山东大学 软件 学院

机器学习 课程实验报告

|  |  |  |
| --- | --- | --- |
| 学号：201800301153 | 姓名： 傅显坤 | 班级： 一班 |
| 实验日期： 2020.10.07 | | |
| 实验题目：    判别函数：    考虑上机题2中的3个类别，设P（wi）=1/3.  (a)以下各测试点与上机练习2中的各类别均值间的Mahalanobis距离分别是多少：（1，2，1）t，（5，3，2）t，（0，0，0）t，（1，0，0）t  (b)对以上各点进行分类  (c)若设P（w1）=0.8，P（w2）=P（w3）=0.1再对以上测试点进行分类 | | |
| 软件环境：  MacOS Catalina  Python3.0  IDE：PyCharm | | |
| 实验步骤与内容：  1.输入样本数据：以矩阵的形式储存    2.计算马氏距离  公式：    先由函数get\_u(w)计算出均值u  再根据函数get\_sigma(w)计算出协方差矩阵  调用python的numpy中的矩阵转置函数和取逆矩阵函数获得转置矩阵和逆矩阵  求得测试点到三个类别均值之间的马氏距离。  3.通过先验概率和判别函数对测试点进行分类  判别函数公式  第一项可以看作马氏距离的平方\*（-1/2），第二项中d是数据的维数，  第三项中的行列式调用numpy的库函数求得，第四项中为先验概率的对数。  计算出判别函数的值，通过比较测试点在不同类别的判断函数的值，哪个函数值最大就属于哪个类别。 | | |
| 实验结果：  1.计算马氏距离：    2.分类 | | |
| 部分重要代码：  1. Python代码计算协方差矩阵：  def get\_sigma(w):  row = w.shape[0] # 获取第一维度的数目（行）  col = w.shape[1] # 获取第二维度的数目（列）  ls\_average = []  for i in range(col):  sum = 0  for j in range(row):  sum += w[j][i]  ls\_average.append(sum / row)  ls\_u = []  ls\_u.append(ls\_average)   for i in range(col):  for j in range(row):  # print(str(ls\_u[0][i]))  w[j][i] = w[j][i] - ls\_u[0][i]   sigma = 1 / (row - 1)  sigma = sigma \* numpy.dot(w.T, w)  return sigma  2. Python代码实现计算马氏距离  def get\_mahalanobis\_distance(x, w):  # sigma = numpy.cov(w.T) # 求协方差矩阵 库函数  temp = w.copy()  sigma = get\_sigma(temp) # 求协方差矩阵 自己实现  u = get\_u(w)  sigma\_inverse = numpy.linalg.inv(sigma) # 矩阵求逆  tp = x - u  return numpy.sqrt(numpy.dot(numpy.dot(tp.T, sigma\_inverse), tp))[0][0] # 矩阵相乘  3. Python代码实现计算判别函数的值：  # 判别函数 def discriminant\_function(x, w, p):  temp = w.copy()  sigma = get\_sigma(temp)  u = get\_u(w)  sigma\_inverse = numpy.linalg.inv(sigma)  tp = x - u  # 返回判别函数的值 d是维数  return -0.5 \* numpy.dot(numpy.dot(tp.T, sigma\_inverse), tp) - u.shape[0] / 2.0 \* numpy.log(  2 \* math.pi) - 0.5 \* numpy.log(abs(numpy.linalg.det(sigma))) + numpy.log(p)  4.分类函数  # 根据判别函数分类 def classification(x, w, p):  print("当先验分别为：")  for p\_index in p:  print(p\_index)   classifications = []  for i in x:  class\_number = 0  max\_g = -sys.maxsize-1 # 最小的数  ls = []  for j in range(len(w)):  g = discriminant\_function(i, w[j], p[j])  if g > max\_g:  class\_number = j + 1 # 判定所属类别  max\_g = g  ls.append(str(i.T[0]))  ls.append(class\_number)  classifications.append(ls)  return classifications | | |