华东师范大学数据科学与工程学院上机实践报告

课程名称：统计方法与机器学习　　**年级**：大三　　**上机实践成绩**：
指导教师：董启文　　**姓名**：刘蔚璁　　**学号**：10225501443

实验二：文本分类

[摘要]：本实验旨在实现基于朴素贝叶斯算法的文本分类任务，将 20000 篇文档划分为 20 个类别。实验使用 20 Newsgroups 数据集，并对文本进行了清洗、分词、去停用词等预处理操作。采用五重交叉验证评估分类模型的性能，结果显示朴素贝叶斯分类器在高维文本数据中具有较高的准确率和效率，适用于文本分类任务。然而，由于特征条件独立假设的限制，部分类别的分类效果略低。最终，实验探讨了模型的优缺点，并提出了改进建议，如特征选择优化和多模型融合，以进一步提升分类效果。

一、目标与要求

- 使用朴素贝叶斯在 20 Newsgroups 数据集上完成文本分类任务
- 使用五折交叉验证结果

二、实验数据

实验使用的是著名的 20 Newsgroups 数据集，广泛用于文本分类和自然语言处理任务。该数据集包含 20,000 篇文档，分布在 20 个不同的新闻组类别中，每个类别的文档数量基本均衡，单个类别包含约 1000 篇文档。

- **数据来源：** http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html
- **数据内容**

  每篇文档为一段文本，内容来源于新闻组的讨论帖子，文档中可能包含标题信息、邮件头（如发件人、收件人）、帖子正文等，文本长度差异较大，部分文档较短，部分文档较长。

- **样例展示**

```
print("文档示例：", texts[seed])
✓ 0.0s
文档示例： Path: cantaloupe.srv.cs.cmu.edu!magnesium.club.cc.cmu.edu!news.sei.cmu.edu!cis.ohio-state.edu!pacific.mps.ohio-state.edu!
From: perry@dsinc.com (Jim Perry)
Newsgroups: alt.atheism
Subject: Re: Where are they now?
Date: 20 Apr 1993 14:08:38 -0400
Organization: Decision Support Inc.
Lines: 26
Message-ID: <1r1e76INNl18@bozo.dsinc.com>
References: <1qi156INNf9n@senator-bedfellow.MIT.EDU>
NNTP-Posting-Host: bozo.dsinc.com

Perhaps it's prophetic that the week "Where are they now?" appears and
I can claim to be a still-active old-timer, my news software gets bit
rot and ships outgoing articles into a deep hole somewhere...  Anyway,
here's a repost:

In article <1qi156INNf9n@senator-bedfellow.MIT.EDU> tcbruno@athena.mit.edu (Tom Bruno) writes:
>
>Which brings me to the point of my posting.  How many people out there have
>been around alt.atheism since 1990?  I've done my damnedest to stay on top of
>the newsgroup, but when you fall behind, you REALLY fall behind [...]

These days you don't have to fall far behind... Last Monday
(admittedly after a long weekend, but...) I had 800+ messages just in
those few days.  Aside from a hiatus while changing jobs last Fall
...
--
Jim Perry   perry@dsinc.com   Decision Support, Inc., Matthews NC
These are my opinions.  For a nominal fee, they can be yours.
```

三、实验方法

- **算法选择**
  - **分类器选择——多项式朴素贝叶斯**

    在文本分类任务中，多项式朴素贝叶斯是最常用的变体，其模型假设如下：

    - 每个文档由一组词组成，词的出现频率是文本特征的主要来源
    - 特征的条件概率 $P(word \mid Class)$ 由类别下词的频率分布建模

    基于贝叶斯定理，给定一篇文档 X 和类别 C，后验概率为：

    $$P(C|X) \propto P(C) \cdot P(X|C)$$

    其中：

    - $P(C)$: 类别 C 的先验概率，表示每个类别的文档比例
    - $P(X \mid C)$: 在类别 C 下生成文档 X 的条件概率

    多项式朴素贝叶斯将文档 X 表示为词的频率向量 $(x_1, x_2, ..., x_n)$，条件概率进一步分解为：

    $$P(X|C) = \prod_{i=1}^{n} P(\text{word}_i|C)^{x_i}$$

    其中 x_i 是词 $word_i$ 在文档中的出现次数，$P(word_i \mid C)$ 是类别 C 下词 $word_i$ 的条件概率。

  - **未见词处理——拉普拉斯平滑**

    在多项式朴素贝叶斯中，条件概率 $P(word_i \mid C)$ 的计算依赖于训练集中某个词 $word$ 在类别 C 下的频率。如果某个词 $word$ 未在类别 C 中出现，则 $P(word_i \mid C)=0$。这会导致整个 $P(X \mid C)$ 变为 0，从而影响模型的分类结果。

