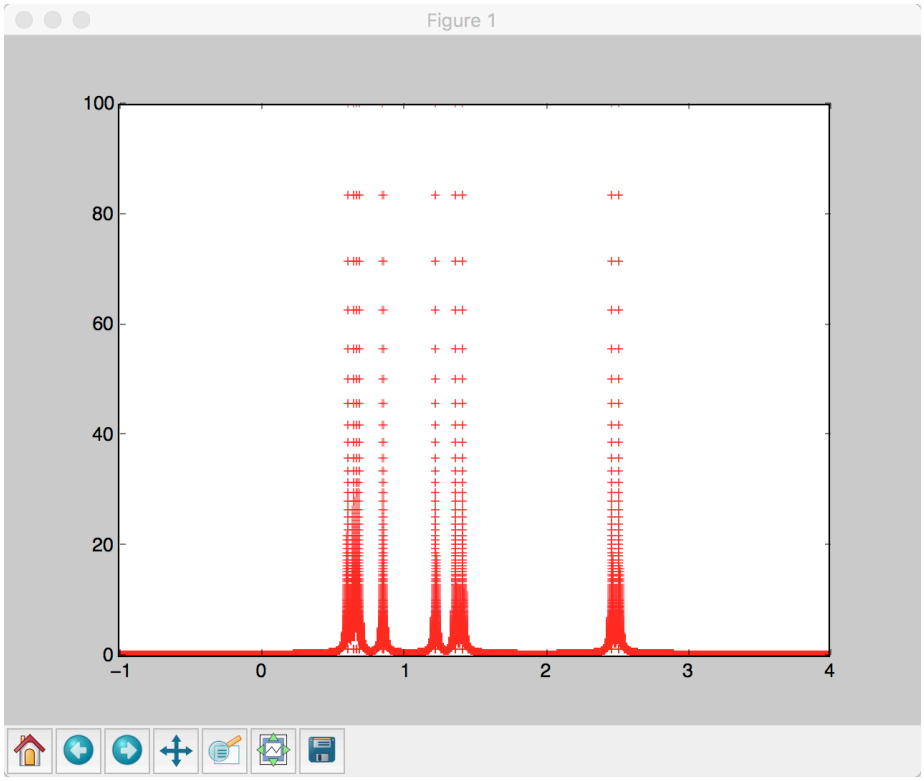
1. 为了估计点 x 的$pn(x)$，我们首先以点 x 为中心进行扩张，直到包含 进kn个样本为止。其中这kn个点被称为 x 的最近邻.
2. 此时，我们的到半径的长度。 而Vn则是一个我们自己定义的，与kn有关的一个空间函数，在二位情况下为面积，三位情况下为

