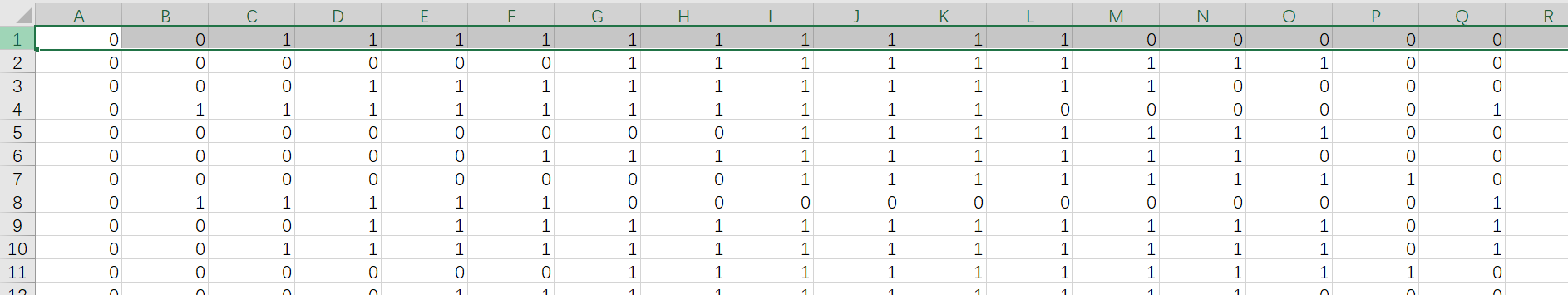
**问题描述：**使用python实现KNN分类器，识别手写数字。训练集的数据格式为266个0、1组成的向量，前256个是一个16\*16的矩阵，对应手写图像的像素点，有笔画的点为1，空白点为0.最后十个0、1，有且仅有一位是1，用来标记该数据的类别，如“0”的标记为10 0000 0000.测试集的格式相同。最终要计算KNN分类器在给定测试集上的错误率（识别结果与原标记不同即为错误）。

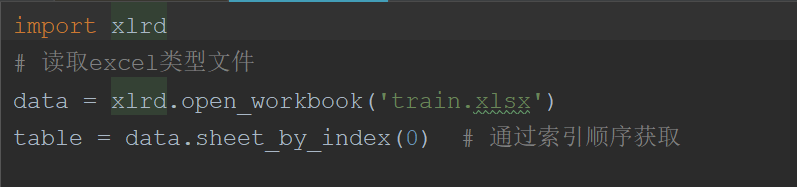
**解决方法：**

1. 处理训练集和测试集数据：

将数据保存为Excel表格，以“空格”为标志分列，将每条数据存在1行，266列

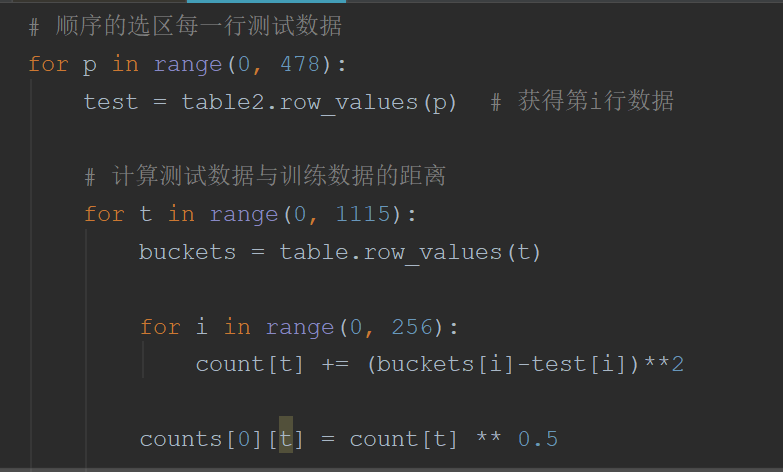


导入xlrd库读入Excel文件:

