Homework1 项目文件介绍

项目文件介绍：

1、preprocess.py:功能：预处理数据。将不同类别下的文件进行分词、去停用词、词干提取等操作，利用文件读写操作将处理后的数据存储下来。最终得到的文件格式为每行一个单词。

2、devide.py：功能：实现对训练集和测试集的划分。遍历预处理后的文件，统计每个类别下文件的数目，将每一类别下 80%的文件拷贝到训练集，20%的文件拷贝到测试集。

3、VSM.py:功能：将每个文档表示成向量。主要包括三个函数：file_count（）、cal_idf（）、create_vector（）

file_count（）:用来计算数据集内所有文档的数目。为数据集建立字典时，需要得到每个单词的 idf 值，计算 idf 值时，需要用到文档总数。

cal_idf（）：返回值为数据集的字典，格式为：{'word1':idf1,'word2':idf2,.....}

其中的 key，并不是数据集中所有出现过的单词，而是通过为数据集建立词频字典，统计每个单词出现的次数，过滤掉词频数小于 10 的单词。计算每个词的 idf 值时，需要用到 df 值。df 指的是，对于一个单词而言，数据集中有多少数目的文档包含着它。为了得到每个单词的 tf 值、df 值，遍历每个文档时，借助 Counter 函数得到每个文档内单词出现次数的词典，由此更新数据集内每个单词的 tf、df 词典。使用数据集的 tf 词典对 df 词典进行过滤筛选，删除出现频率小的无意义的词，然后通过 idf 的计算公式得到各个词的 idf 值，至此得到了数据集的字典。createVector（）：该函数的功能是将数据集中的所有文档均表示成向量，最终将数据集内所有文档表示成向量的列表，

保存到指定路径下。值得注意的是，tf_idf 值中的 tf 指的是标准化词频，它需要用到一个文档中某个单词的出现次数来计算。遍历每个文档，每个文档中的每个单词，只要它存在于数据集的字典中，就需要统计它在该文档中的出现次数，并计算它的 tf_idf 值。对于每个文档，均可得到 value 值为其 tf_idf 值的字典，且字典值中的 key 均是数据集字典中存在的单词。由于 KNN 算法需要用到每一个文档的类型作为 label，所以每个文档的向量形式最终是一个 list，其中包括 label 和字典。为了在接下来的 KNN 算法中减小求两向量之间的相似度时的运算量，所以将一个文档的字典仅保留其 tf_idf 值最大的前 50 个词作为该文档的关键词。

KNN.py:是 Knn 分类器的实现代码。主要包括 cal_eachVC_len ()、cos_twoVC ()、KNN()三个函数。

cal_eachVC_len （）：计算一个向量的模长。

cos_twoVC ()：计算两个向量的相似度。主要利用公式：相似度=AB/|A|*|B|。

KNN（）：对于测试集的每一个向量，计算其与每一个训练向量的相似度，将训练向量的类型和相似度作为二元组存储在列表中，取列表中相似度最大的 K 个元组（需要排序操作），统计元组中类型出现次数，选取出现次数最多的类别作为该测试向量的分类结果。分类结果与训练向量的类名进行比较，统计分类正确的次数与分类错误的次数，计算分类正确率。