机器学习 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号： | 姓名： | | 班级： |
| 实验题目： | | | |
| 实验学时：2 | | 实验日期： | |
| 实验目的：  parzen窗方法和k近邻方法估计概率密度 | | | |
| 硬件环境：  Intel Core（TM） i5-4210U  RAM 8G  64位操作系统，基于x64处理器 | | | |
| 软件环境：  Win 10 64bit  MATLAB | | | |
| 实验步骤与内容：  1）4.3  A）编写函数  function r=lab3\_4\_3\_parzen(w,h,x,classes)  %w 矩阵  %h 边长  %x 待测试数据  %classes 类别  r=zeros(1,classes);  for i=1:classes  selected = w(:,4)==i;  w1 = w(selected,1:3);  rows=size(w1,1);  for j=1:rows  hn=h/sqrt(j);  r(i)=r(i)+exp(-(x-w1(j,:))\*(x-w1(j,:))'/(2\*power(hn,2)))/(sqrt(hn));  end  r(i)=r(i)/rows;  end  B）加载数据  %load lab3\_data.xlsx as lab3data  x = [[0.5,1.0,0]  [0.31,1.51,-0.5]  [-0.3,0.44,-0.1]];  h=0.1;  test\_size = 3;  for i=1:test\_size  r=lab3\_4\_3\_parzen(lab3data,h,x(i,:),test\_size);  result=find(r==max(r));  disp(['result of test',num2str(i),':',num2str(result)]);  disp(['p(x):',num2str(r)]);  end  C）计算结果  lab3\_4\_3  result of test1:2  p(x):0.045283 0.23087 0.12116  result of test2:2  p(x):0.046211 0.24209 0.075891  result of test3:2  p(x):0.066167 0.18673 0.0096394  D）将h变更为0.1  lab3\_4\_3  result of test1:2  p(x):2.8955e-43 2.1399e-07 1.9449e-47  result of test2:2  p(x):1.2786e-44 5.3629e-10 1.6078e-43  result of test3:2  p(x):4.9748e-28 3.7852e-11 2.6433e-127  结果未变，仍全部被分为第二类  2）4.4  首先加载数据  %load lab3\_data.xlsx as lab3data  selected = lab3data(:,4)==2;  w2 = lab3data(selected,1:2);  selected = lab3data(:,4)==3;  w3 = lab3data(selected,1);    lab3\_4\_4\_a(w3);  %lab3\_4\_4\_b(w2);  %lab3\_4\_4\_c(lab3data);   1. 编写程序，对于一维的情况，当有n个数据样本点时，进行k近邻概率密度估计。对于类别w3中的特征x1，画出k=1，3，5时的概率密度估计结果   函数1（之前有乱码，这里存入记事本后再次粘贴）：  function lab3\_4\_4\_a(w3)  %第一问，w3,x1,n个样本点  [rows,~]=size(w3);  p=zeros(1,300);  %生成n(n=300)个测试样本点  x=(0.01:0.01:3);  distance=zeros(size(p,2),rows);  for i=1:1:size(p,2)  for j=1:rows  %距离  distance(i,j)=abs(x(i)-w3(j,1));  end  end  dist\_sort=sort(distance,2); %按行排序  k=[1,2,5];  for index=1:size(k,2)  for i=1:size(x,2)  %eps:MATLAB中的最小正数（一维时必然会出现距离为0）  %p(x)=k/n/V  %体积V=2\*length  p(i)=k(index)/rows/(2\*dist\_sort(i,k(index))+eps);  end  subplot(1,size(k,2),index);  plot(x,p);  end  end  生成图：  lab3-1.jpg  k从1到5，曲线尖峰减少，曲线变平滑   1. 函数2则处理w2，使用前两维数据   function lab3\_4\_4\_b(w2)  %第二问，w2,[x1,x2],n个样本点  [rows,~]=size(w2);  p=zeros(100,100);  %生成n(n=100\*100)个测试样本点  x=(-2.95:0.05:2);  y=(0.04:0.04:4);  distance=zeros(size(p,1),size(p,1),rows);  for i=1:1:size(p,1)  for j=1:size(p,1)  for k=1:rows  %距离 sqrt[(x-x1)2+(y-y1)2]  distance(i,j,k)=sqrt((x(i)-w2(k,1))^2+(y(j)-w2(k,2))^2);  end  end  end  k=[1,2,5];  for index=1:size(k,2)  for i=1:size(x,2)  for j=1:size(y,2)  dist\_sort=sort(distance(i,j,:)); %升序  %eps:MATLAB中的最小正数  %p(x)=k/n/V  %体积V=pi\*r2  p(i,j)=k(index)/rows/(pi\*dist\_sort(k(index))^2);  end  end  subplot(1,3,index);  mesh(x,y,p);  end  end  概率密度图：  lab3-2.jpg  这里使用了欧几里得距离，结论与之前相似   1. 函数3：   function [ output\_args ] = lab3\_4\_4\_c( w )  x=[-0.41,0.82,0.88;  0.14,0.72,4.1;  -0.81,0.61,-0.38];  distance=zeros(3,10,3);  for i=1:3  for j=1:3  selected = w(:,4)==j;  w3 = w(selected,1:3);  for k=1:10  %三维空间求距离  distance(j,k,i)=sqrt((x(i,1)-w3(k,1))^2+(x(i,2)-w3(k,2))^2+(x(i,3)-w3(k,3))^2);  end  end  end  dist\_sort=sort(distance,2);  k=3;  p=zeros(3,3);  for i=1:3  for j=1:3  %球体体积  p(i,j)=k/10/(4\*3.14\*dist\_sort(j,k,i)^3/3);  end  end  for i=1:3  display([' [',num2str(x(i,:)),'] ：',num2str(p(i,:))]);  end  end  计算后得出结果为：  [-0.41 0.82 0.88] ：0.0021009 0.055325 0.035852  [0.14 0.72 4.1] ：0.0042909 0.00096527 0.0020522  [-0.81 0.61 -0.38] ：0.0026341 0.086899 0.0084593  可以看出，三个样本可根据概率大小依次被判别为第二类、第一类和第二类。 | | | |
| 结论分析与体会：  通过这次实验，我对最大似然估计有了更加详尽的认知，了解了KNN的基本原理，学会了在不同分布情况下针对不同未知数据求已知数据的方法公式，并了解了其推算过程。可以看出，parzen窗需要先确定条件概率密度后才能对测试点进行分类，而k近邻可以直接根据k个近邻中哪个类别的样本点最多就将其分为哪一类，k近邻模型几乎没有进行什么学习，来一个测试点都要重新计算。之后的实验也将继续使用MATLAB。 | | | |