    因此引入拉普拉斯平滑，通过为每个词加上一个固定值（通常为 1），避免零概率问题。公式如下：

    $$P(word|C) = \frac{\text{词频} + 1}{\text{类别中所有词的总频数} + |\text{词汇表大小}|}$$

- **数据预处理**

  在文本分类任务中预处理非常关键，它直接影响特征的质量以及分类模型的效果。

  - **标签分布**

    样本是否均衡是决定模型性能的很重要部分，所以我们先查看标签分布：

```
# 统计标签分布
from collections import Counter
label_counter = Counter(labels)
for i in range(1, len(category_map), 10):
    for j in range(10):
        print(f"{i+j}: {label_counter[i+j]}", end='; ')
    print()
```

✓ 0.0s

```
1: 1000; 2: 1000; 3: 1000; 4: 1000; 5: 1000; 6: 1000; 7: 1000; 8: 1000; 9: 1000; 10: 1000;
11: 1000; 12: 1000; 13: 1000; 14: 1000; 15: 1000; 16: 997; 17: 1000; 18: 1000; 19: 1000; 20: 1000;
```

    可以看出标签分布基本均衡。

  - **文本清洗**

    因为文档内容来源于新闻组的讨论帖子，如样例所示，包含了很多与分类无关或

有干扰的内容，所以需要对原文档进行清洗。

✧ **元数据清理**

文档头部包含如文档的传输路径、发件人、收件人、发布文档的主机信息、文档行数等诸多冗余信息，与分类任务无直接关联，去除它们可以减少噪声，提升分类的有效性。

✧ **电子邮件地址、URL 清理**

查看文档会发现除了头部元数据，文档还会出现各种电子邮件地址和网址，这些信息同样对分类没有直接帮助，反而会造成词表的无意义扩大，故进行删除。

```
def remove_metadata(text):
    metadata_pattern = r"(Path|From|Date|Organization|Lines|Message-ID|References|NNTP-Posting-Host|Reply-To|Sender|Xref|In-reply-to):.*"
    text = re.sub(metadata_pattern, "", text)
    return text.strip()
✓ 0.0s

def normalize_emails_and_urls(text):
    # 替换邮件地址
    text = re.sub(r"\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b", "", text)
    # 替换 URL
    text = re.sub(r"http\S+|www\.\S+", "", text)
    return text.strip()
✓ 0.0s
```

➢ **分词处理 & 去停用词**

分词是将一段连续的文本分解为独立的单词或词组的过程；停用词是指在文本中出现频率较高、但对分类等任务贡献较小的词，例如"the"、"is"、"and"等。

✧ **分词处理**

首先将文本中的字母转为小写，保证特征一致性；然后使用 `string.punctuation` 提供的标点符号列表，将文本中的标点符号替换为空字符；最后使用 `nltk` 的 `word_tokenize` 方法，将文本转化为单词列表。

✧ **去停用词**

同样使用 `nltk` 提供的 `stopwords.words('english')`获取英文停用词列表，然后遍历分词结果，检查每个单词是否在停用词表中，若不在则保留。

文档处理后如下（仍使用样例进行展示）：

```
print("文档示例（预处理后）：", texts_tokenized[seed])
✓ 0.0s
文档示例（预处理后）： ['newsgroups', 'altatheism', 'subject', 'perhaps', 'prophetic', 'week', 'appears', 'claim',
```

● **实验设计**

➢ **数据划分——五折交叉验证**

五折交叉验证将数据集等分为五个折。在每次训练时，选取其中一折作为验证集，其余四折作为训练集，轮流进行训练和验证，最终计算 5 次实验结果的平均值和标准差作为模型的整体性能指标。五折交叉验证能够充分利用数据集，避免因单次随机划分数据导致的结果偏差，使评估结果更加稳定。

✧ 使用 Scikit-learn 提供的用于实现交叉验证的函数 KFold 进行划分

```
kfold = KFold(n_splits=k_folds, shuffle=True, random_state=seed)
kfold.split(train_dataset)
```

➢ 模型训练

该模型训练过程较简单，只需要根据算法选择部分的公式在训练集上计算每个类别的先验概率和条件概率，然后在验证集上计算概率并加上拉普拉斯平滑即可。

➢ 模型评估

利用测试集对模型性能进行评估，计算分类任务的主要指标：

✧ **准确率**

$$\text{Accuracy} = \frac{\text{预测正确的样本数}}{\text{总样本数}}$$

✧ **精确率**

$$\text{Precision} = \frac{\text{被正确分类的正类样本数}}{\text{被预测为正类的样本总数}}$$

✧ **召回率**

$$\text{Recall} = \frac{\text{被正确分类的正类样本数}}{\text{正类样本总数}}$$

✧ **F1 值**

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

除此之外，我还绘制了混淆矩阵，通过混淆矩阵可以直观显示模型对每个类别的分类效果，对角线元素代表正确分类的样本数，非对角线元素则表示错误分类的样本数，显示类别之间的混淆情况。