体积。

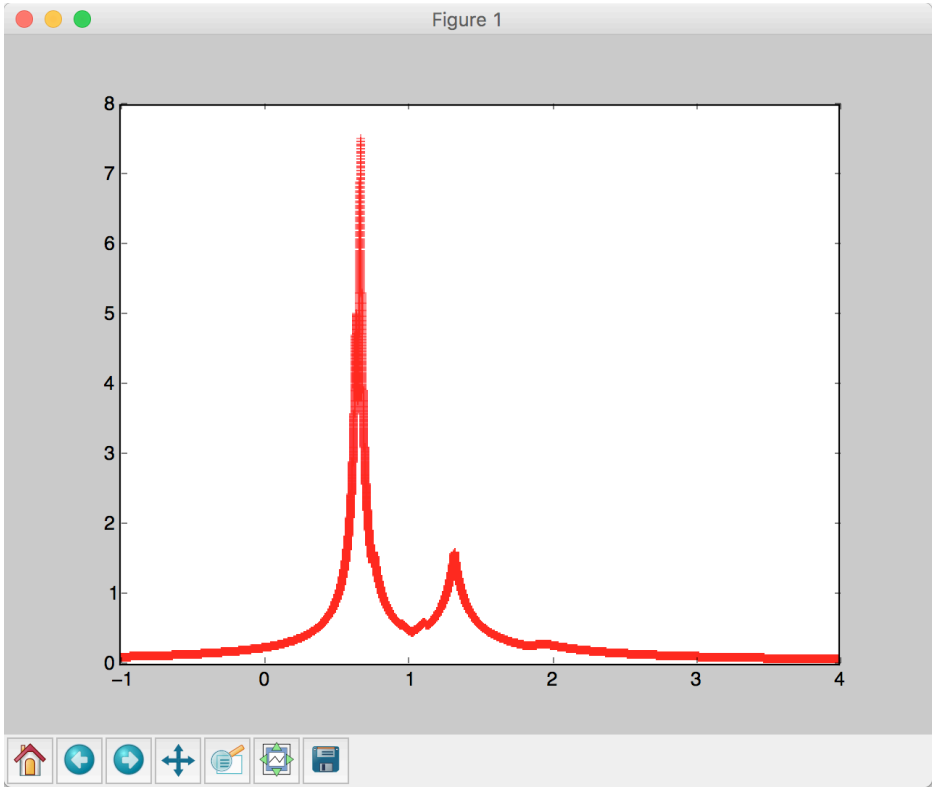
3. 带入公式，计算对应的$pn(x)$。

a） 第一问

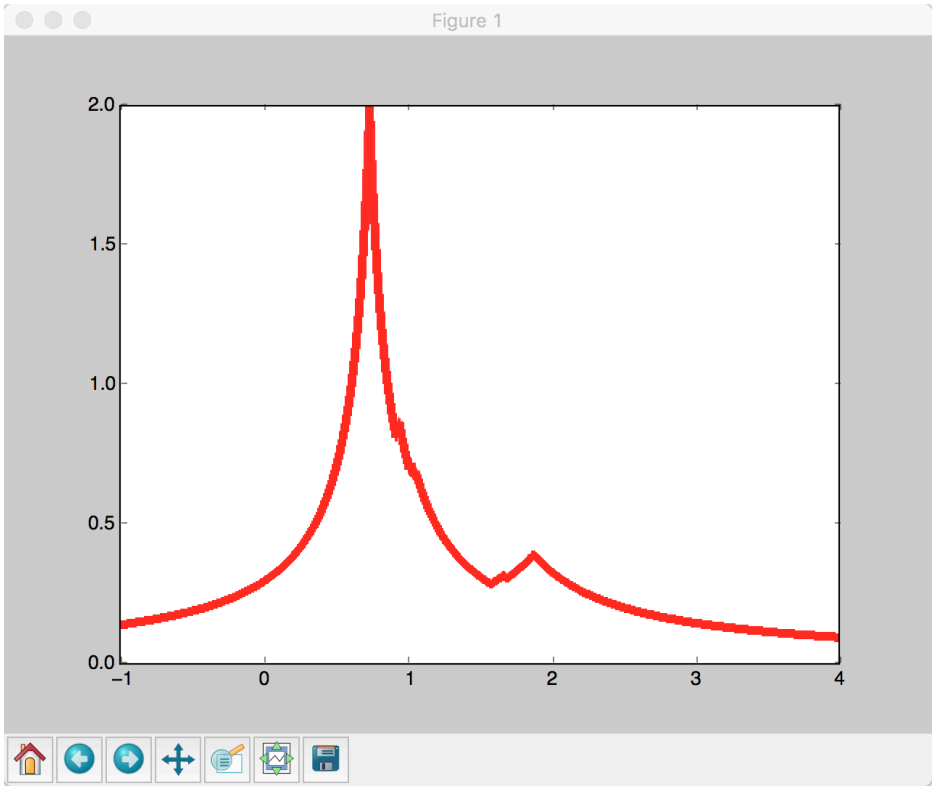
