《机器学习》设计报告

|  |  |
| --- | --- |
| 学院 | 计算机与信息学院 |
| 专 业 | 电子信息工程 21级 |
| 学生姓名及学号  (排名不分先后） | 刘懿 2021212303 |
| 学生姓名及学号  (排名不分先后） | 陈起越 2021212048 |
| 学生姓名及学号  (排名不分先后） | 刘宁宁 2021212751 |
| 指导教师 | 吴乐 |
| 课题名称 | LLM - Detect AI Generated Text |
| 2023年12月31日 | |

目录

[一、任务概述 2](#_Toc154946470)

[1.1任务说明 2](#_Toc154946471)

[1.1.1任务要求 2](#_Toc154946472)

[1.1.2任务分析 2](#_Toc154946473)

[1.2任务数据集 3](#_Toc154946474)

[1.2.1 数据集概述 3](#_Toc154946475)

[1.2.2 数据说明 3](#_Toc154946476)

[二、数据处理 6](#_Toc154946477)

[三、任务模型 10](#_Toc154946478)

[3.1模型准备 10](#_Toc154946479)

[3.2模型撰写 15](#_Toc154946480)

[3.3模型训练 18](#_Toc154946481)

[四、结果分析 21](#_Toc154946482)

[4.1模型性能评估 21](#_Toc154946483)

[4.2模型优化思路 22](#_Toc154946484)

[4.3 模型最终得分（队伍排名） 23](#_Toc154946485)

[五、心得与体会 23](#_Toc154946486)

[六、附加题 24](#_Toc154946487)

[七、参考文献 27](#_Toc154946488)

# 一、任务概述

## 1.1任务说明

### 1.1.1任务要求

该比赛名称为 "LLM - Detect AI Generated Text"，其中 LLM 是指 large language model。这个比赛的目标是通过建立一个模型来检测 AI 生成的文本。随着自然语言处理和生成模型的快速发展，AI 生成的文本在互联网上越来越常见，而有时候这些文本可能被滥用或误导人们。因此，这个比赛旨在提供一种方法来区分 AI 生成的文本与真实人类生成的文本。

任务要求是建立一个模型，以识别哪些文章是由初中和高中学生撰写的，哪些是使用大型语言模型（LLM）撰写的。数据集包括约10,000篇文章，其中一部分由学生撰写，另一部分由LLM生成。比赛的目标是确定作文是否由LLM生成。训练集中有两个提示的作文，其中大部分是由学生撰写的，只有少数是生成的作文。测试集中有约9,000篇文章，既有学生撰写的，也有LLM生成的。参赛者需要使用训练集的样本来训练他们的模型，并根据测试集的数据来评估模型的性能。

### 1.1.2任务分析

该比赛旨在通过建立一个模型来检测 AI 生成的文本，来区分是由学生所撰写的文章还是使用大型语言模型（LLM）所生成的文章。

对此任务进行分析，若要建立一个性能很好的模型，我们需要进行以下操作：首先为了更好地理解数据集，并为后续的分析和处理提供基础。我们需要对数据集进行基本的探索，包括查找空缺数值、删除重复的训练数据以及绘制标签分布图和论文长度分布图等。之后考虑到是文本分类问题，需要将原始文本转换为机器学习算法可以处理的特征向量，即我们需要对文本进行分词化。然后根据任务的要求选择合适的模型，以及通过交叉验证等方式评估模型的性能，并对其进行不断优化。最后使用训练好的模型对测试数据进行预测，并将生成文本的预测概率添加到测试数据中，最终将结果保存为提交文件（submission.csv）。至此，便实现了任务要求。

经过查阅资料，对以上思路进行补充和具体化。即在数据集探索中，可以使用 Pandas 库读取数据集，并利用 describe() 函数、isnull() 函数、duplicated() 函数等进行基本的数据探索。可以使用 Matplotlib 绘制标签分布图和论文长度分布图。在将原始文本转换为机器学习算法可以处理的特征向量时，可以使用TF-IDF 算法将文本转换为向量表示。随后便进行模型选择并评估，最后进行预测和结果提交。

## 1.2任务数据集

### 1.2.1 数据集概述

数据集包含约10,000篇文章，其中一部分由学生撰写，一部分由各种大型语言模型（LLMs）生成。竞赛的目标是确定作文是否由LLM生成。所有的作文都是针对七个作文提示之一撰写的。在每个提示中，学生被要求阅读一篇或多篇资料文章，然后写出答复。当生成文章时，可能会将相同的信息作为LLM的输入，也可能不会。其中两个论文提示的论文组成了训练集；其余的论文组成了隐藏测试集。几乎所有的训练集文章都是由学生撰写的，只有少数生成的文章作为示例给出。

test\_essays.csv中的数据只是用来编写解决方案的虚拟数据。在评分时，这个示例测试数据将被完整的测试集替换。测试集中大约有9,000篇论文，既包括学生撰写的，也包括LLM生成的。

### 1.2.2 数据说明

竞赛网址提供的数据集包括四个数据文件：

分别是test\_essays.csv，train\_essays.csv，train\_prompts.csv，sample\_submission.csv，下面分别对这四个数据文件进行介绍：

（1）train\_essays.csv

数据信息共1378行，数据文件中包括以下四项：

id：每篇论文的唯一标识符；

prompt\_id：指示论文是针对哪个提示撰写的；

text：文章的文本内容；

generated：论文是由学生撰写（0）还是由LLM生成（1），其中学生写的为0，机器写的为1。查看文件后发现文件中学生所写的文章的数量远超机器所写的文章。

可以此这个文件描述的是一份包含学生和自动生成的作文的数据集。每个作文有一个唯一的标识符和一个对应的题目编号，作文文本也被包含在内。此外，该文件还包含了一个"generated"字段，用于区分每篇作文是由学生写出（0）还是由自然语言处理模型生成（1）。这个数据集可以用于训练文本分类模型，以助于后面利用该模型区分由学生写出的作文和由模型生成的作文。

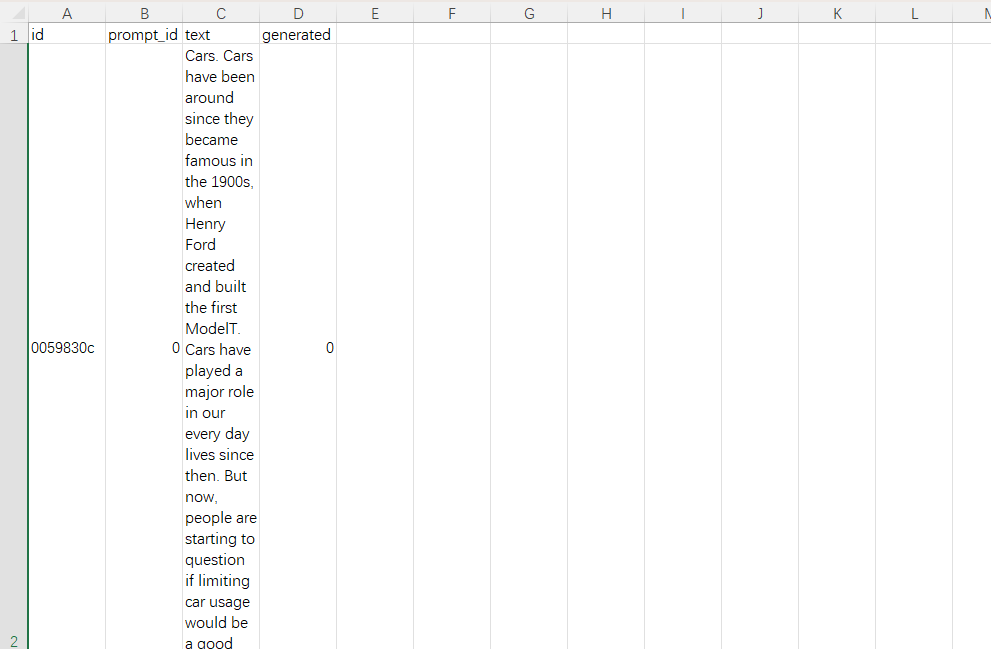


图1.2.1 train\_essays.csv数据文件

（2）test\_essays.csv

数据信息共3行，test\_essays.csv中的数据只是用来编写解决方案的虚拟数据。在评分时，这个示例测试数据将被完整的测试集替换。数据文件中包括以下三项：

id：每篇论文的唯一标识符；

prompt\_id：指示论文是针对哪个提示撰写的；

text：文章的文本内容；

此数据集文件主要用于检测和评估参赛者在检测人工智能生成文本方面的模型性能，即用于测试和评估模型，以确定模型是否能够正确地识别出由人类编写的文本和由人工智能生成的文本。这有助于提高模型对自动生成文本的检测准确性。

