# 目录

[实验1：基于朴素贝叶斯实现过滤垃圾邮件 1](#_Toc471318283)

[一、实验目的 1](#_Toc471318284)

[二、实验原理 1](#_Toc471318285)

[三、实验过程 2](#_Toc471318286)

[四、实验结果 5](#_Toc471318287)

[五、实验总结 6](#_Toc471318288)

[实验2：k-近邻算法实现手写识别系统 7](#_Toc471318289)

[一、实验目的 7](#_Toc471318290)

[二、实验原理 7](#_Toc471318291)

[三、实验过程 8](#_Toc471318292)

[四、实验结果 11](#_Toc471318293)

[五、实验总结 12](#_Toc471318294)

# 实验1：基于朴素贝叶斯实现过滤垃圾邮件

## 一、实验目的

1. 使用概率分布进行分类。

2. 学习朴素贝叶斯分类器，构建另一个分类器，观察其在真实的垃圾邮件数据集中的过滤效果。

## 二、实验原理

1. 贝叶斯决策理论

贝叶斯，这里使用的概率解释属于贝叶斯概率理论的范畴，该理论非常流行且效果良好。贝叶斯概率以18世纪的一位神学家托马斯.贝叶斯（Thomas Bayes）的名字命名。贝叶斯概率引入先验知识和逻辑推理来处理不确定命题。另一种概率解释称为频数概率（frequency probability）,它只从数据本身获得数据，并不考虑逻辑推理及先验知识。

贝叶斯决策理论是主观贝叶斯派归纳理论的重要组成部分。贝叶斯决策就是在不完全情报下，对部分未知的状态用主观概率估计，然后用贝叶斯公式对发生概率进行修正，最后再利用期望值和修正概率做出最优决策。其基本思想是：

（1）已知类条件概率密度参数表达式和先验概率。

（2）利用贝叶斯公式转换成后验概率。

（3）根据后验概率大小进行决策分类。

贝叶斯公式：

　　　　 　　 P(B[j]|A[i])=P(A[i]|B[j])P(B[j]) / P(A[i])

朴素贝叶斯是基于一个简单假设所建立的一种贝叶斯方法，朴素贝叶斯假定样本的不同特征属性对样本的归类影响是相互独立的。此时若样本A中同时出现特征A[i]与A[k],则样本A属于类别B[j]的概率为：

　　　　　　P(B[j]|A) = P(B[j]|A[i],A[k]) = P(B[j]|A[i])P(B[j]|A[k])

2. 基于贝叶斯决策理论的分类方法

使用朴素贝叶斯

优点：在数据较少的情况下仍然有效，可以处理多类别问题

缺点：对于输入数据的准备方式较为敏感

适用数据类型：标称型数据

假设现在我们有一个数据集，它由两类数据组成，数据分布如图1所示。

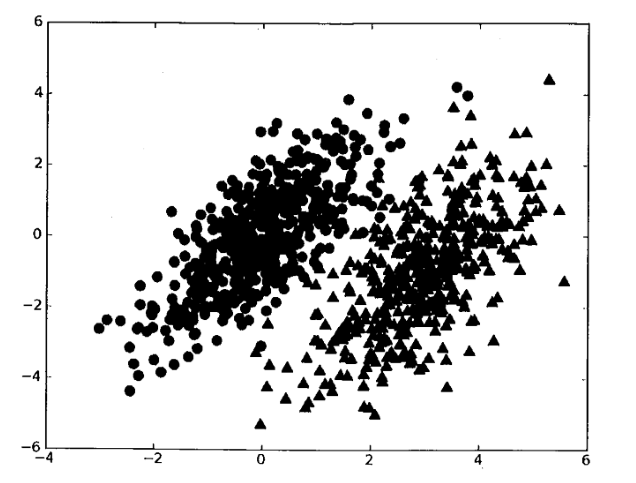


图1 两个参数已知的概率分布，参数决定了分布的形状

我们现在用p1（x，y）表示数据点（x，y）属于类别1（图中用圆点表示的类别）的概率，用p2（x，y）表示数据点（x，y）属于类别2（图中用三角形表示的类别）的概率，那么对于一个新数据点（x，y），可以用下面的规则来判断它的类别：