① 当 k=1 时



② 当 k=3 时

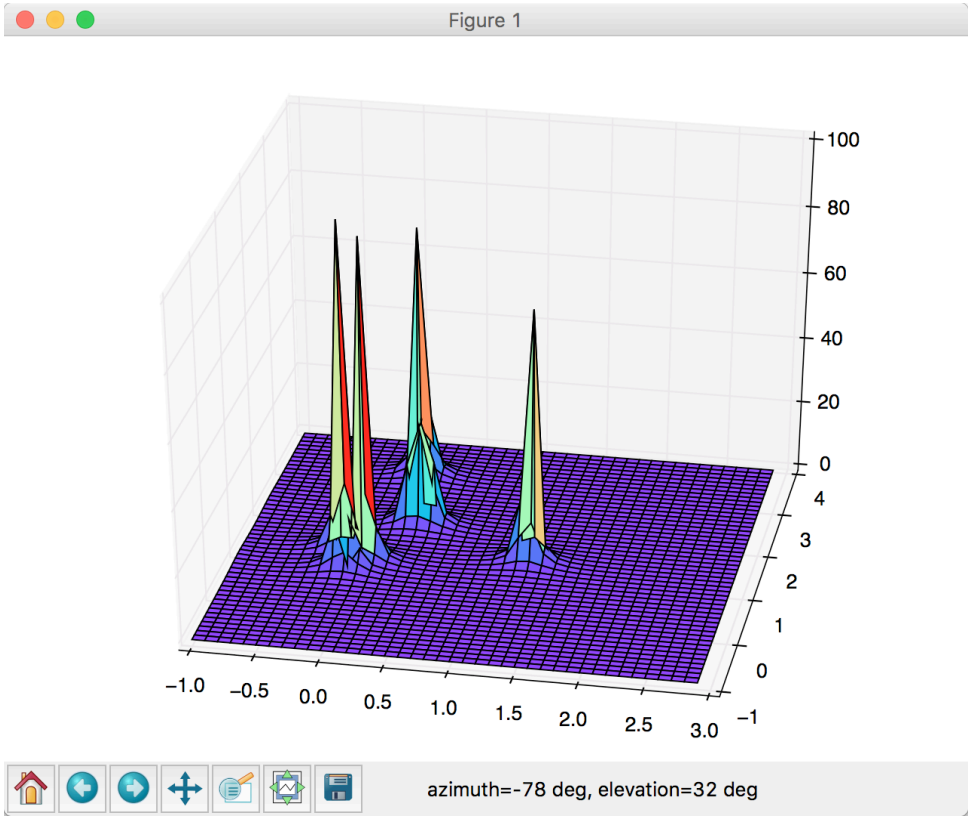


③ 当 k=5 时



b） 第二问

① 当 k=1 时



② 当 k=3 时

