山东大学计算机科学与技术学院

数据挖掘实验报告

实验题目： Homework 2: NBC		学号： 201814852
日期：2018.11.14	班级： 18 级学硕班	姓名： 周培衍
Email：zhoupeiyan123@hotmail.com		

实验目的：

1) 实现朴素贝叶斯分类器，测试其在 20 Newsgroups 数据集上的效果。

2) 比较不同三种计算模型的优劣，观察分类效果

硬件环境：

Inter Core i7-7700 @ 3.60GHz 四核 + Dell 062KRH + 4G + P3-256

软件环境：

Windows 10 + Git + JetBrains PyCharm Community Edition 2018.2.4

实验步骤与内容：

一、 数据集

1) 数据集的预处理

数据集的获取和划分 Homework1 中已经介绍，这里不在赘述，从已经分词预处理之后的文档（训练、测试）开始操作。

二、 NBC

1) 生成文档词频计数及合并后计数---FileNB.py

统计预处理后的文档的词频计数。

首先读入所有的文档，每篇文档记录其词生成的词典：

word_list {"文档名": "["a", "b", ...] ..."}

生成每篇文档的词频计数：

countlist {"文档名": Counter('car': 12, ...), ...}

将训练数据和测试数据的 countlist 以 json 形式分别存储下来。

FileNB：训练文档词频计数

FileNBTesting：测试文档词频计数

训练数据合并同类别文档，为了之后 NBC 计算方便。

具体就是遍历 countlist，将相同类别的统一计算词频，在每一个类别

的 value 值词典中加入键值”zhoupeiyan”，他的 value 为该类别下文档总数，方便计算 P(Vj)：

merge_countlist {"类别": Counter('car': 12, ...'zhoupeiyan': 5555), ...}

将 merge_countlist 以 json 形式存储下来，文件名为 MergeFileNB。

这里需要注意一下，伯努利模型下不需要考虑训练数据中的的词频重复，所以每篇训练文档的词频计数都更改为 1，再次调用上述的合并同类别文档，以 json 形式存储下来，命名为 MergeCountFileNB。

MergeFileNB：生成合并后的文档词频计数

MergeCountFileNB：生成合并后的文档篇数计数

2) Naïve Bayes Classifier

朴素贝叶斯分类器公式推导：

vj 为不同类别，x1…xn 为测试文档里的每一个词

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j | x_1, x_2, \ldots, x_n)$$

运用 bayes’ s rule

$$= \arg\max_{v_j \in V} \frac{P(x_1, x_2, \ldots, x_n | v_j) P(v_j)}{P(x_1, x_2, \ldots, x_n)}$$

因为对于不同的 vj，比较的分母都是相同的，所以

$$= \arg\max_{v_j \in V} P(x_1, x_2, \ldots, x_n | v_j) P(v_j)$$

使用 naïve bayes 假设，条件独立

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(x_i | v_j)$$

3) 小 tricks

(1) 将连乘Π取对数(log)变为连加Σ，有效的减轻了计算量，而且连乘也会最后越来越趋向于 0，导致比较效果不好。

(2) 如果测试数据中的词在某个类别的词频词典中没有出现过，那么 P(xi|vj)等于 0，乘起来就等于 0，效果肯定不好，所以采取**平滑处理技术(Laplace 技术)**，具体方法就是：