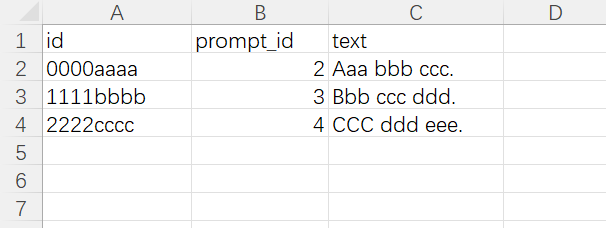


图1.2.2 test\_essays.csv数据文件

（3）train\_prompts.csv

数据信息共2行，数据文件中包括以下四项：

prompt\_id：每个提示的唯一标识符。

prompt\_name：提示的标题。

instructions：给学生的指导说明。

source\_text：以Markdown格式呈现的文章文本，这些论文是作为回应撰写的。

该数据文件的用途是为了给竞赛参与者提供写作任务的具体要求以及相关的文章来源，帮助其准备撰写文章。这些数据将被用于评估参赛者的表现，以确定其对于所提出任务的理解和回应能力。

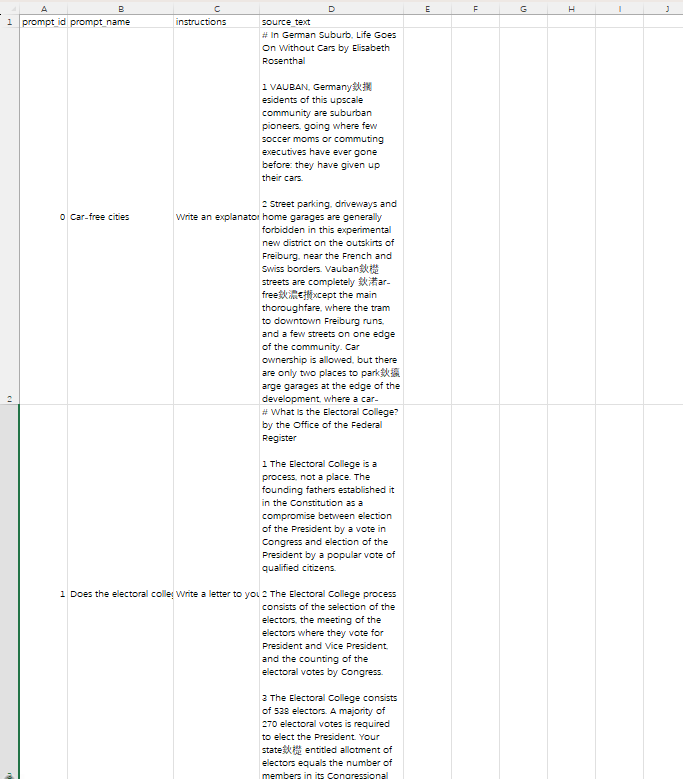


图1.2.3 train\_prompts.csv数据文件

（4）sample\_submission.csv

是样本提交文件，数据文件中包括以下两项：

id: 表示作文的唯一标识符。

generated: 表示模型判断该作文是由学生写出还是由人工智能生成的概率。

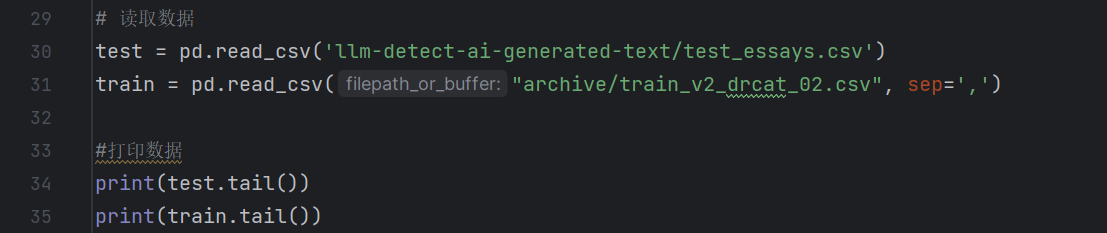
此文件的用于参赛者提交自己对测试集中每篇作文生成概率的预测结果。

# 二、数据处理

数据处理的目的是确保数据的质量，转换数据的形式，整合多个数据源，提取有用的特征，并分析数据的规律和趋势。通过清洗、转换、整合和分析原始数据，我们可以保证数据的准确性和可信度，将不同形式的数据转换为机器学习算法可处理的数值输入，整合多个数据源以获取更全面和准确的信息，提取与问题相关的有用特征，以建立准确且有意义的模型，最终探索数据中的规律和趋势，提高决策效率和精度[1]。

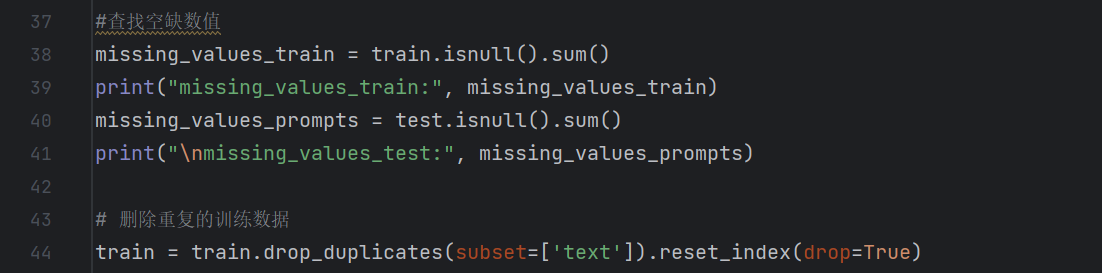
基于任务的要求，我们对于数据主要进行了以下处理：

1．读取并打印数据集中的数据



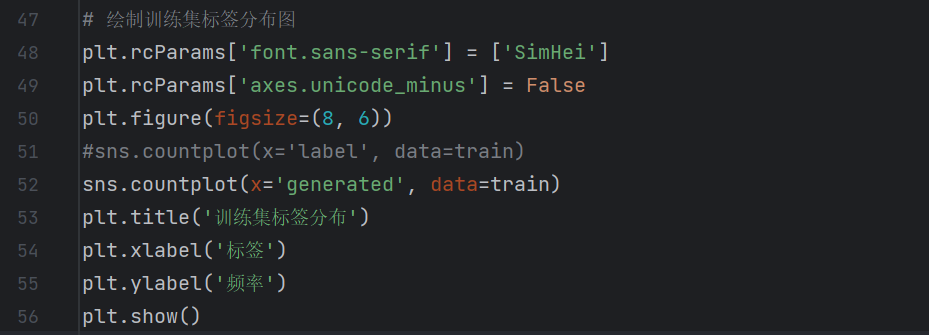
2．进行数据探索性分析

对数据进行初步的探索和分析，以了解数据的特征、结构和分布情况。此处我们在读取并打印了训练集和测试集的数据后，查找并统计了其中的空缺数值，并删除了其中重复的训练数据。



