**华中科技大学计算机科学与技术学院**

**《机器学习》课堂三结课报告**



专 业： 计算机科学与技术

班 级： ACM1901

学 号： U201914965

姓 名： 卫云泽

成 绩：

指导教师： 何琨

**完成日期： 2021年 12月 9日**

目录

[新闻数据情感预测 2](#_Toc89960506)

[一、实验题目与分析 2](#_Toc89960507)

[二、数据清洗 4](#_Toc89960508)

[三、文本向量化 6](#_Toc89960509)

[四、模型方法 7](#_Toc89960510)

[五、实验环境与平台 8](#_Toc89960511)

[六、程序实现 9](#_Toc89960512)

[七、实验结果与分析 13](#_Toc89960513)

[八、总结与反思 16](#_Toc89960514)

[参考资料 18](#_Toc89960515)

# 新闻数据情感预测

## 一、实验题目与分析

**1.1 题目背景**

随着计算机技术与互联网的飞速发展，新闻媒体逐渐由传统的报纸、电视等形式走向多元的网络化形式。网络新闻具有发行门槛低、迭代速度快等特点，在人人都是自媒体的时代，相比传统新闻媒体，网络新闻的内容往往更能够反应一个地区、一段时间的人们对某个事件的真实情感态度，新闻数据的情感预测的重要性日益显著。

新闻数据情感预测的应用领域十分广泛，不仅可以为相关部门的舆情监控提供新的手段，也可以作为新闻应用推送定制新闻的评估指标。在新闻量呈指数级增长的今天，新闻的人工审查已经成为过去式，寻找全新、可靠的新闻情感预测方法迫在眉睫。随着自然语言处理技术的融合发展，使用机器学习方法分析新闻情感已经成为一种较为普遍的应用。

**1.2 数据集**

本次实验的数据集是针对近年新冠疫情的新闻数据。数据集被包括用于训练的 2200 条新闻与用于测试的 907 条新闻，label 由 0 或 1 表示，其中 0 代表负面（negative），1 代表正面（positive）。训练集内容概览如图1-1所示。

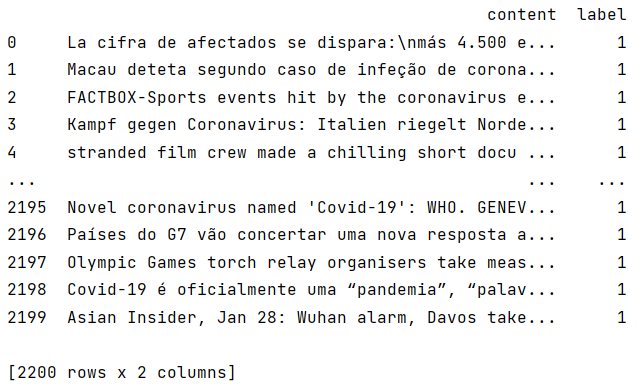


图 1‑1 训练集概览

训练集具有以下两个特征：1. 训练集由多国语言构成，因此数据清洗时可能需要考虑翻译问题；2. 展示的训练集条目的标签都为1，因此训练集可能存在样本不平均的问题。针对上述两个问题，进一步分析数据集如下。

表 1‑1 训练集标签分布

|  |  |
| --- | --- |
| 正面新闻（label=1） | 负面新闻（label=0） |
| 2000 | 200 |

表 1‑2 训练集语言分布

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 语言 | es | pt | en | de | ru | af | zh-cn |
| 数目 | 54 | 556 | 1405 | 162 | 21 | 1 | 1 |

由表1-1可知，训练集正、负样本比例为10:1，比例极不均衡，数据处理时需要根据模型需要平衡样本比例；由表1-2可知，训练集由7国语言构成，且各国语言比例也极不均衡，因此数据处理时可能需要考虑翻译。

