Lab5　kNN 应用实践

实验简介：利用 Python 实现 kNN 分类器。内容包括导入数据，归一化数据，距离计算，实现 kNN 分类器，并应用 kNN 分类器改进约会网站，以及实现手写数字体的识别。

作业要求：见 QQ 群文件："作业要求.pdf"

1、实现 kNN 分类器

在构建完整 kNN 分类器之前，需要编写一些基本的通用函数，包含在 knn1.py 中，导入此模块。

```
>>> import knn1
# 函数 createDataSet()创建一个简单数据集合和标签：
测试函数功能：创建变量 group 和 labels
>>> group, labels = knn.createDataSet()
# 查看变量 group 和 labels 的值：
>>> group
>>> labels
# 函数 classify()实现 kNN 分类器
  测试分类器功能：
>>> knn.classify([0,0], group, labels, 3)
```

输出结果是 B，可以改变输入[0, 0]为其它值，继续测试运行结果。

2、利用 kNN 分类器改进约会网站的配对效果

利用收集的在线约会网站的约会数据，将约会网站推荐的匹配对象归入适当的类别（不喜欢的人，魅力一般的人，极具魅力的人）。

（1）准备数据

收集的数据存放在文本文件"datingTestSet.txt"中，每条数据占一行，总共 1000 行，主要包括 3 个特征：每年获得的飞行里程数，玩游戏视频所耗时间百分比，每周消费的冰激凌公升数。在特征数据输入分类器之前，需要将待处理数据的格式转换为分类器可以接受的格式。

函数 file2matrix()解决格式输入问题，输入参数为文件名字符串，输出为训练样本矩阵和类标签向量。

```
>>> import knn2
>>> datingDataMat, datingLabels = knn2.file2matrix('datingTestSet2.txt')
>>> print(datingDataMat)
>>> print(datingLabels[0:20])
```

 # 数据散点图

```
>>> import matplotlib.pyplot as plt
>>> %matplotlib inline
>>> plt.scatter(datingDataMat[:,1], datingDataMat[:,2])
```

>>> plt.scatter(datingDataMat[:,1], datingDataMat[:,2], 15.0*array(datingLabels), 15.0*array(datingLabels))

 # 数据归一化

在处理不同取值范围的特征值时，通常采用数值归一化方法将取值范围处理为 0 到 1 或-1 到 1 之间，将任意取值范围的特征值转化为 0 到 1 区间值的公式：

newValue = (oldValue-min)/(max-min)

其中 max 和 min 分别是数据集中的相应维度的最大特征值和最小特征值。

函数 autoNorm()将数字特征值转化为 0 到 1 的区间。

>>> normMat, ranges, minVals = knn2.autoNorm(datingDataMat)

>>> normMat

>>> ranges

>>> minVals

（2）测试分类器

函数 datingClassTest()测试分类器效果：

>>> knn2.datingClassTest()

（3）使用模型

给用户提供程序，通过该程序用户会在约会网站上找到某个人并输出它的信息。输入一条数据，kNN 分类器给出用户对某个人的喜欢程度预测值，函数 classifyPerson()完成此功能。

>>> knn2.classifyPerson()

3、利用 kNN 分类器识别手写体数字

实验所用到的实际图像存储在两个子目录中：目录“trainingDigits”中包含了大约 2000 个例子，命名规则如 9_45.txt，表示该文件的类别是 9，是数字 9 的第 45 个实例，每个数字大概有 200 个实例。目录“testDigits”中包含了大约 900 个测试例子。使用目录“trainingDigits”中的数据训练分类器，使用目录“testDigits”中的数据测试分类器的效果，两组数据没有重叠。

（1）准备数据

使用 kNN 分类器，首先将图像处理为一个向量。实验中，将把一个 32*32 的二进制图像矩阵转换成 1*1024 的向量，函数 img2vector()将图像转换为向量，该函数创建 1*1024 的 Numpy 数组，然后打开给定的文件，循环读出文件的前 32 行，并将每行的前 32 个字符值存储在 Numpy 数组中，最后返回数组。

（2）构建训练数据集

函数 trainingDataTest 利用目录“trainingDigits”中的文本数据构建训练集向量，以及对应的类别标签向量（标签向量可理解为对应的文件中数字的正确分类）。由于文件名的规律命名，函数 classnumCut()实现从文件名中解析分类数字，提供分类标签。

注意：程序开头写上 from os import listdir 语句，以导入 listdir 模块，它可以列出给定目录的文件名。

（3）测试模型

通过测试“testDigits”目录下的样本，计算准确率。

函数 handwrtingTest()实现分类器测试。每个数据文件中的数字按顺序展开成一个 1024 维的向量，而向量之间的距离用欧式距离。

切换至文件 knn3.py 所在目录，并在 cmd 窗口执行：

> python knn3.py

观察、分析程序的运行结果。

4、操作习题

（1）利用 Numpy 和 Python 更常用的函数对程序代码（knn1.py，knn2.py，knn3.py）进行优化（距离计算、排序等），并对代码进行规范，使代码看起来更加简洁。

（2）利用 sklearn 的 kNN 分类器改进约会网站的配对效果，打印模型的混淆矩阵，并对结果进行解释，绘制学习曲线和验证曲线，并对曲线进行分析。

（3）利用 sklearn 的 kNN 分类器别手写体数字，通过绘制累积可解释性方差贡献率曲线确定 PCA 的最佳参数 n_components，并比较应用 PCA 降维前后 kNN 分类器结果。