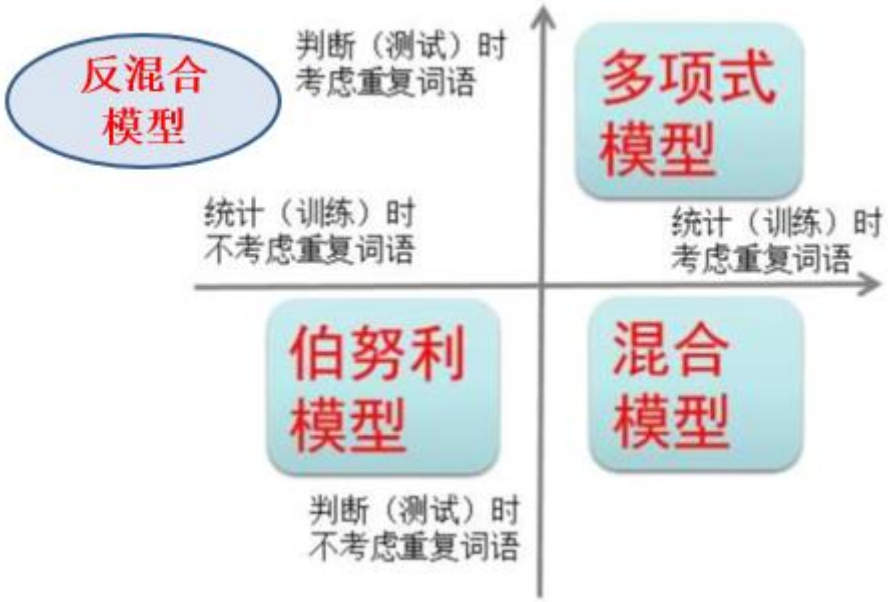
$$\hat{P}(w_i \mid c) = \frac{count(w_i, c)}{\sum_{w \in V}\left(count(w, c)\right)}$$

$$= \frac{count(w_i, c)+1}{\left(\sum_{w \in V} count(w, c)\right) + |V|}$$

虽然很土很朴实，但是直接且有效。

(3) 如果只是判断“是不是”问题的话，即分类类别为 2 时，可以直接将 log(P(xi|v0)/ P(xi|v1))求和，判断和 0 的大小比较得出归为哪一种类别。但是此次实验做的是 20 个类别，所以该 trick 并不能用。

4) 编程实现 NBC---naïve_bayes.py



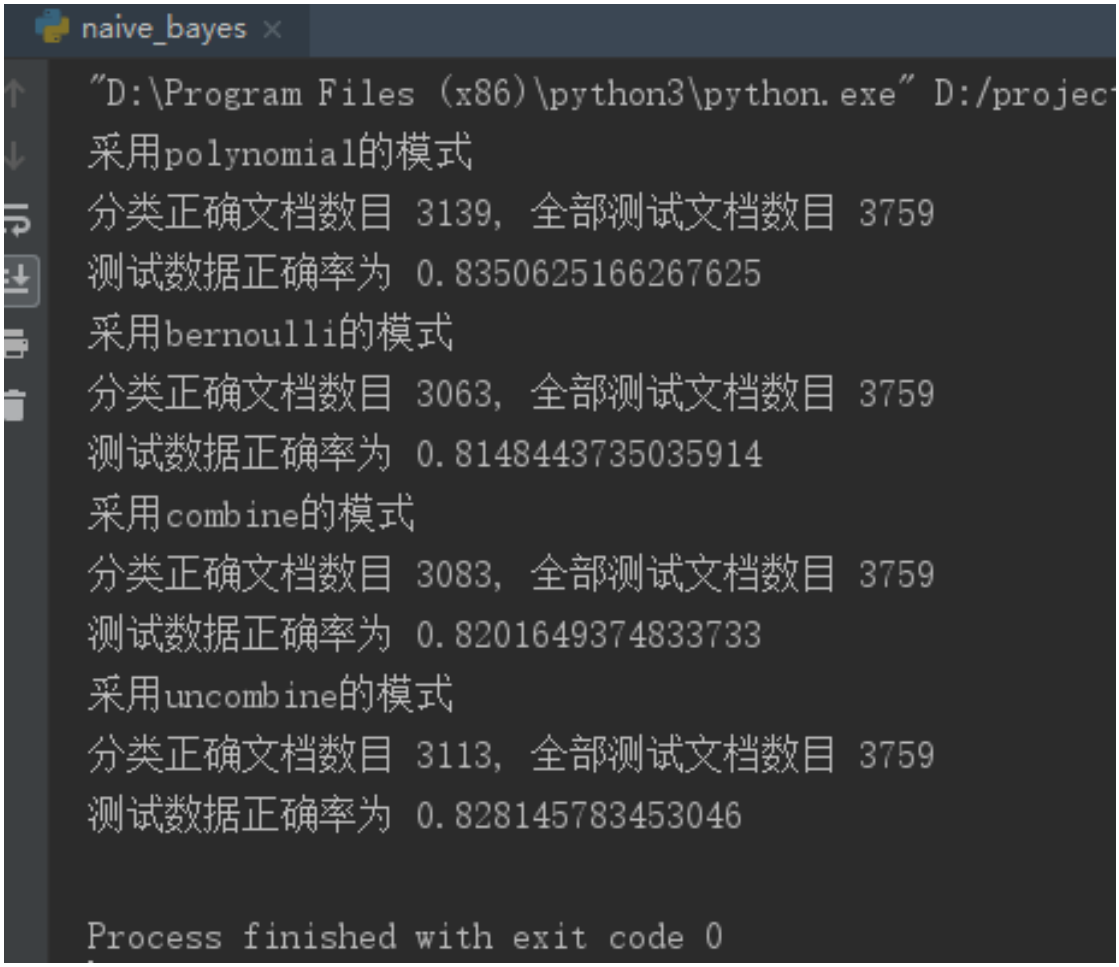
(1) 多项式模型，重复的词语视为出现多次，统计和判断都关注重复次数。因此会有一些 P(xi|vj)的次幂。使用 MergeFileNB。