为了便于更好地了解数据集的分布情况、异常值情况等。我们随后绘制了训练集中标签的频率分布图和论文长度的分布图，以及学生和AI最常用词汇的统计图。

其中，通过绘制标签频率分布图，可以了解训练集中不同标签的数量分布情况。这对于我们理解数据的类别分布、平衡性以及样本的分布情况十分重要。在论文长度的分布图中，我们可以观察到是否存在过长或过短的论文。通过绘制分布图，便可以较为直观地发现数据集中的异常值。而绘制学生和AI最常用词汇的统计图可以帮助我们了解学生和AI之间使用词汇的差异。







3. 进行文本预处理

为了提取文本的语义信息，将其转化为机器学习算法可以处理的数值型输入，以及考虑到文本数据通常具有很高的维度，即文本中包含的不同词汇数量很大，需要减少特征空间的维度来使机器学习算法更容易对其处理，并且去除其中无用的信息，所以此处对训练集文本进行分词处理，将每个文本转换为一系列的词汇[2]。

此处利用Byte-Pair Encoding tokenizer对训练集文本进行分词处理。



4．特征提取

机器学习算法通常只能处理数值型输入，而无法直接处理文本数据。因此需要将文本数据转换为数字特征向量，使得机器学习算法能够有效地处理文本数据，以进行模型训练和预测。因此对文本数据进行特征提取。

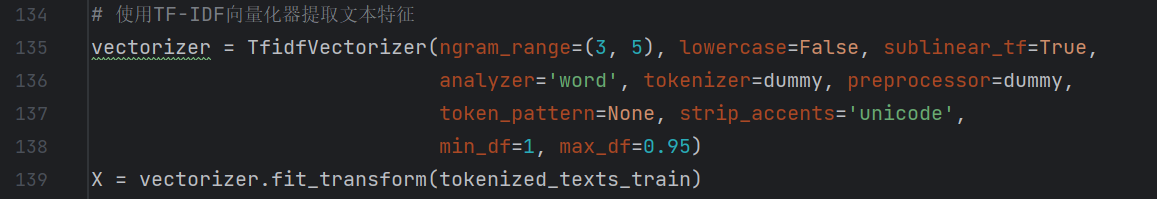
在查找可以使用什么方法来实现特征提取时，了解到有以下几个作用及优点：

（1）TF-IDF向量化器可以将每个词汇在文本中的重要性进行量化，并将其转换为数字特征向量。这样做的目的是保留每个词汇的信息，以便在后续的模型建立中发挥作用。并且重要的词汇会被赋予较高的权重，而无关紧要的词汇则会被赋予较低的权重。

（2）TF-IDF 向量化器可以将文本数据转换为相对较小的特征空间，从而减少特征空间的维度，提高模型的效率和性能。

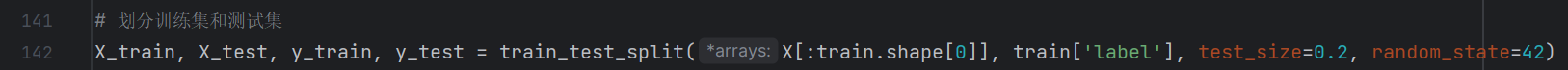
（3）TF-IDF 向量化器可以通过词频和逆文档频率来量化每个词汇的重要性。词频表示一个词汇在当前文本中的重要程度，而逆文档频率表示一个词汇在整个文本集合中的重要程度。这样，TF-IDF 向量化器可以帮助捕捉到文本的语义信息，以提供更具有区分性的特征向量。

因此我们最终选用的方法是使用TF-IDF向量化器将分词后的训练集文本转换为数字特征向量。



5. 数据集划分

将向量化后的训练集数据划分为训练集和验证集，用于模型的训练和性能评估。



通过以上五步，便实现了对原始数据的基本处理，通过对原始数据进行上述的一系列操作，保证了数据的准确性和可信度，获得了更全面和准确的信息，并且将不同形式的数据转换为机器学习算法可处理的数值输入等，方便后面直接用于模型使用，经过数据处理我们可以建立更加准确、有意义和实用的模型，并且提高决策效率和相应精度。

# 三、任务模型

## 3.1模型准备

3.1.1方案对比

在讲方案之前，先要明确检测人类写的和AI生成的文本之间到底需要解决一些什么问题。首先既然要区分，那么肯定是要找出两者的不同点，也即第一步特征提取，这就包括分析文本的结构、语法、词汇的使用以及在上下文连贯性等方面的问题。据搜集到的数据进行基本分析，并且在上文的数据预处理阶段我们对拿到的数据要有一定的分析与理解，所以总结来说有以下一些特征上的不同。

1. 语法和结构分析：

人写的通常来说大体符合一个正常人的正常思维，不会出现太大的逻辑上的错误，但是对于AI模型生成的文本就可能由于**训练样本的差异性**而导致出现一些逻辑上的错误。

1. 上下文的一致性

这个主要是AI在生成长文本的时候难以保持上下文的一致性，就是AL在生成**长文章的时候容易胡说八道，偏离主题**。

1. 词汇的选择和用法

AI生成的文本可能更加倾向于使用在**训练数据中频繁出现的词汇和短语**，也就是说对于词汇使用的多样性不高，但是人类的写作词汇更多样还可能更有创造性。

1. 情感与表达：

人类写作通常能够在表达情感时展现除更多的细腻和多样性，一般不太具有别扭的重复、排比语句。但是AI生成的文本可能在情感表达生就显得比较匮乏，而且对于一些特定的模式，**有固定的模板，**其实有时候人类甚至可以一眼辨认出来到底是人写的还是AI写的，因为都是固定模板的套用，**一句话包含的信息量不大。而且AI比较擅长总结，特别喜欢在最后进行一段话的总结[3]。**

1. 文本的长度和连贯性：

在生成较长的文本时，AI模型可能由于记忆限制而在文本的长期依赖关系上表现出一定的偏差，其次就是AI擅长生成的文本一般都比较短小。而人类写作能够很好的保持文本的连贯性。

1. 特定领域的专业知识：

人类的写作涉及到特定领域的专业知识，而AI生成的文本可能在理解和表达特定领域的复杂概念存在局限性。

下面介绍一些目前比较常见的处理检测文本任务的方法。

1. 基于统计的方法：

这种方法使用文本的统计特征，分析文本的属性，如词汇、语法结构、句子结构等，可以帮助识别自动化生成的文本。例如，某些生成文本可能倾向于使用特定的短语或结构。等来区分人工生成的文本和人类写的文本。

1. 深度学习模型：

基于深度学习的方法在这个任务上取得了显著的进展。递归神经网络（RNN）、长短时记忆网络（LSTM）、变压器（Transformer）等模型可以用于**捕捉文本的长期依赖关系和语义信息**。

1. 对抗性学习：

对抗生成网络GANS（Generative Adversarial Networks GANs）是一类用于生成模型的深度学习框架，主要的思想就是训练两个神经网络，一个是生成器，一个是**判决器**，让他们相互对抗，达到生成逼真样本的目的，那么这里面的**判决器的任务就是区分生成器生成的样本和真实训练数据的作用。**

3.1.2模型原理

所以对于以上的步骤，我们的模型进行了一个整合。

**第一，数据认知。**我们对于拿到的数据首先进行了一些基本的分析，比如词频，并且打印出来，让我们对拿到的数据有一些基本的认知。

**第二，**我们既然要提取文本的特征，那么最常用以及直接的方法就是先**训练一个分词器**（tokenizer）。因为通过这个分词器，我们可以将文本划分成更小的子词或者标记，以便更好的捕捉语言的复杂性和灵活性，我们知道在现实世界中，语言不单是子词，字母，还有很多emojy，这就是为什么有些文本分类准确率低，因为不能够很好的处理这些非字母的符号，那么对于使用基于Byte-Pair Encoding(BPE)算法，针对的单位是一个字母，从而创建的一个自定义的分词器，能够很好的解决这个问题。

BPE(Byte-Pair Encoding)是一种用于数据压缩和自然语言处理的无监督分词算法[4]。主要的思想就是从数据中发现并且学习出现频率高的子词单元（即byte-pair），然后将这些子词单元合并成一个更大的单词，可以一直迭代，直到某个预定义的词汇大小或者某个停止条件。