**1.3 任务描述**

基于上述训练集，使用合适的数据预处理方法和机器学习模型，对测试集的新闻情感进行预测，目标是获得最高的Macro F1-score。

**1.4 评测标准**

模型的评价标准有许多种，不同的评价标准有不同的侧重点。对于二分类而言，其混淆矩阵如表1-3所示。

表 1‑3 二分类混淆矩阵示意

|  |  |  |
| --- | --- | --- |
|  | 预测为正 | 预测为负 |
| 正样本 | 真正类（TP） | 假负类（FN） |
| 负样本 | 假正类（FP） | 真负类（TN） |

考虑正样本中的实例被判断为正类的比率，引入召回率：

考虑预测为正中的实例中的实际为正样本的比率，引入精确率：

负样本的召回率和精确率可类似给出。

对于每一类样本，引入F1-score：

综合正负类样本的F1-score ，引入Macro F1-score：

Macro F1-score的优点是可以综合各个类别的召回率和精确率，同时不受数据不平衡的影响，可以较为准确地评价模型对各个类样本的预测效果。

## 二、数据清洗

**3.1 文本翻译**

由1.2节可知，训练集由数量极不均衡的多国语言新闻条目构成，其中英语语料占据最大比例。不同语种的文本材料清洗方法有所不同，且目前英语文本的数据清洗方法最为成熟；此外，从模型训练的角度分析，不同语种往往实际上是独立的样本空间，若不进行文本翻译，则在后续预测中非英语新闻条目的预测准确度可能较低。综合上述两条因素，数据清洗之前有必要进行文本翻译。本次实验文本翻译使用开源谷歌翻译API：googletrans[2]。翻译的实现细节详见6.1节。

**3.2 数据增强**

由于训练集的正负样本比例为10：1，比例极不均衡，且逻辑回归、支持向量机等模型对样本比例不均衡较为敏感，因此需要采取一定的数据平衡手段。其中一种策略是在模型训练时，针对不同类别的样本，对其惩罚参数进行加权，使分类器更加“关注”少数类的样本[5]。这样的方法固然有效，但可能会对负样本的噪声数据较为敏感，同时对正样本的预测准确性带来影响。

另一种策略是对数据集本身进行增强。数据集增强的方法有许多种，如基于近义词或词向量的词汇替换、反向翻译、文本表面转换和随机噪声注入等[3]。由于词汇替换往往采用从近义词库中随机选取近义词进行替换的方法，并未考虑词语在样本中的上下文关系，故无法保证替换后的词汇与原词汇拥有一致的情感倾向；反向翻译则以分句为最小单位，一定程度上可以确保翻译文本的整体情感倾向不变。基于文本翻译的思路，本实验采用反向翻译的策略进行数据增强，即通过不同语言之间的相互翻译，实现相同语义的不同表达，表现为数据样本的增加，反向翻译的示例如图2-1所示。数据增强的实现细节详解6.2节。

****

图 2‑1 反向翻译示例

**3.3 文本处理**

原始数据集是未加任何处理的新闻文本，新闻文本符合人类阅读习惯，但对于新闻情感预测而言，新闻文本中并非所有字符都是“有用”的，其中与新闻情感无关的字符可能成为数据集中的噪声，故应当使用一定手段去除这些噪声数据，即文本过滤；此外，考虑到同一英文单词可能具有多种形态，还可以对英文单词进行处理，旨在降低数据维度、提升预测效果。下面介绍本文使用的若干文本处理方法[4]。