* 如果p1（x，y）> p2（x，y），那么类别为1
* 如果p2（x，y）> p1（x，y），那么类别为2

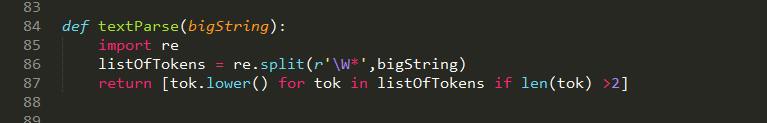
也就是说，我们会选择高概率对应的类别。这就是贝叶斯决策理论的核心思想，即选择具有高概率的决策。

## 三、实验过程

1. 收集数据：提供文本文件

使用RSS源，数据保存在email文件夹下，下面有两个文件夹ham和spam文件夹，两个文件夹下有50封邮件进行测试。

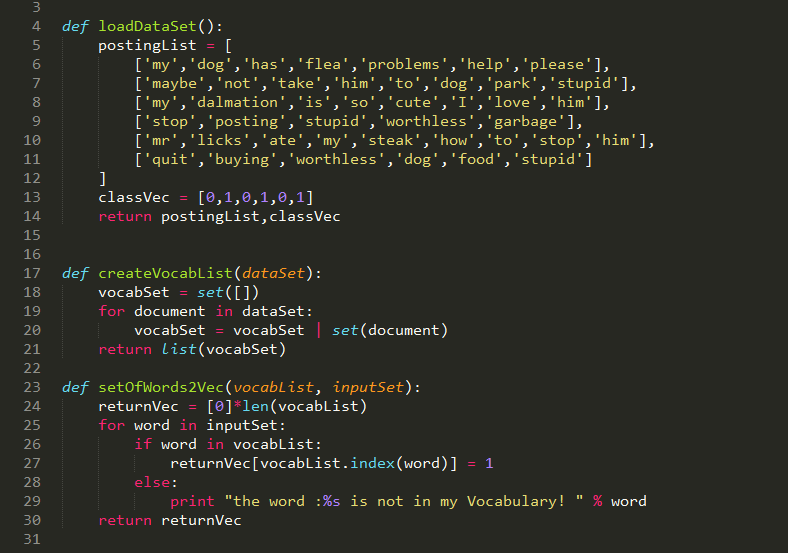
2. 准备数据：将文本文件解析成词条向量

这个方法是从邮件的文本中构建词向量。

该函数接受一个大字符串并将其解析为字符串列表。该函数去掉少于两个字符的字符串，并将所有字符串转换成小写。

3. 分析数据：检查词条确保解析的正确性

在bayes.py文件中有下列三个函数loadDataSet()，createVocabList()和setOfWords2Vec()



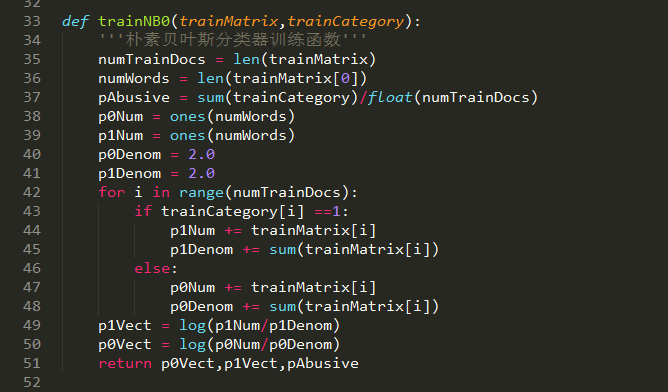
（1）第一个函数loadDataSet()创建了一些实验样本。该函数返回的第一个变量是进行词条切分后的文档集合。LoadDataSet()函数返回的第二个变量是一个类别标签的集合。这里有两类，侮辱性和非侮辱性。这些文本的类别由人工标注，这些标注信息用于训练程序以便自动检测侮辱性留言。