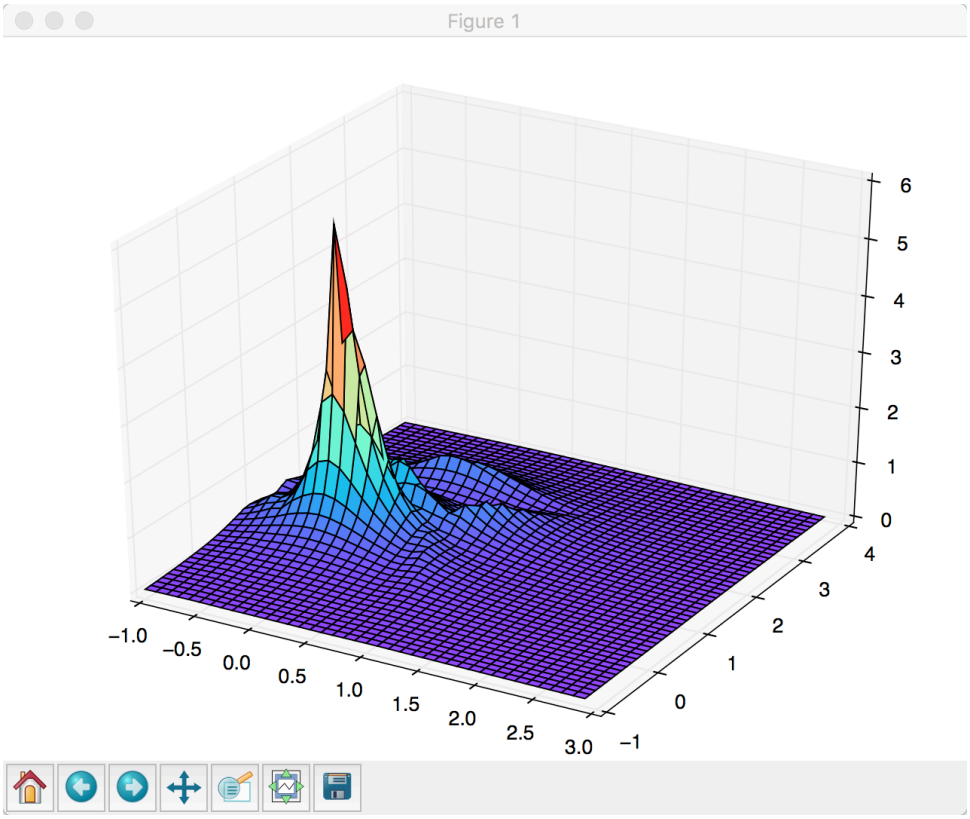


Figure 1
6
5
4
3
2
1
0
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0
−1
0
1
2
3
4