* **分词**：由于词形转换、词干提取和去除停用词等方法的操作最小单元是单词，故文本处理前需要对新闻进行分词。分词不只是将单词“分开”，还要根据操作的目的采取不同的策略，分词的实现细节详见6.3节。
* **去除标点、数字**：数字往往与新闻的情感态度无关，大多数标点符号如“.”、“，”和“-”等也不具备表达情感的功能，故应当去除这些字符。标点符号“!”、“?”与新闻情感的关联程度存在争议，是否应当去除取决于实验结果。
* **词形转换**：英语单词的动词存在现在分词、过去式、过去分词等多种形态，名词也具有单数、复数等形态，词形转换的目的是将同一单词的不同形态归一化。
* **词干提取**：词干提取是去除词缀得到词根的过程，例如“fish”、“fisher”和“fishing”同属于词根“fish”。将单词缩减为词干往往不改变其褒贬性，故在新闻情感预测中或可使用词干提取进行数据清洗。此外，词干提取和词形还原的结果具有交叉性，但考虑到词干提取后得到的未必是有实际意义的单词，如“happy”提取词干后变为了“happi”，故实际使用时首先进行词形转换，之后再执行词干提取操作。
* **去除停用词**：停用词是文本中使用频率相当高的一类词汇，如英文中的“I”、“a”、“do”等，它们往往没有特殊的情感倾向，在不同类别的文本中的平均出现频率相近，是文本中的噪声数据，故应当去除。但在新闻情感分析领域，诸如“not”等停用词是否应该被过滤存在争议，需要在实验中进行验证。

## 三、文本向量化

**3.1 词袋法**

词袋法（Bag of Words, BOW）是一种文本单词的向量化方法，即不考虑单词出现的顺序，只统计一篇文章中各个词汇出现的次数，构成一个词汇表，形成一个特征向量。由于一篇文章中的词汇量往往较大，但许多单词出现的频次可能非常低（如特殊名词等）或非常高（如停用词），使用词袋法时，往往会对提取的特征维度进行限制，如限制最大特征数、过滤部分出现次数最多或最少的词汇等，可以降低向量化后的样本维度、实现部分噪声数据的过滤等。

**3.2 TF-IDF算法**

词频-逆文档率（Term Frequency–Inverse Document Frequency, TF-IDF）是一种用于信息检索和数据挖掘的加权基数，它可以评估一个分词对于一份文本的重要程度。

对于相同重要程度的词语而言，其在较短样本中的词频会低于较长样本的，因此通常使用归一化词频来评估一个词在文本中的重要性。归一化词频算法如下：

其中是词语在样本​中的归一化词频，是词语​在样本中出现的次数，是样本的词语总数。

逆文档率是衡量一个词语总体重要性的指标，它可以用来降低各个文本中都含有的词语的词频权重。词语的逆文档率计算公式如下：

其中是词语的逆文档率，是样本总数，表示包含的样本数目。

结合和，词语在样本​中的TF-IDF权值可表示为：

## 四、模型方法

**4.1 模型算法**

实验中对比使用的机器学习模型算法[1]如下：

* K近邻算法（KNN）
* 支持向量机（SVM）
* 朴素贝叶斯
* 逻辑回归

其中支持向量机使用了高斯核、线性核等核函数，朴素贝叶斯包括伯努利朴素贝叶斯、多项式朴素贝特斯等。不同模型应用于新闻情感预测的效果对比见7.1节模型选择。

**4.2 模型评估**

由于评测目标是macro F1-score（见1.4节），因此模型评估使用基于macro F1-score的K折交叉验证[1]进行。K折交叉验证常用于模型调优，目的是找到使得模型泛化性能最优的超参值，其优点是采用了无重复抽样技术，在每一轮迭代过程中，每一个样本只会被划入测试集或训练集一次。5折交叉验证示意图如图4-1所示。

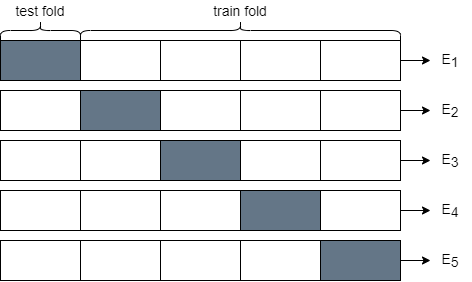


图4‑1 5折交叉验证示意图

图中每一行代表训练集在每一轮迭代中的划分情况，其中颜色较深的部分为测试集，其余为训练集，训练集和测试集的比例为4:1。为每一轮交叉验证的评估结果，交叉验证最终结果可以用均值表征。