（2）createVocabList()会创建一个包含在所有文档中出现的不重复词的列表，为此使用了Python的set数据类型。此词条列表输给set构造函数，set就会返回一个不重复词表。

首先，创建一个空集合，然后将每篇文档返回的新词集合添加到该集合中。

（3）setOfWords2Vec()，该函数的输入参数为词汇表及某个文档，输出的是文档向量，向量的每一元素为1 或 0，分别表示词汇表中的单词在输入文档中是否出现。函数首先创建一个和词汇表中的单词，则将输出的文档向量中的对应值设为1。

4. 训练算法：使用建立的trainNB0（）函数



代码函数中的输入参数为文档矩阵trainMatrix，以及由每篇文档类别标签所构成的向量trainCategory。首先，计算文档属于侮辱性文档（class=1）的概率，即p(1)。因为这是一个二分类分类问题，所以可以通过1-p(1)得到p(0)。

上述程序中的分母变量是一个元素个数等于词汇表大小的NumPy数组。在for循环中，要遍历训练集trainMatrix中的所有文档。一旦某个词语（侮辱性或正常词语）在某一文档中出现，则该词对应的个数（p1Num或者p0Num）就加1，而且在所有的文档中，该文档的总次数也相应加1。

5. 测试算法：使用classifyNB()，并且构建一个新的测试函数来计算文档集的错误率



classifyNB函数有四个输入：要分类的向量vec2Classify以及使用trainNB0（）计算得到的三个概率。使用NumPy()的数组来计算两个向量相乘的结果。这里的相乘是指对应元素相乘，即先将两个向量中的第1个元素相乘，然后将第2个元素相乘，以此类推。接下来将词汇表中所有词的对应值相加，然后将该值加到类别的对数概率上。最后，比较类别的概率返回大概率对应的类别标签。