下面举个简单的例子来加深对这个**训练算法**理解，因为这个算法是训练分析器比较核心的部分。

所以比方说，对于下面这些词：

“hug”，“pug”，“pun”，“bun”，“hugs”

他们在语料库中出现的频率是不一样的，那么比方说：

("hug", 10), ("pug", 5), ("pun", 12), ("bun", 4), ("hugs", 5)

括号里面的后一项是在语料库中出现的次数。然后我们给每个词切片。得到这样的结果：

("h" "u" "g", 10), ("p" "u" "g", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "u" "g" "s", 5)

我们来看看切片后词组搭配，比方说（“h”，“u”）出现的次数，在hug里面出现10次，在hugs里面出现5次，所以我们加起来，变成总共是出现15次，（“hu”，15）。但是，这个词汇组合不是这次分词组合中出现最多的，出现最多的应该是“ug”，总共出现了20次，在hug中的10次，pug中5次，hugs中的5次，所以加起来就有20次，那么我们将最初建立的词汇表以及更新语料库，以下形式展现结果：

Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug"]

Corpus: ("h" "ug", 10), ("p" "ug", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "ug" "s", 5)

就是第一次算法的结果，接下来进行下一次迭代。组合（“u”，“n”）总共出现了16次，所以有如下结果。

Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug", "un"]

Corpus: ("h" "ug", 10), ("p" "ug", 5), ("p" "un", 12), ("b" "un", 4), ("h" "ug" "s", 5)

接下来再次迭代，我们发现出现频率最高的是hug，更新结果，这个时候我们就已经得到了第一个三个字母的单词。

Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug", "un", "hug"]

Corpus: ("hug", 10), ("p" "ug", 5), ("p" "un", 12), ("b" "un", 4), ("hug" "s", 5)

接下来的迭代继续，一直达到我们想要的词汇表的大小。

紧接着讲一下**标记算法**，因为这是**紧随训练算法**的，通过应用以下这些步骤对新输入进行标记：

1、归一化

2、提前标记

3、切片

4、训练

因此对于一个新输入的单词比方说bug，就会被提前标记成【“UNK”，“ug”】为什么呢？因为m不在最初的词汇表中就是上面的vocabulary中。接下来进行训练的迭代。

**第三**，我们在创建好了分词器之后可以**提取文本的特征**了，提取文本的特征我们采用的是TF-IDF Vectorization（TF-IDF向量化），利用这个算法可以考虑到单词在文档中的频率以及它在整个语料中的重要性，这种向量表示能够捕捉文本中关键的重要此，从而得到一个权重，作为机器学习模型的输入特征。

下面解释，什么是TF, Term Frequency，表示一个词在文档中**出现的频率**，计算公式为：

公式（3-1）

IDF用于衡量一个词在整个文档集合中的信息量的度量，计算公式如下:

) 公式（3-2）

TF-IDF Weight:就是上面两个式子的乘积，用来表征一个词的重要程度，计算公式为：

公式（3-3）

**第四**，在我们进行特征提取结束之后，就可以把他当作一个分类任务来解决。所以接下来我们训练一个**集成分类器，并且设置的投票策略是软投票**，也就是每个基础分类器给出的是一个概率统计，最终的分类结果是所有基础分类器概率的加权平均。权重通常根据分类器在验证集上的准确度来分类的，所以准确度较高的我们给的权重就更多一些。

我们选取的模型有**逻辑回归Logistic Regression**, **多项式朴素贝叶斯Multinomial Naïve Bayes**、以及支持向量积Support Vector Machine，这些都是比较常见的分类模型，并且也能够在sklearn这个库里面直接调用。

简单介绍一下三个模型的优劣：

逻辑回归就是简单，它是一种线性的分类器，易于理解和实现，可以输出样本属于每个类别的概率，不仅仅是最终的类别标签，而且在数据量不是很大的情况下，不容易过度拟合。这就是我们这次选择他的一个原因。

多项式朴素贝叶斯的特点是在高维数据和文本分类上的表现比较好，其次呢就是在小规模数据集上的表现好，因为我们这次训练的数据集样本容量不是很大。

支持向量积能够处理非线性决策边界，通过核函数可以映射到高维空间，对于高维空间和复杂问题的分类表现良好[5]。

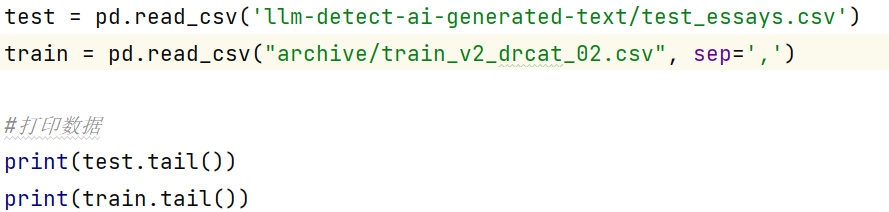
使用 cross\_val\_score 对集成分类器进行了 5 折交叉验证，使用准确度（accuracy）作为评价指标，计算了交叉验证的平均准确度。

所以总结一下我们模型的搭建思路，就是首先拉取数据进行一个可视化分析，有一个大致的认知。其次我们进行分词，使用的是BPE算法，接着我们用TF-IDF算法进行特征的提取。对于分类我们采取集成分类器，初始化了一个投票器。 在整个训练集上进行训练，然后通过测试机输出查看准确度。

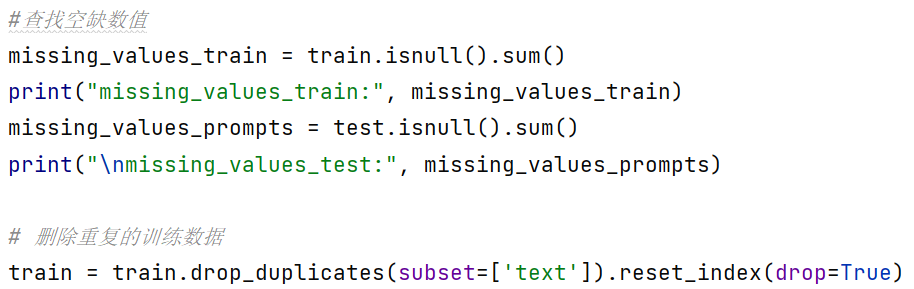
这里还要解释一下，因为我们在前期对于数据整体分析的时候发现，整个训练数据是非常不具有普遍性的，也就是说整个train大概一千五百个文章里面只有3个是机器生成的文章，所以如果用这个训练很容易出现过拟合的现象，因为没有足够的数据去分析机器生成文本的特征。所以我们使用的是train\_v2\_drcat\_02.csv数据集来训练，它同样也是用于训练检测机器生成和人类撰写文章的数据集。