**3.3 参数调优**

模型的参数调优使用网格搜索（GridSearch）[6]和k折交叉验证结合的方法。网格搜索是一种基于穷举的自动参数调优方法，首先对输入的不同超参数的候选列表值取笛卡尔积，获得超参数组，再使用超参数组训练模型并使用指定的模型评估方法（如基于macro F1-score的5折交叉验证）进行评估，最终选出评估结果最优的超参数组合。参数调优部分实现见6.4节**。**

## 五、实验环境与平台

**5.1 硬件环境**

* 处理器：Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz
* 内存：16.0 GB
* 硬盘：512 GB

**5.2 软件环境**

* 操作系统：Windows 11 家庭中文版 (21H2)
* IDE：Pycharm 2021.2.2 (Community Edition)
* Python版本：3.9.9

其中主要使用的包与版本如表5-1所示。

|  |  |
| --- | --- |
| Packet | Version |
| googletrans | 4.0.0rc |
| langdetect | 1.0.9 |
| nltk | 3.6.5 |
| numpy | 1.21.2 |
| pandas | 1.3.4 |
| sklearn | 0.0 |
| sklearn | 0.0 |

## 六、程序实现

**6.1 文本翻译**

基于对翻译语种、翻译质量等方面的考量，本实验的文本翻译基于开源的谷歌翻译API实现。文本翻译的核心代码如下：

1. **import** json
2. **from** googletrans **import** Translator
3. **from** langdetect **import** detect
4. **from** numpy **import** size
5. **from** nltk **import** tokenize
6. googletranslator = Translator(service\_urls=['translate.google.cn'])
7. **for** td **in** data:
8. **if** detect(td['content']) != 'en':  # 语言检测，非英语需要翻译
9. thislen = len(td['content'])
10. transed = ""  # 用于存放翻译后的新闻
11. sp = td['content'].split('\n')  # 首先按照段落分段
12. maxsp = list()
13. **for** spp **in** sp:
14. spoint = tokenize.sent\_tokenize(spp)  # 再进行分句
15. **for** sppp **in** spoint:
16. **if** len(sppp) > 5000:
17. **print**("分句长度大于5000！")  # googletrans限制5000字符
18. exit()
19. **else**:
20. **if** size(maxsp) == 0:
21. **if** len(spp) != 0:
22. maxsp.append(sppp)
23. **elif** len(sppp) + len(maxsp[-1]) < 5000: # 不大于5000字
24. **if** len(spp) != 0:
25. maxsp[-1] = maxsp[-1] + sppp
26. **else**:
27. maxsp.append(sppp)
28. **for** spp **in** maxsp:
29. transed += googletranslator.translate(spp, dest='en').text
30. td['content'] = transed

代码中对googletrans API的调用位于第30行。由于数据量较大，结合API限制，使用以下策略提高翻译速度与可靠性：

* 1. 使用国内服务器代替默认服务器，减小翻译时延，见代码第7行。
  2. 使用langdetect工具对新闻语言类型进行预检测，只对非英文语料执行翻译操作，减少不必要的翻译操作，见代码第9行。
  3. 由于googletrans API限制单词最大翻译字符数为5000，故采用先分段再分句，再将分句重组为不大于5000的新段落的方式，最大化减少API调用次数，见代码12~28行。由于翻译的最小单位应为分句，故对于大于5000的分句采用报错并退出的处理方式。值得说明的是，除去测试集中的81号数据为包含大量xml标签的脏数据外，训练集和测试集中不存在大于5000的分句。
  4. 考虑到googletrans服务的不稳定性，实现了“断点续传”功能。该部分实现并未在核心代码中展示，详见代码文件“translator.py”。