6. 使用算法：构建一个完整的程序对一组文档进行分类，将错分的文档输出到屏幕上

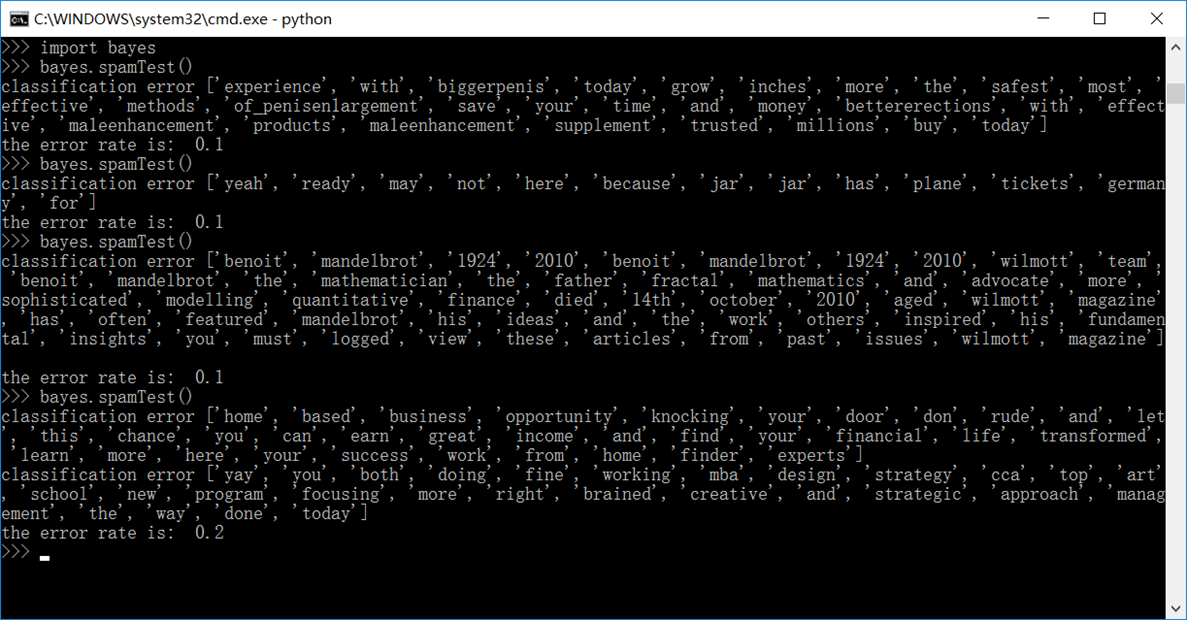


spamTest()对贝叶斯垃圾邮件分类器进行自动化处理。导入文件夹spam与ham下的文本文件，并将它们解析为词向表。接下来构建一个测试集与一个训练集，两个集合中的邮件都是随机选出的。本例中共有50封电子邮件，并不是很多，其中的10封电子邮件被随机选择为测试集。分类器所需要的概率计算只利用训练集中的文档来完成。Python变量trainingSet是一个整数列表，其中的值从0到49。接下来，随机选择其中10个文件。选择出的数字所对应的文档被添加到测试集，同时也将其从训练集中剔除。这种随机选择数据的一部分作为训练集，而剩余部分作为测试集的过程称为留存交叉验证。

接下来的for循环遍历训练集的所有文档，对每封邮件基于词汇表并使用setofWords2Vec()函数来构建词向量。这些词在trainNB0（）函数中用于计算分类所需的概率。然后遍历测试集，对其中每封电子邮件进行分类。如果邮件分类错误，则错误数加1，最后会出总的错误百分比。

## 四、实验结果

运行结果



函数spamTest()会输出在10封随机选择的电子邮件上的分类错误率。既然这些电子邮件是随机选择的，所以每次的输出结果可以有些差别。如果发现错误的话，函数会输出错分文档的词表，这样就可以了解到底是哪篇文档发生了错误。如果想要更好的估计错误率，那么就应该将上述过程重复多次，求平均值。

## 五、实验总结

实验运行过程中一直出现的错误是将垃圾邮件误判为正常邮件。相比之下，将垃圾邮件误判为正常邮件要比将正常邮件归到垃圾邮件好。为避免错误，有多种方式可以用来修正分类器。

对于分类而言，使用概率有时要比使用硬规则更为有效。贝叶斯概率及贝叶斯准则提供了一种利用已知值来估计未知概率的有效方法。

可以通过特征之间的条件独立性假设，降低对数据量的需求。独立性假设是指一个词的出现概率并不依赖于文档中的其他词。当然我们也知道这个假设过于简单。这就是之所以称为朴素贝叶斯的原因。尽管条件独立性假设并不正确，但是朴素贝叶斯仍然是一种有效的分类器。

利用现代编程语言来实现朴素贝叶斯时需要考虑很多实际因素。下溢出就是其中一个问题，它可以通过对概率取对数来解决。词袋模型在解决文档分类问题中比词集模型有所提高。还有其他一些方面的改进，比如说移除停用词，当然也可以话大量时间对切分器进行优化。

# 实验2：k-近邻算法实现手写识别系统

## 一、实验目的

1. 学习如何在系统上使用k-近邻算法。

2. 使用k-近邻算法改进手写数字识别系统。

## 二、实验原理

1. K最近邻(k-Nearest Neighbor，KNN)分类算法，是一个理论上比较成熟的方法，也是最简单的机器学习算法之一。该方法的思路是：如果一个样本在特征空间中的k个最相似(即特征空间中最邻近)的样本中的大多数属于某一个类别，则该样本也属于这个类别。