## 3.2模型撰写

1、读取数据及打印



2、预处理数据

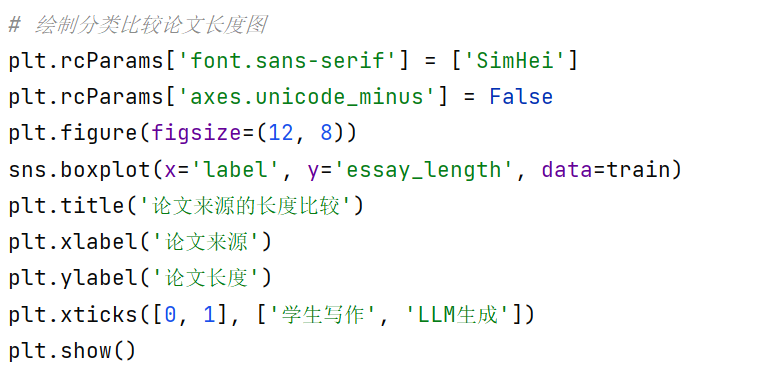


3、对数据的初步认知

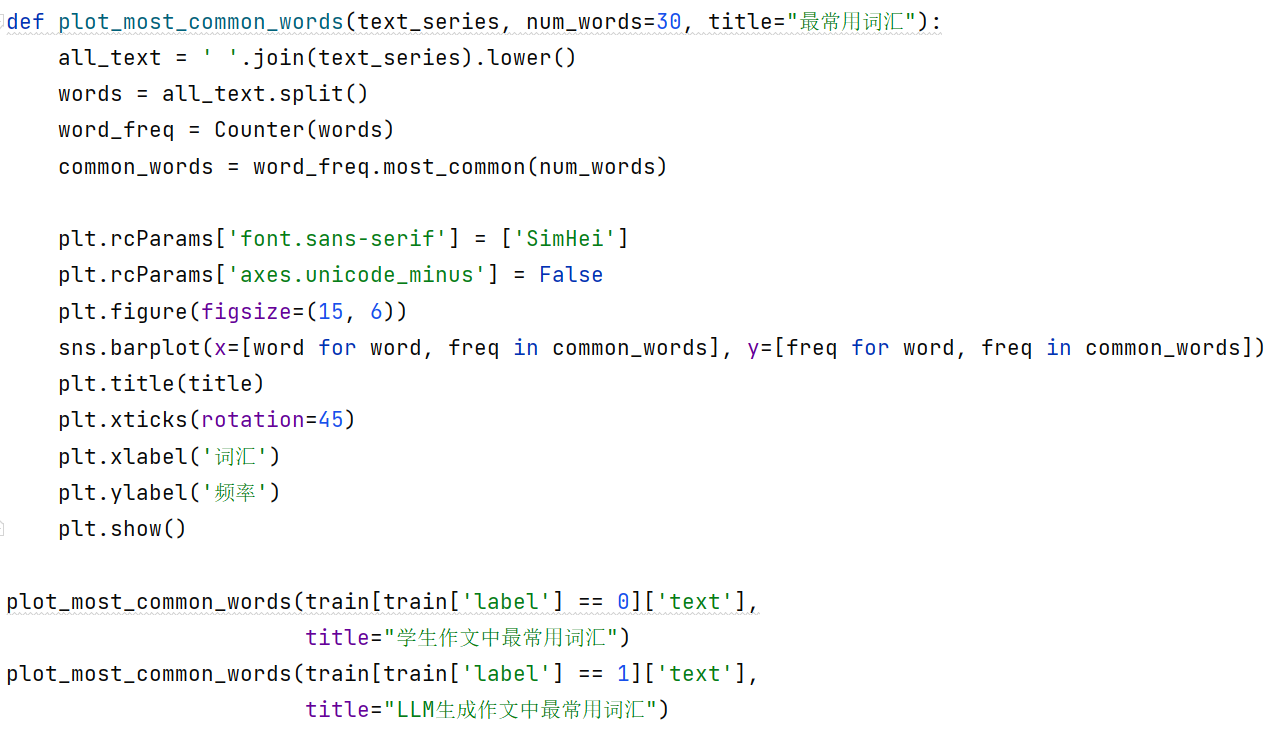
查看一下模型的分类，有哪些是学生写的，那些是机器写的



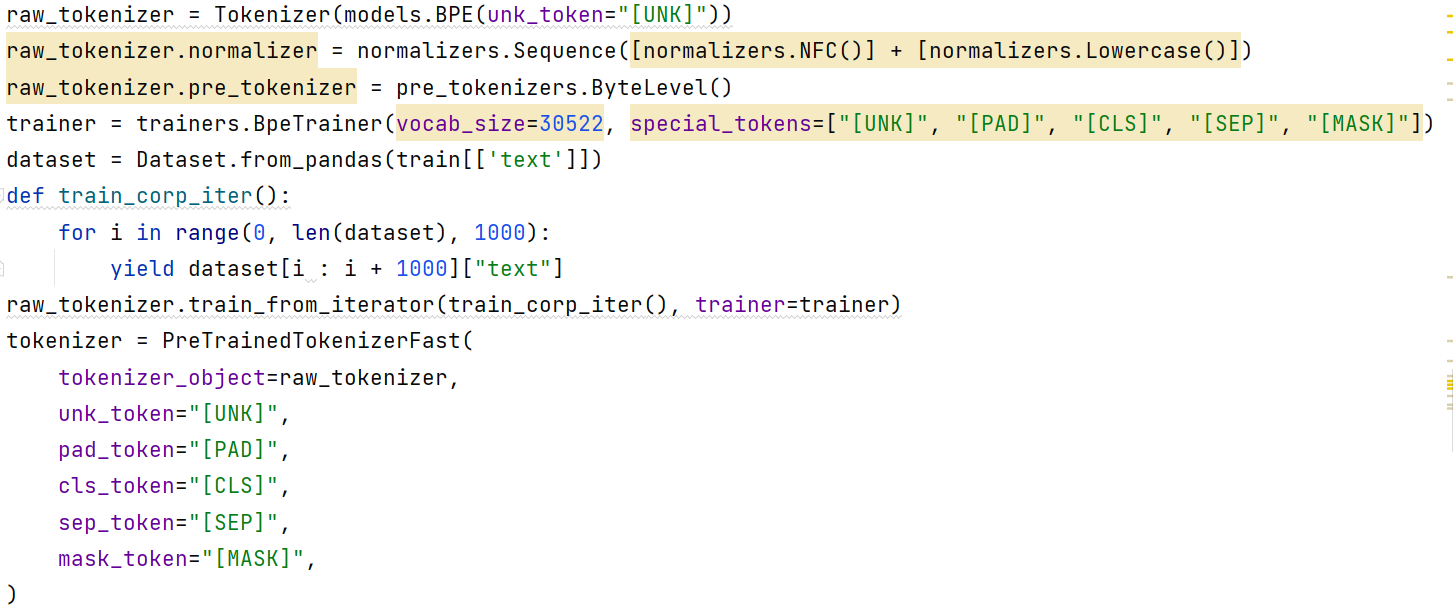
查看两类论文的长度，并将其可视化



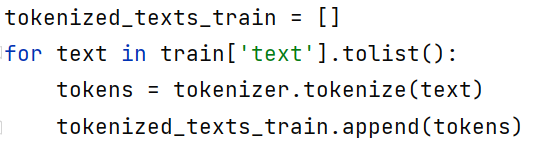
查看一下两类论文的词频



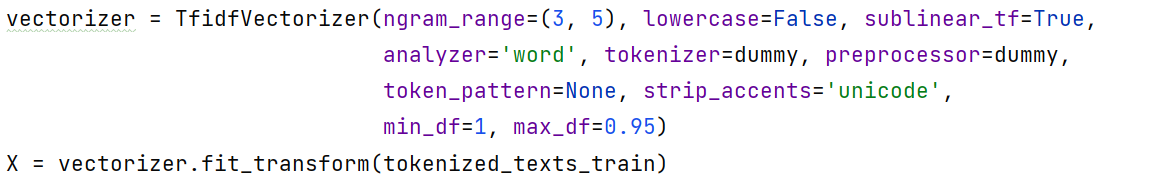
4、创建分词器



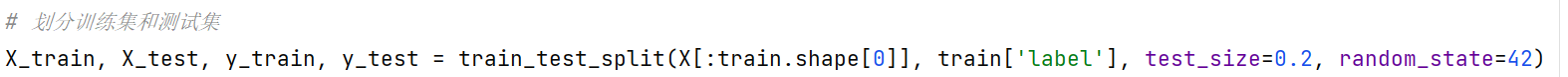
4.1分词化训练集文本



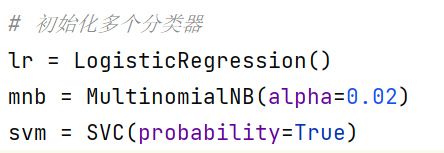
5、使用TF-IDF向量器提取特征