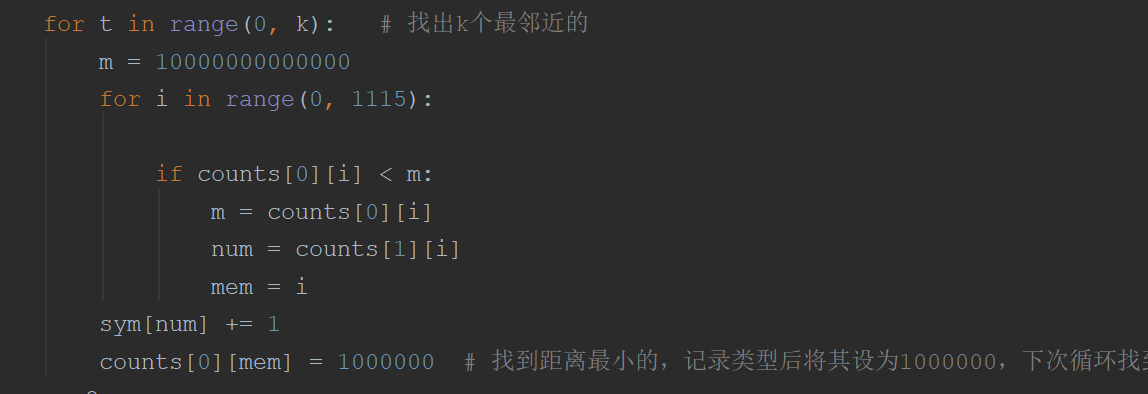


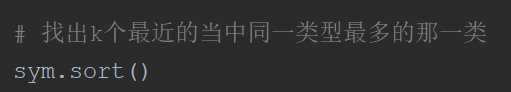
存储每条数据所用的数据结构：数组

1. 计算每条测试数据与所有训练数据之间的距离（向量相减，平方和开根号），“粗暴”的实现，循环嵌套。源代码中关键部分有注释。

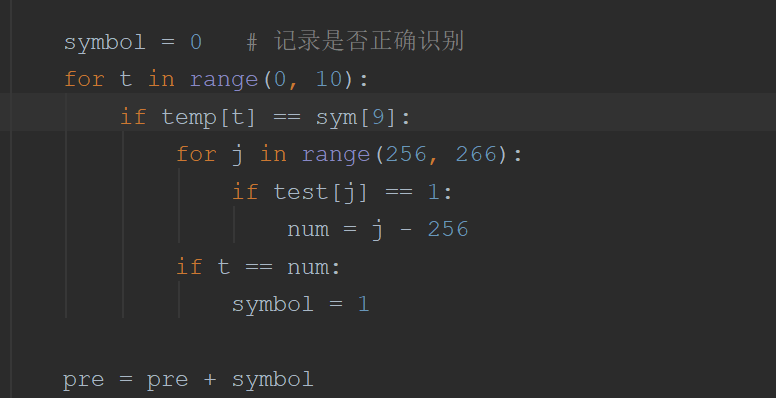


1. 找出与距离最近的K条数据，记录他们的类别0~9，选取数量最多的一个类别作为识别的结果。



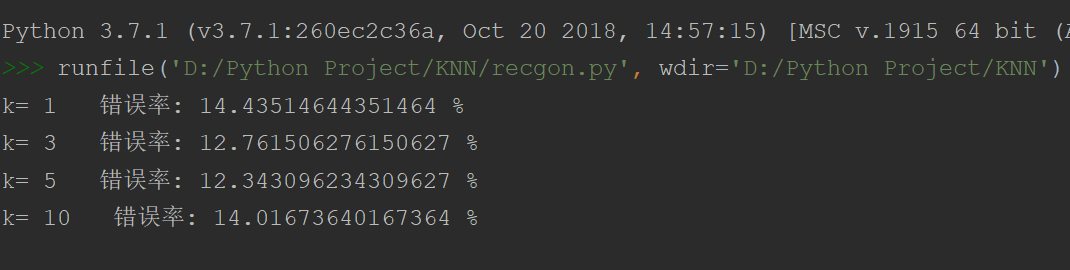


1. 检验是否识别正确、计算错误率。



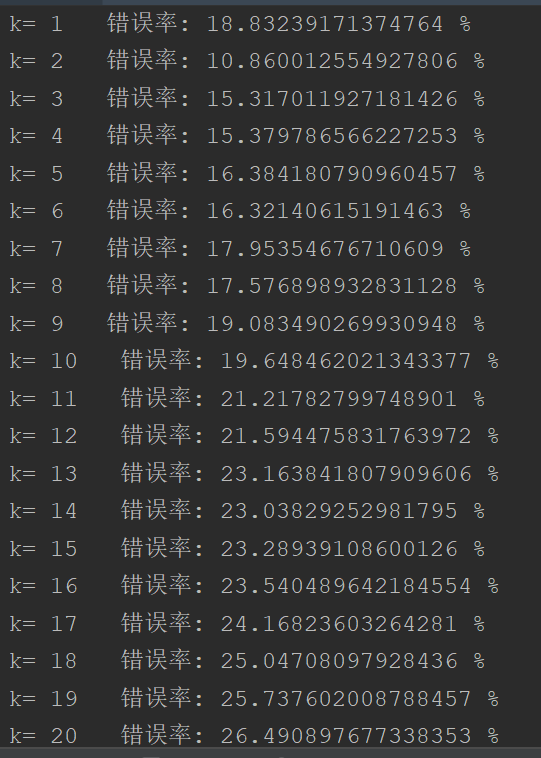


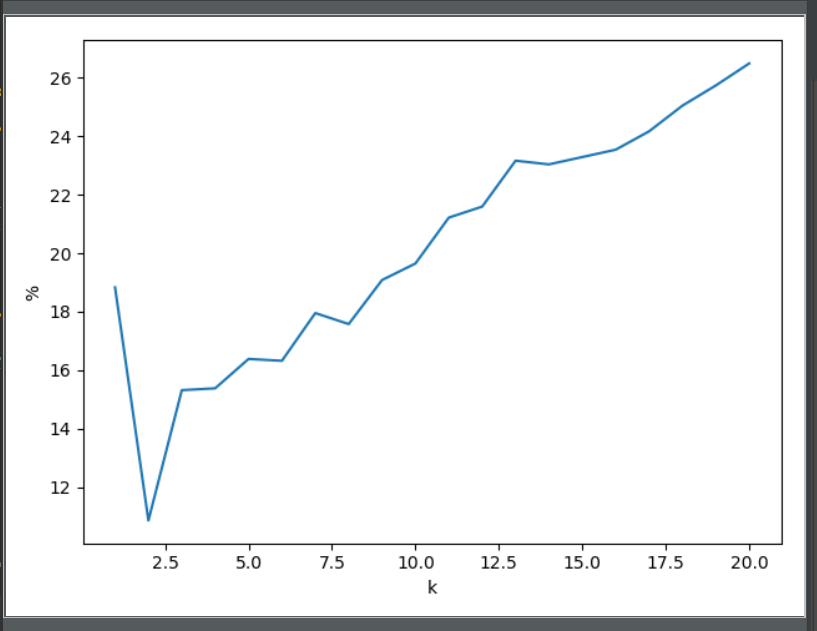