(2) 伯努利模型，重复的词语视为只出现一次，这样会更加简化和方便。因为它丢失了词频的信息，所以效果可能会差一些。使用 MergeCountFileNB。

(3) 混合模型，统计时考虑词语出现的次数，不过判断时不考虑，这样一折，也因为丢失了词频的信息，效果会差一些。使用 MergeFileNB。

(4) 反混合模型，这个老师没有讲，是我自己瞎想的，就是统计时不考虑词语出现的次数，判断时考虑词语出现的次数，和混合模型一样效果会差一些。使用 MergeCountFileNB。

观察运行结果：



明显得知多项式模型分类最好，达到 83.5%，然而伯努利模型分类效果最差，为 81.5%，混合模型和（自己瞎整的）反混合模型分类结果居中，分别为 82.0%和 82.8%，和之前分析的词频重要性影响分类效果一致。

---NBC vs vsm&knn:

运行时间 NBC 更短，1、2 分钟就可以；然而 vsm&knn 就要半个小时左右。

分类效果 NBC 是 83.5%，vsm&knn 是 75.4%，NBC 分类效果更好。

---NBC 优缺点：

-优点：

(1) 对待预测样本进行预测，过程简单速度快（求和，log 下的加法）。

(2) 对于多分类问题同样很有效，复杂度也不会有大程度上升。

(3) 在分布独立这个假设成立的情况下，贝叶斯分类器效果奇好，会略胜于逻辑回归，同师我们需要的样本量也更少一点。

-缺点：

(1) 对于测试集中的一个类别变量特征，如果训练集中没有出现过，直接计算概率 P(xi|vj)就是 0 了，分类效果会减少很多。当然采取平滑技术可以缓解，最常见的是 Laplace 技术。

(2) 朴素贝叶斯算出的概率结果，比较大小还凑活，实际物理意义会小的可怜，不要太当真。

(3) 朴素贝叶斯有分布独立的假设前提，但现实生活中很难完全独立。

结论分析与体会：
Homework2 的代码量比 Homework1 简洁很多，运算时间和分类效果也都更好，看来对于分类问题 NBC 更有优势，也知晓了 NBC 的优点和缺点，更理解了朴素贝叶斯分类器的原理和实现，分类效果在我看来已经蛮好了。虽说 NBC 看起来蠢萌，再加上看起来土土的 laplace 平滑技术，但实践证明在垃圾邮件识别的应用还令人诧异地好。引自 Paul Graham《黑客与画家》“1000 封垃圾邮件能够被过滤掉 995 封，并且没有一个误判。”