2. k-近邻算法（kNN），它的工作原理是：存在一个样本数据集合，也称作训练样本集，并且样本集中每个数据都存在标签，即我们知道样本集中每一数据与所属分类的对应关系。输入没有标签的新数据后，将新数据的每个特征与样本集中数据对应的特征进行比较，然后算法提取样本集中特征最相似数据（最邻近）的分类标签。一般来说，我们只选择样本数据集中前k个最相似的数据，这就是k-近邻算法中k的出处，通常k是不大于20的整数。最后，选择k个最相似数据中出现次数最多的分类，作为新数据的分类。

3. k-近邻算法概述

简单的说，k-近邻算法采用测量不同特征值之间的距离方法进行分类。

k-近邻算法

优点：精度高、对异常值不敏感，无数据输入假定。

缺点：计算复杂度高、空间复杂度高。

适用数据范围：数值型和标称性。

4. KNN算法中，所选择的邻居都是已经正确分类的对象。该方法在定类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。

KNN 算法本身简单有效，它是一种 lazy-learning 算法，分类器不需要使用训练集进行训练，训练时间复杂度为0。KNN 分类的计算复杂度和训练集中的文档数目成正比，也就是说，如果训练集中文档总数为 n，那么 KNN 的分类时间复杂度为O(n)。

KNN方法虽然从原理上也依赖于极限定理，但在类别决策时，只与极少量的相邻样本有关。由于KNN方法主要靠周围有限的邻近的样本，而不是靠判别类域的方法来确定所属类别的，因此对于类域的交叉或重叠较多的待分样本集来说，KNN方法较其他方法更为适合。

K 近邻算法使用的模型实际上对应于对特征空间的划分。K 值的选择，距离度量和分类决策规则是该算法的三个基本要素：

（1）K 值的选择会对算法的结果产生重大影响。K值较小意味着只有与输入实例较近的训练实例才会对预测结果起作用，但容易发生过拟合；如果 K 值较大，优点是可以减少学习的估计误差，但缺点是学习的近似误差增大，这时与输入实例较远的训练实例也会对预测起作用，使预测发生错误。在实际应用中，K 值一般选择一个较小的数值，通常采用交叉验证的方法来选择最优的 K 值。随着训练实例数目趋向于无穷和 K=1 时，误差率不会超过贝叶斯误差率的2倍，如果K也趋向于无穷，则误差率趋向于贝叶斯误差率。

（2）该算法中的分类决策规则往往是多数表决，即由输入实例的 K 个最临近的训练实例中的多数类决定输入实例的类别

（3）距离度量一般采用 Lp 距离，当p=2时，即为欧氏距离，在度量之前，应该将每个属性的值规范化，这样有助于防止具有较大初始值域的属性比具有较小初始值域的属性的权重过大。

5. KNN算法不仅可以用于分类，还可以用于回归。通过找出一个样本的k个最近邻居，将这些邻居的属性的平均值赋给该样本，就可以得到该样本的属性。更有用的方法是将不同距离的邻居对该样本产生的影响给予不同的权值(weight)，如权值与距离成反比。　该算法在分类时有个主要的不足是，当样本不平衡时，如一个类的样本容量很大，而其他类样本容量很小时，有可能导致当输入一个新样本时，该样本的K个邻居中大容量类的样本占多数。 该算法只计算“最近的”邻居样本，某一类的样本数量很大，那么或者这类样本并不接近目标样本，或者这类样本很靠近目标样本。无论怎样，数量并不能影响运行结果。可以采用权值的方法（和该样本距离小的邻居权值大）来改进。

该方法的另一个不足之处是计算量较大，因为对每一个待分类的文本都要计算它到全体已知样本的距离，才能求得它的K个最近邻点。目前常用的解决方法是事先对已知样本点进行剪辑，事先去除对分类作用不大的样本。该算法比较适用于样本容量比较大的类域的自动分类，而那些样本容量较小的类域采用这种算法比较容易产生误分。