实验结果：



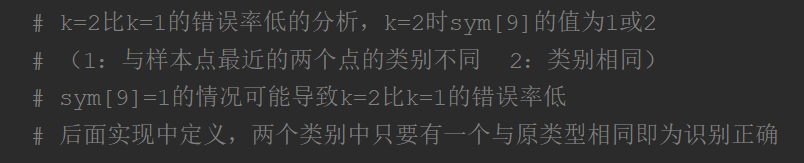
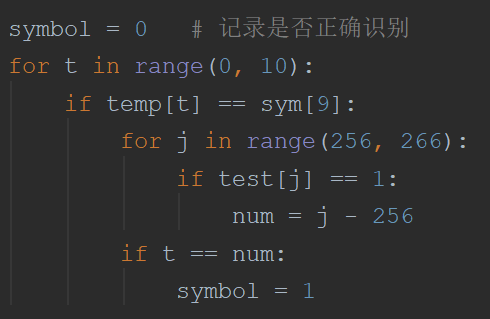
提高部分，采用留一法选取合适的K值，结果及图形化表示：

（使用第一次给出的网页上下载的训练集，共1593条，运行时间较长）





从以上结果可看出，在1~20的范围内，k选2时该分类器的错误率最低。

**实验分析：**

从实验的结果可以看出，并不是K的值越大准确率越高，可能存在“在K个最近的样本点中，某类样本占比最多，但却不是距离最近”的情况，由于K选的够大，将待测数据判定为占比较多的那一类型。因此，在实际应用中要根据训练集选取合适的K值，或者采取“根据距离赋予权重”等方法改进KNN分类器。

从算法角度来看，原始的KNN算法是一种“暴力”算法，每有一条测试数据就要遍历一遍训练集，实现上简单，效率上底下，还有很大改进的空间。