④ 当 k=5 时

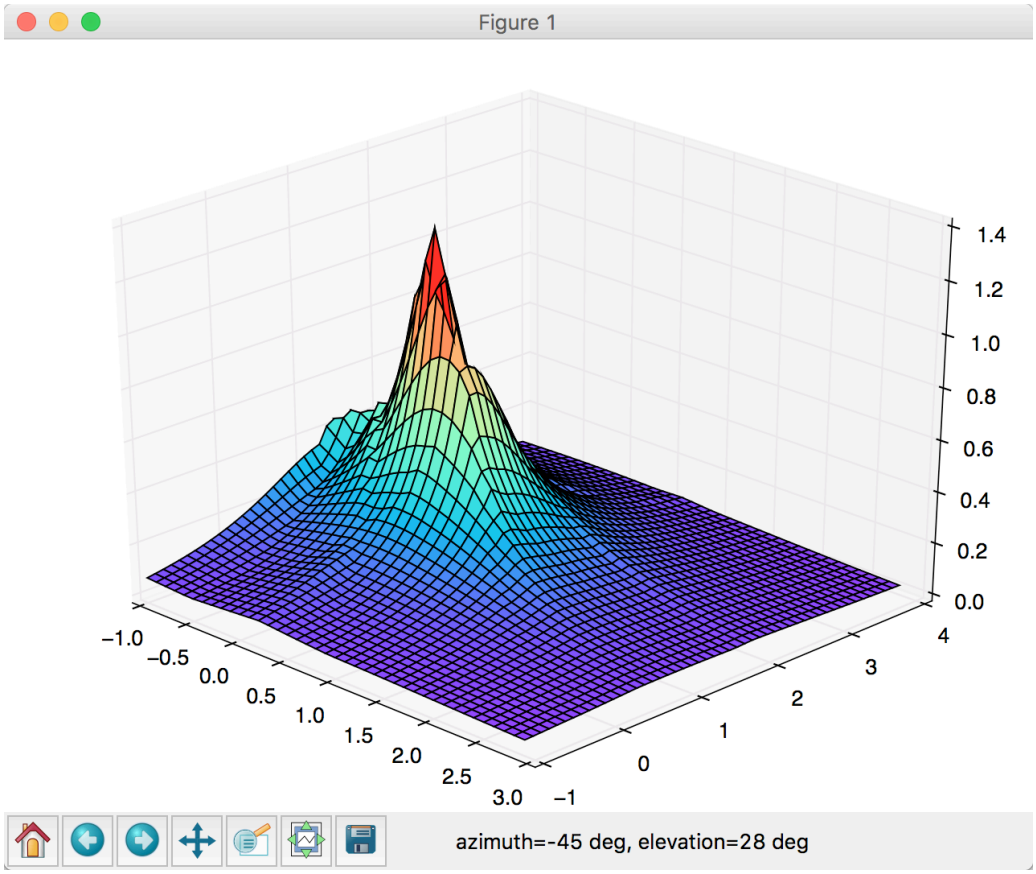


Figure 1
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
3.0
−1
0
1
2
3
4
azimuth=-45 deg, elevation=28 deg

c）第三问

第三问如果作图要四维了，应该没法输出图吧，最后的结果是输 p

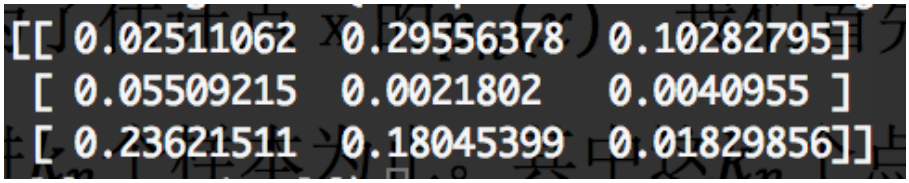
① k=1 时

这个结果的每一行分别是代表这三个测试点相对于$\omega 1$,□$\omega 2$ 以及$\omega 3$ 所计 算出来的$pn(x)$的结果。所以我们可以发现，

测试点一：第二类

测试点二：第一类

测试点三：第一类



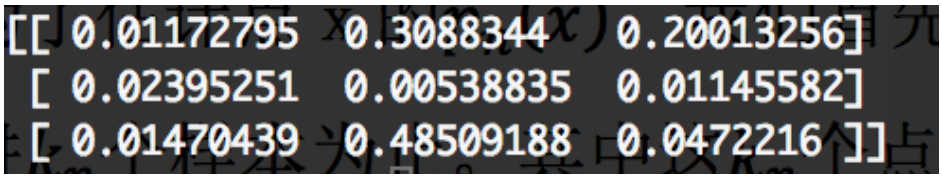
② k=3 时

这个结果的每一行分别是代表这三个测试点相对于$\omega 1$,□$\omega 2$ 以及$\omega 3$ 所计 算出来的$pn(x)$的结果。所以我们可以发现，

测试点一：第二类

测试点二：第一类

测试点三：第二类



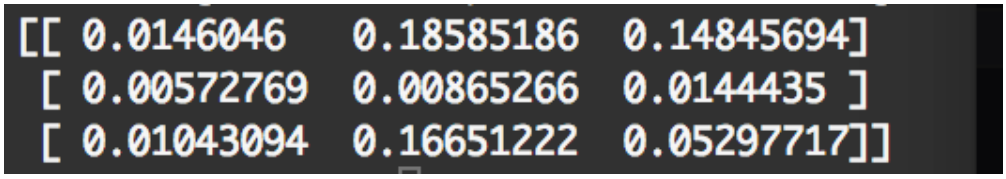
③ K=5 时

这个结果的每一行分别是代表这三个测试点相对于$\omega 1$,□$\omega 2$ 以及$\omega 3$ 所计 算出来的$pn(x)$的结果。所以我们可以发现，

测试点一：第二类

测试点二：第三类

测试点三：第二类



四． 遇到的问题

a)遇到了用 pycharm 画三维图像的问题，总是报 ImportError: No module named 这样的错，弄了一个下午最后没想到把程序在命令行中运行就好了，而且运行效率似乎也比 pycharm 快。可能的原因是电脑上 python 版本太

多了 pycharm 没有配好正确的路径

五. 收获与感悟

a) 经过这次实验首先我更加熟悉了 python 的使用

b) 我了解 Parzon 算法的使用

c) 我了解 K-邻近算法的使用

六. 代码

位于同项目路径的文件中