实现 K 近邻算法时，主要考虑的问题是如何对训练数据进行快速 K 近邻搜索，这在特征空间维数大及训练数据容量大时非常必要。

## 三、实验过程

本实验一步步地构造使用k-近邻分类器的手写识别系统。为了简单起见，这里构造的系统只能识别数字0到9，参加图2：

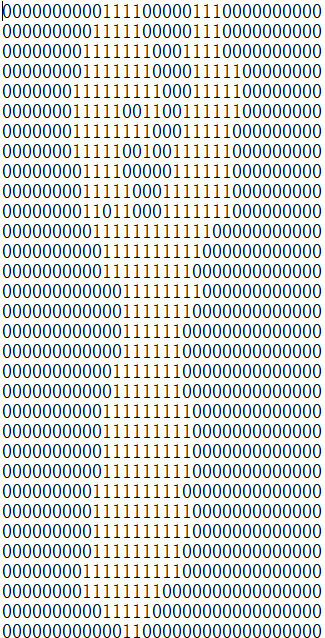


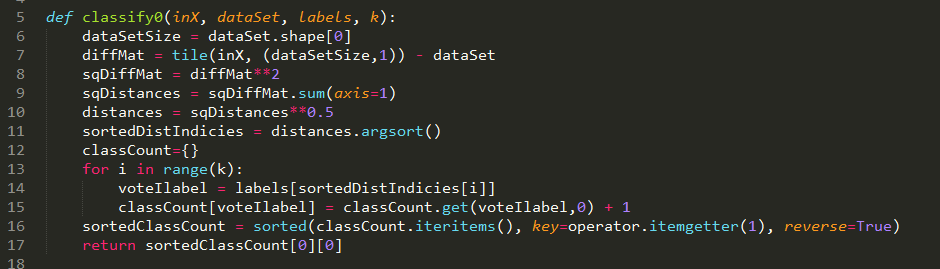
图2 手写数字数据集例子

需要识别的数字已经使用图形处理软件，处理成具有相同的色彩和大小:宽高是32像素x32像素的黑白图像。尽管采用文本格式存储图像不能有效地利用内存空间，但是为了方便理解，我们还是讲图像转换为文本格式。

1. 收集数据：提供文本文件

实际图像存储在两个子目录下，目录trainingDigits中包含了大约2000个例子，每个例子的内容如图2所示，每个数字大约有200个样本；目录testDigits中包含了大约900个测试数据。我们使用目录trainingDitgits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。

2. 准备数据：编写函数classify0()，将图像格式转换为分类器使用的list格式。

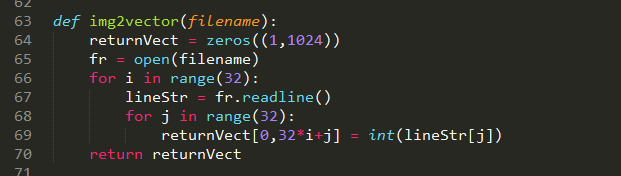


Classify0()函数有4个输入参数：用于分类的输入向量是intX，输入的训练样本集为dataSet，标签向量为labels，最后的参数k表示用于选择最近邻近的数目，其中标签向量的元素数目和矩阵dataSet的行数相同。

程序使用欧式距离公式，计算两个向量点xA和xB之间的距离:



计算完所有点之间的距离后，可以对数据按照从小到大的次序排序。然后，确定前k个距离最小元素所在的主要分类，输入k总是正整数，最后，将classCount字典分解为元组列表，然后使用程序第二行导入运算符模块的itemgetter方法，按照第二个元素的次序对元组进行排序。此处的排序为逆序 。



编写一段函数img2vector，将图像转换成向量：该函数创建1x1024的Numpy数组，然后打开给定的文件，循环读出文件的前32行，并将每行的头32个字符值存储在Numpy数组中，最后返回数组。