**6.2 数据增强**

本实验使用反向翻译的方法执行数据增强，对于标签为负的新闻文本，抽取其中20%的句子执行英译汉、汉译英的翻译并对原句进行替换，形成相同语义的不同表述，其余80%的句子保持不变，由此构成200条标签为负的“新样本”。结合对API调用频次、API响应速度及英汉互译质量的权衡，数据增强基于有道翻译API实现，核心代码如下：

1. **import** random
2. **from** youdaoAPI **import** YoudaoAPI
4. youdao = YoudaoAPI()
5. **for** i **in** range(0, 2200):
6. **if** data[i]['label'] == 0:
7. splitnews = tokenize.sent\_tokenize(data[i]['content']) # 分句
8. newnews = ''
9. **for** sent **in** splitnews:
10. **if** random.random() < 0.2:   # 随机替换20%的语料
11. newsent = youdao.connect(youdao.connect(sent))
12. newnews = newnews + newsent + '. '
13. **else**:
14. newnews = newnews + sent + '. '
15. **print**(data[i]['content'])
16. **print**(newnews)
17. newitem = {}
18. **print**(i)
19. newitem['content'] = newnews
20. newitem['label'] = 0
21. data.append(newitem)    # 将生成的新语料加入数据集结尾

代码中YoudaoAPI 为自定义类，详见“youdaoAPI.py”，API调用于代码第11行，其中connect函数返回参数对应的英汉互译结果。上述代码首先对原始数据集进行遍历，若是负样本则首先对样本新闻进行分句，之后以20%的概率替换该句子为反向翻译后的文本，见代码10,11行。之后再对替换或未替换的句子加上句点，重新拼接，形成新的负面新闻语料，达到数据增强的目的。

**6.3 文本处理**

* **分词**：文本分词可直接使用python 的split函数，将字符串默认按照空格、“\n”等空字符作为分隔符进行分词；还可以使nltk的tokenize中的word\_tokenize函数进行分词，会将单词分的更细，如“isn’t”被分为“is”和“n’t”，在处理新闻样本时，需要根据分词后进一步处理的需求选取分词方法。分词的实现如下：

1. **import** nltk.tokenize as tk
2. # nltk分词 #
3. splitnews = tk.word\_tokenize(datanews['content'])
4. # split分词 #
5. splitnews = datanews['content'].split()

* **去除数字、标点**：数字和标点符号可由python的string类中的punctuation和digits获得，由于数据集中存在少量全角字符，故单独添加一并去除。去除数字、标点的实现如下：

1. **import** string
2. remove\_char = string.punctuation + string.digits + "—“”‘’"
3. news = "".join([char.lower() **for** char **in** news **if** char **not** **in** remove\_char])

* **词形转换**：词性转换主要用于名词复数和动词时态的还原，但在文本处理中较难断定一个单词在语境中的词性，故实验中采取“试探转换”的思路，即分别将单词进行名词词形转换和动词词性转换，若出现任一结果与原单词不同，则进行单词替换。由于很少出现一个单词既是名词复数又是动词某时态的情况，故上述思路基本有效，词形转换实现如下：

1. **import** nltk.stem as ns
2. lemmatizer = ns.WordNetLemmatizer()  # 词形转换器
3. splitnews = [lemmatizer.lemmatize(word, pos='v') **if** lemmatizer.lemmatize(word, pos='v') != word **else** lemmatizer.lemmatize(word, pos='n') **for** word **in** splitnews]

* 词干提取：词干提取器有许多种，如波特词干提取器、思诺博词干提取器等。在新闻情感分类中，需要避免获得过于精简的词干，如波特词干提取器会将“his”和“hi”均提取为“hi”，会损失一定文本信息，故实验中选用效果相对较好的思诺博词干提取器，词干提取实现如下：

1. **import** nltk.stem.snowball as sb
2. sb\_stemmer = sb.SnowballStemmer("english")  # 思诺博词干提取器
3. splitnews = [sb\_stemmer.stem(word) **for** word **in** splitnews]
4. news = "".join(word + " " for word in splitnews if word != " ")