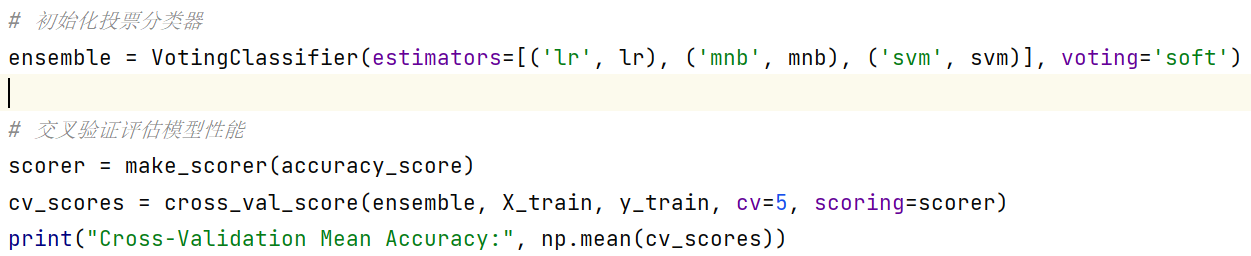
6、划分训练集与测试集



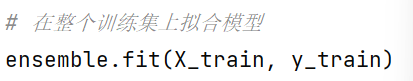
7、初始化多个分类器



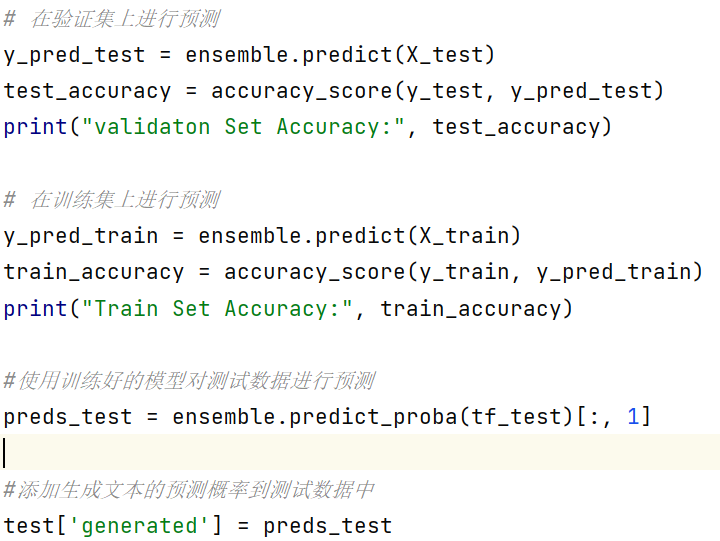
8、分类器性能评估



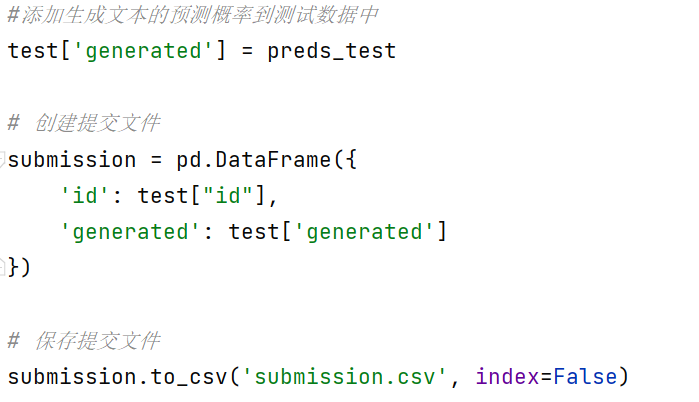
9、整个训练模型训练集



10、测试集进行预测输出



10、保留预测结果到sub文件



## 3.3模型训练

1、读取数据

一共是test的三条数据。比赛给出的train数据1378条。自己的训练数据集train\_v2\_drcat\_02一共有44868条

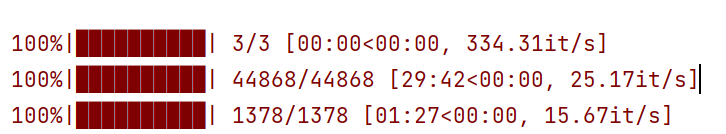
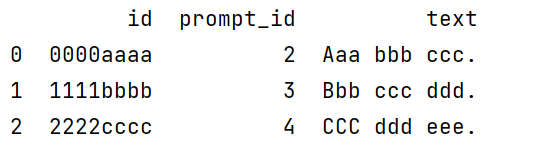