3. 分析数据：在Python命令提示符中检查数据，确保它符合要求。

4. 测试算法：使用k-邻近算法识别手写数字



将trainingDigits目录中的文件内容存储在列表中，然后可以得到目录中有多少文件，并将其存储在变量m中。接着，代码创建一个m行1024列的训练矩阵，该矩阵的每行数据存储一个图像。我们可以从文件名中解析出分类数字。该目录下的文件按照规则命名，如文件8\_34.txt的分类是8，它是数字8的第34个实例。然后我们可以将类代码存储在hwLabels向量中，使用前面讨论的img2vector函数载入图像。在下一步中，我们对testDigits目录中的文件执行相似的操作，不同之处是我们并不将这个目录下的文件载入矩阵中，而是使用classify0()函数测试该目录下的每个文件。由于文件中的值已经在0和1之间

classifyNB函数有四个输入：要分类的向量vec2Classify以及使用trainNB0（）计算得到的三个概率。使用NumPy()的数组来计算两个向量相乘的结果。这里的相乘是指对应元素相乘，即先将两个向量中的第1个元素相乘，然后将第2个元素相乘，以此类推。接下来将词汇表中所有词的对应值相加，然后将该值加到类别的对数概率上。最后，比较类别的概率返回大概率对应的类别标签。