* 去除停用词：nltk中自带的停用词库较为简陋，仅收录了179个英语停用词，在实际使用中对分类效果的提升有限；实验中从互联网获取了一份规模相对更大的停用词库，其英语部分收录了570个停用词。去除停用词的实现如下。

1. splitnews = [word.lower **for** word **in** splitnews **if** word.lower() **not** **in** SWdict[lang]]

其中SWdict是已加载的多语言停用词库字典，停用词库中均为小写单词，故过滤时需要将单词转换为小写形式。

**6.4 基于Pipeline的模型实现**

Pipeline意为“流水线”，可以实现模型从数据预处理到模型训练、预测的流式管理，既有利于使代码思路清晰，也为GridSearch提供便捷[7]。由于各个模型基于Pipeline的实现大同小异，故本节以逻辑回归为例进行介绍。

以下代码是基于词袋法（CountVectorizer）、TF-IDF（TfidfTransformer）和逻辑回归模型（LogisticRegression）的Pipeline实现。

1. **from** sklearn.pipeline **import** Pipeline
2. **from** sklearn.feature\_extraction.text **import** CountVectorizer, TfidfTransformer
3. **from** sklearn.linear\_model **import** LogisticRegression
4. pl\_lr = Pipeline([(‘cv’, CountVectorizer(max\_features=8000)),
5. (‘tf’, TfidfTransformer()),
6. (‘clf’, LogisticRegression(C=1.5, class\_weight=’balanced’))])
7. pl\_lr.fit(train\_data, train\_label)
8. res = pl\_lr.predict(test\_data)

代码中的4,5,6行定义了一个Pipeline，训练数据和测试数据会依次经由CountVectorizer、TfidfTransformer进行预处理，最后由LogisticRegression执行训练或预测。代码的7,8行分别是基于训练集的模型训练和对测试数据的预测。

以下代码是Pipeline和GridSearch的结合使用。

1. **from** sklearn.model\_selection **import** train\_test\_split, GridSearchCV
2. **from** sklearn.pipeline **import** Pipeline
3. **from** sklearn.feature\_extraction.text **import** CountVectorizer, TfidfTransformer
4. **from** sklearn.linear\_model **import** LogisticRegression
5. pl\_lr = Pipeline([(‘cv’, CountVectorizer()),
6. (‘tf’, TfidfTransformer()),
7. (‘clf’, LogisticRegression(class\_weight=’balanced’))])
8. para\_grid = {
9. ‘cv\_\_max\_features’: np.arange(5000, 12000, 2000),
10. ‘clf\_\_C’: np.arange(0.7, 2.1, 0.2),
11. }
12. grid\_res = GridSearchCV(pl\_lr, para\_grid, scoring=’f1\_macro’, cv=5)
13. grid\_res.fit(train\_data, train\_label)
14. **print**(grid\_res.best\_params\_)
15. **print**(grid\_res.best\_score\_)

代码中第8行定义了GridSearch的参数表para\_gird，表明参数搜索范围。第12行定义了一个GridSearchCV的网格搜索对象，其搜索模型是5~7行定义的Pipeline pl\_lr，参数是para\_grid，同时指明模型评估指标为f1\_macro，使用5折交叉验证进行评估。

## 七、实验结果与分析

本节介绍实验中使用的不同模型、不同数据清洗方法的预测结果。若无特殊说明，实验中使用的模型参数为GridSearch获得的较优参数与手动调参的综合结果，预测结果中的macro-f1（交叉验证）为基于训练集的5折交叉验证的平均值，macro-f1（测试集）为头歌平台反馈的测试集预测结果。