四、实验结果

- **模型性能**

➢ 性能指标

可以看出朴素贝叶素分类器在各折上表现平均，且均取得较好的结果：

折序号	准确率	精确率	召回率	F1 值
1	89.40%	89.30%	89.26%	89.00%
2	88.62%	88.66%	88.65%	88.52%
3	89.17%	88.92%	89.04%	88.80%
4	89.12%	89.28%	89.35%	89.10%
5	88.85%	88.83%	89.00%	88.66%

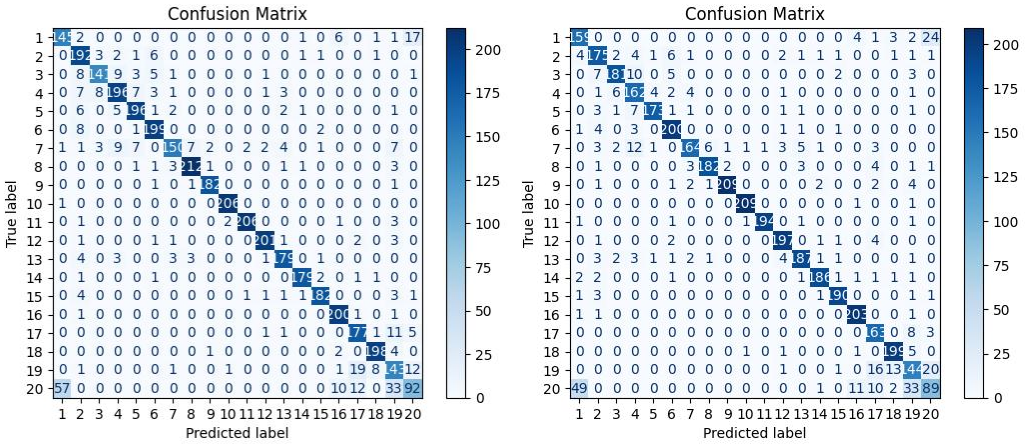
➢ 混淆矩阵

因为各折混淆矩阵表现相似，所以此处选取第一折、第三折展示，可以看出第一类和第二十类，第十七、十八和第十九类上出现误分类的情况比较多，其中：

✧ 第一类 talk.religion.misc 和第二十类 alt.atheism 都是宗教相关的话题，第一类涵盖所有宗教相关话题，第二类专注于无神论及与宗教信仰的关系，二者关联性较强，词汇重合性较高，所以出现误分类的概率大。

✧ 第十七类 talk.politics.guns、十八类 talk.politics.mideast、和十九类 talk.politics.misc

都是和政治相关的分类，同样话题重合度较高。



- **未进行预处理的数据分类效果**

  最后我在未进行任何预处理的数据集上进行分词验证，分类准确率**显著下降**，只有83.30%，且每折**训练时长大大提高**，在预处理的数据集上每折训练时间约 1.96min，而未经预处理的数据集上每折训练时长达到了 16min+，因为没有预处理会导致数据特别稀疏。

五、分析与讨论

- **朴素贝叶斯分类器在文本分类任务中的表现**

  实验结果表明，朴素贝叶斯分类器在文本分类任务中具有较高的准确率和效率，尤其是对于高维稀疏数据（如文本数据）表现出色。混淆矩阵分析显示，类别之间的混淆主要集中在主题相似的类别上。

- **数据预处理和特征选择对模型效果的影响**

  去除元数据（如邮件头信息、标点符号、特殊字符）显著提高了模型的性能，避免了冗余信息对分类的干扰，且降低了矩阵的稀疏性。停用词过滤减少了无意义的高频词对分类的影响，使模型更加关注有意义的特征。

- **优缺点分析**

  ✧ 朴素贝叶斯分类器通过统计先验概率和条件概率进行分类，不涉及复杂的参数优化过程，训练和预测速度非常快。文本数据通常是高维稀疏的，词汇表可能包含数万甚至数十万个词。朴素贝叶斯能够很好地适应这种数据结构。

  ✧ 朴素贝叶斯假设特征之间是条件独立的，而实际数据中这种假设往往不成立，这会导致模型在某些类别上无法捕捉复杂的语义关系，从而影响分类性能。如果某些类别的样本较少，模型对这些类别的条件概率估计可能不准确，进而影响分类效果。

- **改进建议**

  ✧ 通过特征选择减少无关特征的干扰。例如卡方检验或互信息方法，用于挑选与类别相关性较强的特征。

  ✧ 尝试更复杂的模型，对比使用逻辑回归、SVM 或神经网络模型，进一步提升分类性能。

  ✧ 针对类别不平衡或低频特征问题，引入采样技术（如过采样或欠采样）或模型平滑技术。