6. 使用算法：在Python命令提示符中输入kNN.handwritingClassTest()，测试该函数的输出结果。

## 四、实验结果

运行结果如图1 和图2所示。

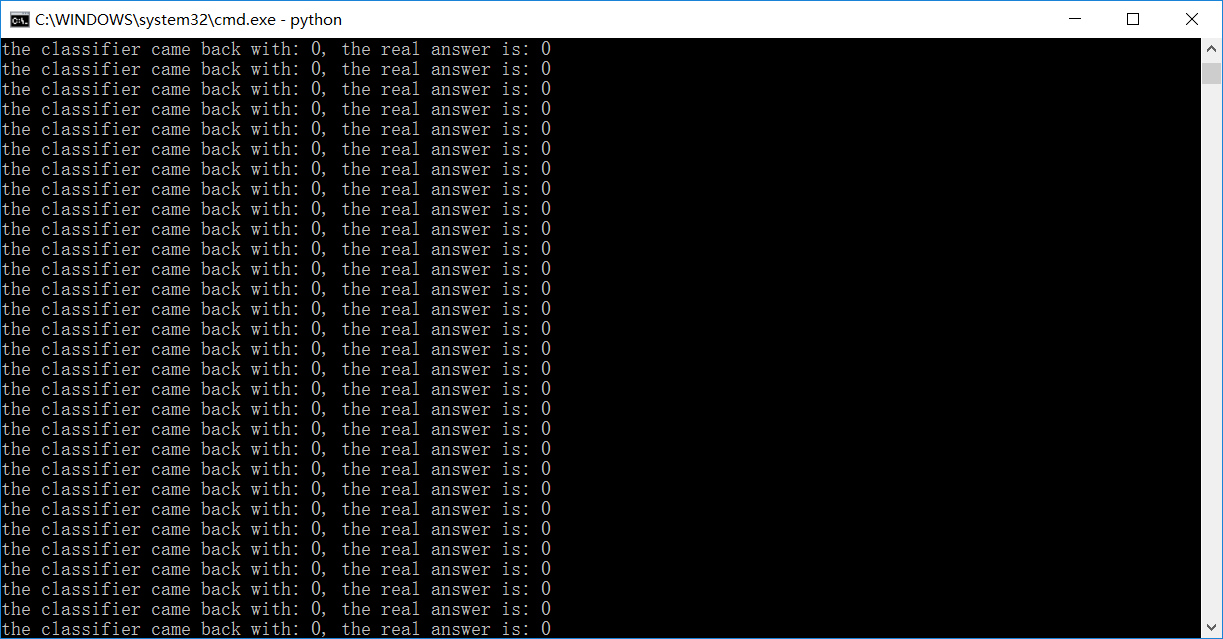


图 1 实验结果运行1

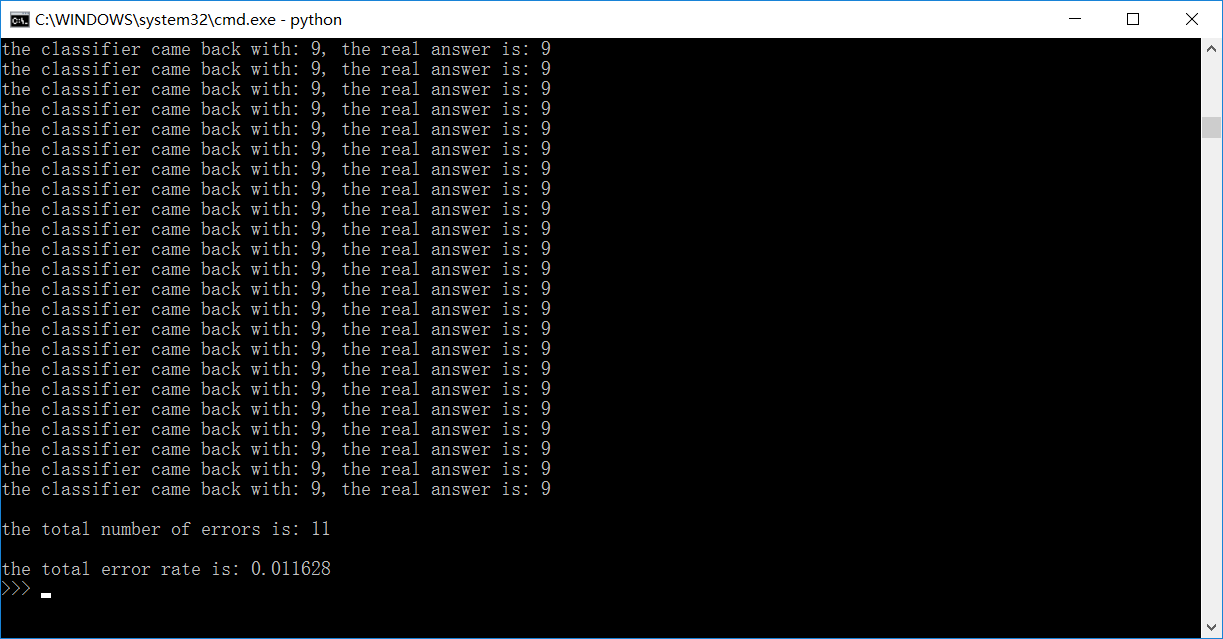


图2 实验运行结果2

k-邻近算法识别手写数字数字集，错误率为1.2%。改变变量k的值，修改函数handwritingClassTest随机选取训练样本、改变训练样本的数目，都会对k-邻近算法的错误率产生影响。

实际使用这个算法的时候，算法的执行效率并不高。因为算法需要为每个测试向量做2000次距离计算，每个距离计算包括了1024个维度浮点运算，总计要执行900次，此外，我们还需要为测试向量准备2MB的存储空间。

## 五、实验总结

k-邻近算法是分类数据最简单最有效的算法，本章通过两个例子讲述了如何使用k-邻近算法构造分类器。K-邻近算法是基于实例的学习，使用算法时我们必须要有接近实际数据的训练样本数据。K-邻近算法必须保存全部数据集，如果训练数据集的很大，必须使用大量的存储空间。此外，由于必须对数据集中的每个数据计算距离值，实际使用时可能非常耗时。

k-邻近算法的另一个缺陷是它无法给出任何数据的基础结构信息，因此我们也无法知晓平均实际样本和典型实例样本具有什么特征。