**7.1 模型选择**

基于本文介绍的四种机器学习模型，使用词袋模型、TF-IDF算法进行文本向量化，在原始数据集上的预测结果如表7-1所示。

表 7‑1 不同机器学习模型的预测结果对比

|  |  |  |
| --- | --- | --- |
| 机器学习模型 | macro-f1（交叉验证） | macro-f1（测试集） |
| KNN | 0.5900 | 0.5985 |
| SVM（rbf核） | 0.6851 | 0.7331 |
| SVM（linear核） | 0.6583 | 0.6662 |
| 多项式朴素贝叶斯 | 0.6281 | 0.6161 |
| 伯努利朴素贝叶斯 | 0.6560 | 0.6907 |
| 逻辑回归 | 0.6987 | 0.7494 |

**分析：**由表7-1可知，逻辑回归模型对新闻情感的分类效果相对最好，故后续实验基于**逻辑回归**模型展开。

**7.2 文本翻译**

基于逻辑回归模型，使用翻译前后数据集的预测结果如表7-2所示。

表 7‑2 翻译前后的预测结果对比

|  |  |  |
| --- | --- | --- |
|  | macro-f1（交叉验证） | macro-f1（测试集） |
| 翻译前 | 0.6987 | 0.7494 |
| 翻译后 | 0.7042 | 0.7653 |
| 提升比例 | 0.55% | 1.6% |

**分析：**由表7-2可知，翻译后相比翻译前的预测结果有一定的提升，且在测试集上的提升效果更为明显，结果验证了文本翻译思路的正确性，其原因在于文本翻译将多国语言样本集中到英语文本这一个样本空间中，变相地“扩大”了训练集；此外，结果也侧面说明了谷歌翻译并未改变新闻文本的情感趋向。

**7.3 数据增强**

基于对每个负样本随机反向翻译20%分句的方法，获得了负样本增加量为200,400,600,800的增强样本，使用逻辑回归模型对不同增强样本的评估结果如图7-1、图7-2所示。

图 7‑1数据增强的测试集的评估结果

图 7‑2 数据增强的交叉验证的评估结果

**分析：**数据增强对训练集交叉验证的结果有显著的提升效果，其原因在于数据增强的新样本与原样本相似度达80%，在交叉验证中可能会将相似度较高的新旧样本分别划分到训练集和测试集中，因此交叉验证的评估结果“虚高”；测试集评估结果可以较为真实的反应数据增强的效果，负样本增强数量为400时在训练集上获得最高的预测结果0.7728，相比不增强提升0.75%

**7.4 文本处理**

文本处理基于翻译、负样本增强量为400的数据集开展，由于数据增强会给本地交叉验证带来评估结果虚高的问题，故本节实验结果不再展示交叉验证结果，仅展示测试集评估结果。实验中对3.3节介绍的文本处理方法的不同组合的效果进行评估，基于逻辑回归模型的对比结果如表7-3所示。

表 7‑3 不同文本处理方法在测试集上的优化效果

|  |  |  |
| --- | --- | --- |
| 使用的文本处理方法（组合） | macro-f1（测试集） | 优化效果 |
| 无 | 0.7728 | **/** |
| pd | 0.7731 | **+** |
| pd（不过滤“?”和“!”） | 0.7727 | **-** |
| lemma | 0.7722 | **-** |
| stem | 0.7611 | **--** |
| sw | 0.7695 | **-** |
| lemma+pd | 0.7691 | **-** |
| lemma+pd+sw | 0.7785 | **++** |
| stem+pd | 0.7679 | **-** |
| stem+pd+sw | 0.7605 | **--** |
| sw+pd | 0.7676 | **-** |
| lemma+pd+sw+stem | 0.7684 | **-** |

*注：pd：去除标点和数字，lemma：词形转换，stem：词干提取，sw：去除停用词。 优化效果中的“+”表示正优化，“-”表示负优化，符号数目表示优化程度。*