图3.3.1 读取数据

可视化一下数据集



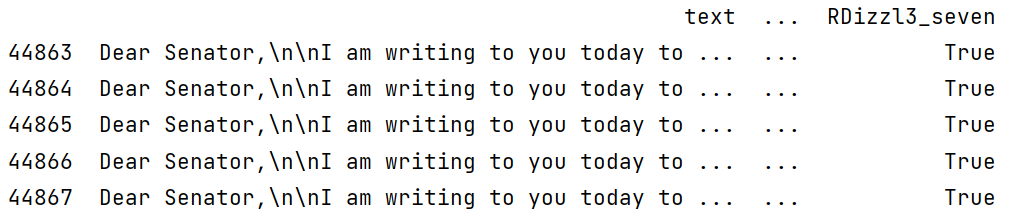


图3.3.2 可视化数据集

打印空缺值信息

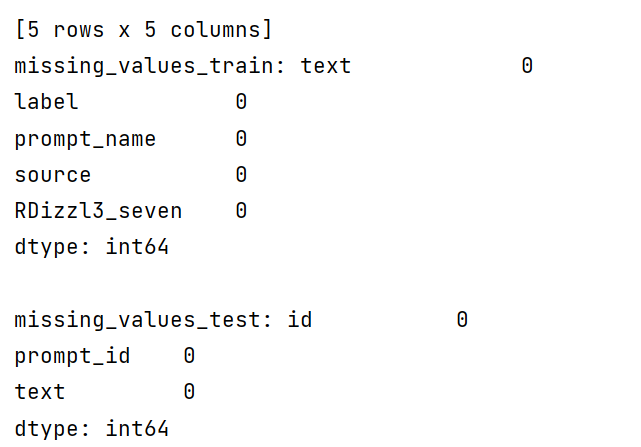


图3.3.3 打印空缺值信息

训练集的标签，0表示的是人类写的，1表示的是AI生成的

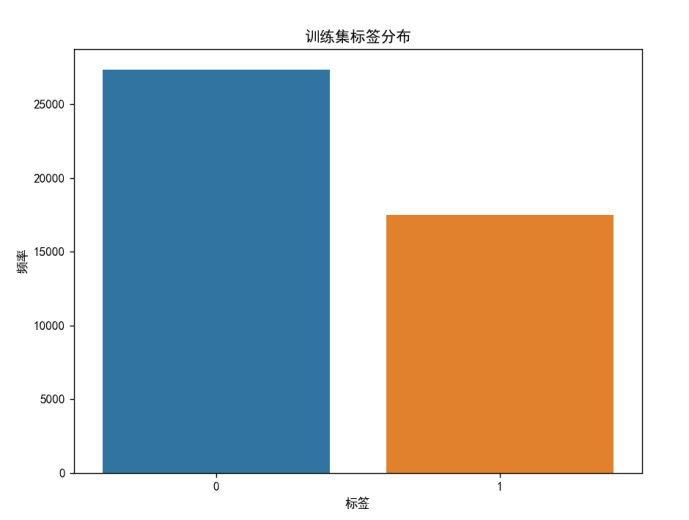


图3.3.4 训练集标签分布

论文的长度，对比你能发现就是学生的论文长度的范围是比机器生成的要大的，其次就是对于字符长度没有那么集中。

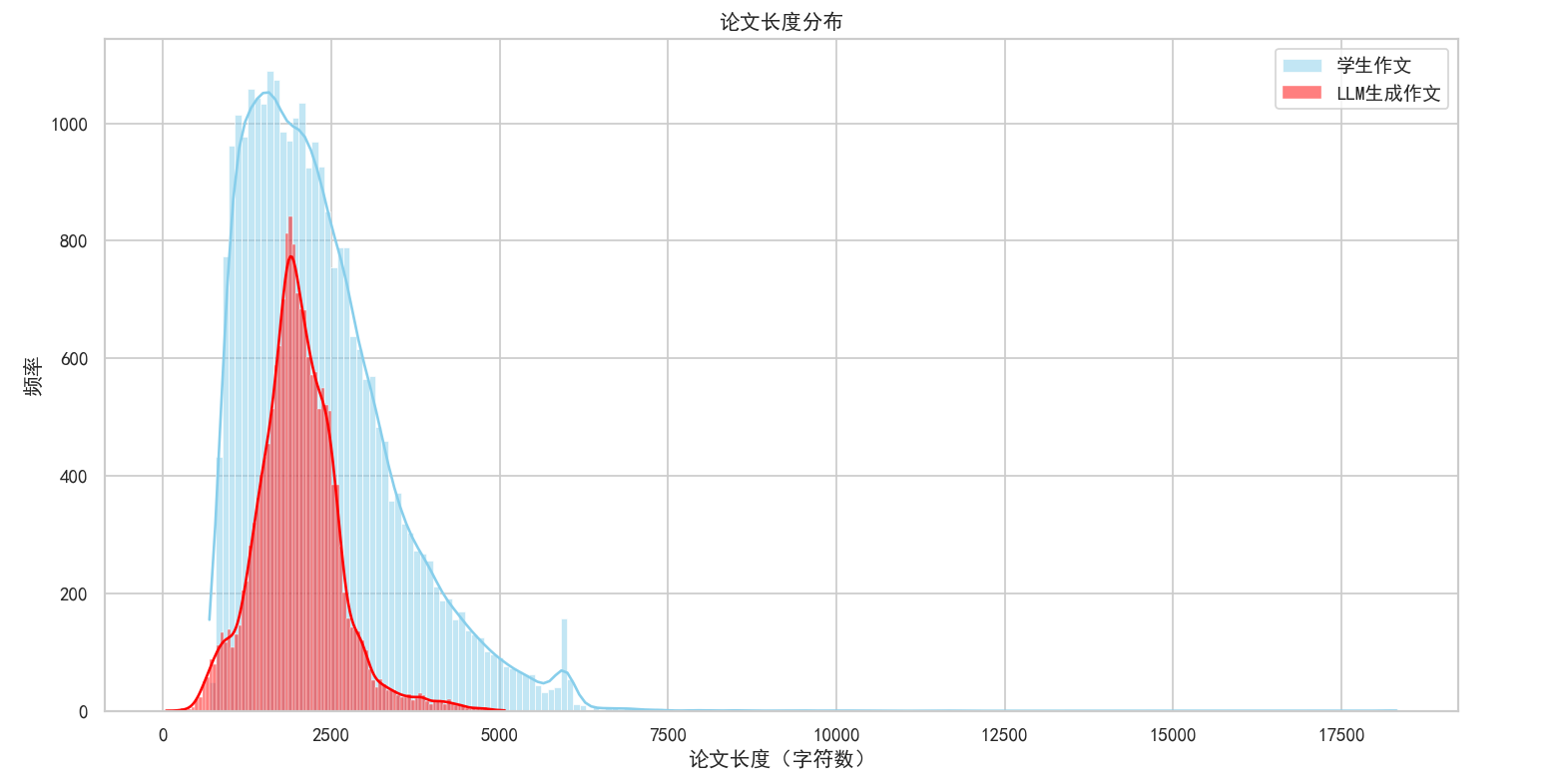


图3.3.5 论文长度分布

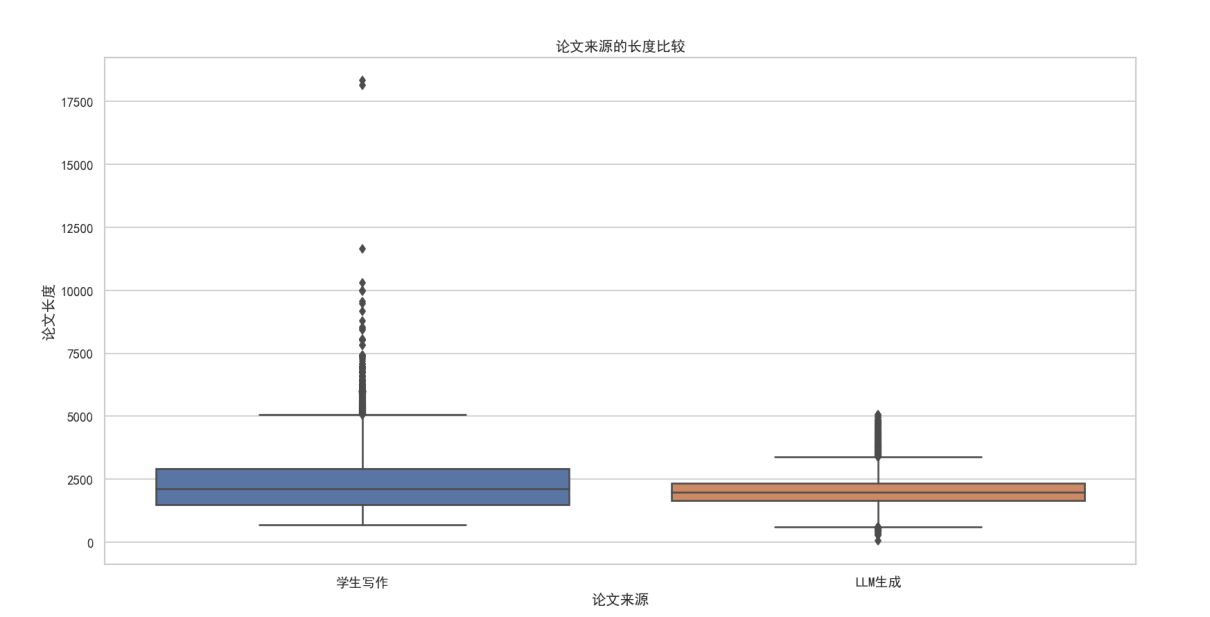


图3.3.6 论文来源的长度比较

对比词频得出的结论是机器生成的有几个非常集中使用单词。

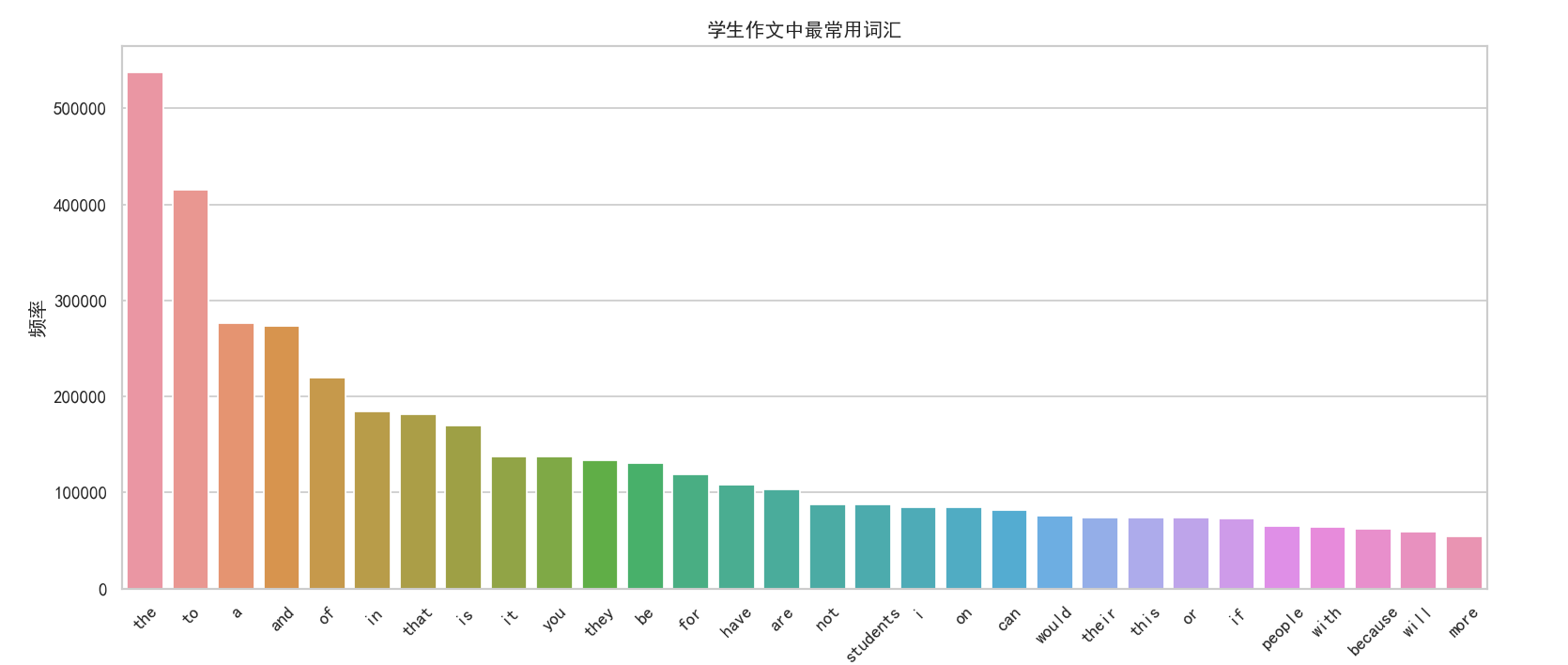


图3.3.7 学生作文中的最常用词汇

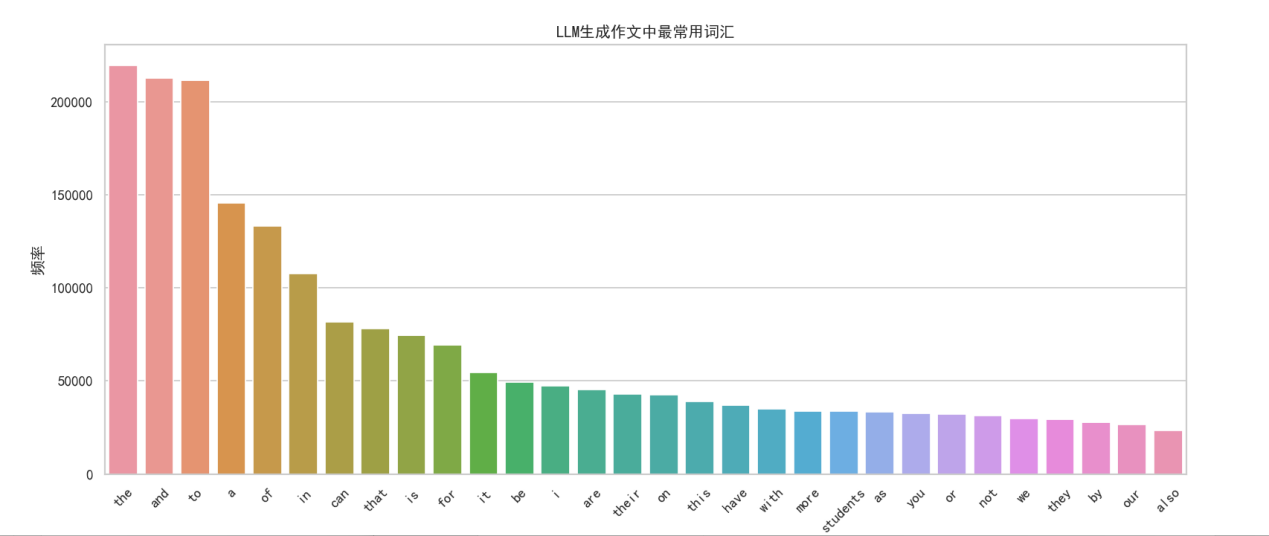
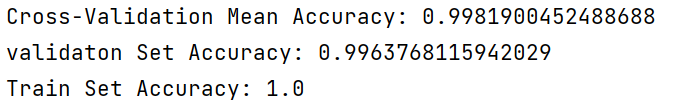


图3.3.8 LLM生成作文中最常用的词汇

分词器结果可视化：

{'ĠAaa Ġbbb Ġccc': 0, 'Ġbbb Ġccc .': 6, 'ĠAaa Ġbbb Ġccc .': 1, 'ĠBbb Ġccc Ġddd': 2, 'Ġccc Ġddd .': 7, 'ĠBbb Ġccc Ġddd .': 3, 'ĠCCC Ġddd Ġeee': 4, 'Ġddd Ġeee .': 8, 'ĠCCC Ġddd Ġeee .': 5}