**分析：**整体而言，文本处理效果并不理想，在四种文本处理方法的单独使用中，只有去除标点符号（pd）是正优化，其余均为负优化。标点符号“?”“!”的保留并未获得更好结果，说明标点符号“?”“!”与新闻情感倾向关系较小。单独使用词形转换（lemma）对结果几乎无影响，而单独使用词干提取（stem）则会引起结果较大幅度的下降，说明词干提取并不适合新闻文本分析，即一个单词的不同派生词可能具有不同的情感表达。单独的停用词使用同样使结果不升反降，说明停用词噪声数据对新闻情感预测的关系不大，而停用词中的部分单词（如“not”等）反而与新闻情感有关。

值得注意的是，即便某种处理方法单独使用会造成负优化，该方法与其他处理方法的结合使用未必是负优化的线性叠加。如单独使用词形转换（lemma）会引起负优化，但词形转换（lemma）、去除标点和数字（pd）和去除停用词（sw）的结合使用则获得了实验中的最优结果0.7785。不同文本处理方法的之间的组合与预测结果之间的关系有待进一步的研究与思考。

## 八、总结与反思

**8.1 实验总结**

本次实验主要探索并尝试了多种自然语言文本预处理方法，并基于不同的机器学习模型进行训练、预测，最终获得了较好的结果。

基于多国语言的类别不平衡新闻样本，逻辑回归模型获得了比KNN、SVM和朴素贝叶斯更好的预测效果。在数据清洗中，翻译预处理对预测结果带来了较大幅度的提升，在测试集上的macro-f1提升了1.6%；数据增强也为预测结果带来了一定的提升，在测试集上的macro-f1提升了0.75%；但之前一度认为效果会较为明显的多种文本处理操作却并未获得较好的效果，停用词、词干提取、词形分析的单独使用均会带来负优化，单独过滤数字、标点也并未给结果带来很大的提升。多种处理方法的结合使用给实验结果带来了提高，最终在测试集上的macro-f1为0.7785。

**8.2 反思与体会**

这次机器学习课程设计的完成可谓“痛并快乐着”，也让我领略到机器学习的魅力所在。相比之前做过的人工智能导论课程项目、其他机器学习有关项目等，这次课设的关注重点不在模型本身，而在于使用模型之前对数据集的清洗。数据清洗的效果并不总是“线性累加”的，许多直觉上认为一定会有优化效果的方法，在做出来之后却会是负优化，再加上每次对数据处理过后都需要重新“调参”，实验过程就像是一次次“开盲盒”，经过漫长的数据清洗、网格搜索操作之后，反馈的结果往往会“出乎意料”。然而，这种实验体会反倒让我“有些上头”，每次获得最高结果的突破都会让我心花怒放。

何老师的课程让我收获颇丰，这次课程设计的完成过程也让我深刻体会到自己机器学习方面知识累积的欠缺，我相信机器学习这门选修课会成为我打开机器学习殿堂的“敲门砖”,它展现给我的只是冰山一角，其中更多的奥妙还在等待着我慢慢挖掘。

# 参考资料

[1] 何琨. 机器学习内部讲义. 2021.

[2] 简书博主-小叶Little\_Ye. [[Python]第三方模块：谷歌翻译接口—googletrans.](https://www.jianshu.com/p/ec86b571ee30) 2020.

[3] Amit Chaudhary. [A Visual Survey of Data Augmentation in NLP.](https://amitness.com/2020/05/data-augmentation-for-nlp/) 2020.

[4] 博客园博主-我的星空123. [02 NLTK 分句、分词、词干提取、词型还原.](https://www.cnblogs.com/wodexk/p/10292947.html) 2019.

[5] CSDN博主- Zen of Data Analysis. [机器学习-二分类SVC中的样本不均衡问题：重要参数class\_weight](https://blog.csdn.net/gracejpw/article/details/103054668). 2019.

[6] 简书博主-潇洒坤. [基于SVM、Pipeline、GridSearchCV的鸢尾花分类.](https://www.jianshu.com/p/6b5ef5afdf14) 2018.

[7] CSDN博主-五道口纳什. [sklearn 中的 Pipeline 机制.](https://blog.csdn.net/lanchunhui/article/details/50521648) 2016.