最终输出的结果：



# 四、结果分析

## 4.1模型性能评估

1、交叉验证平均准确度（0.9981）：

这个分数是通过使用投票分类器（VotingClassifier）在训练数据的不同子集上执行交叉验证计算得到的。平均准确度非常高，表明模型在各个训练子集上都表现出色。

2、验证集准确度（0.9964）：

这是模型在验证集上的准确度，即模型在未见过的数据上的性能。高验证集准确度表明模型能够很好地泛化到新数据，这是一个积极的迹象。

3、训练集准确度（1.0）：

训练集准确度为1.0，但不排除这是过度过拟合可能，即模型可能在训练数据中过于拟合，而在新数据上的性能可能较差。

## 4.2模型优化思路

我们的基本任务是完成了，也能输出一个预测值。在接触这个项目的一开始，就是开始队数据的处理的时候我们就发现给的训练数据集是非常不友好的，首先就是test只有三条数据，train虽然有一千三百多条数据，但是只有3条是机器生成的，如果用这样的不均匀而且数据量不大的数据训练，模型没有捕捉到足够的特征信息，就会很容易过度拟合。所以我们采取了拉取一个更大的数据集进行训练，然后再结合上给的数据集。

其次就是分类我们使用的是集成分类器，逻辑回归能够很好的二分类，多项式朴素贝叶斯主要是再文本分类的领域表现得很出色，SVM主要能够适应不同类型，就是异值很好的处理。通过集成多个模型的预测结果，能够降低单个模型的不稳定性[6]。但是这里也有一个比较明显的缺点，就是集成的方法队数据的小波动不敏感，特别是数据集比较小的时候。

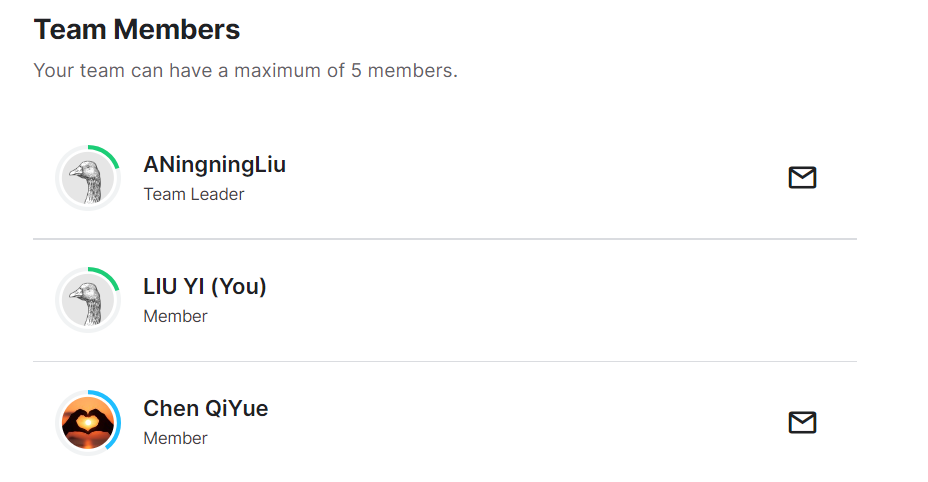
改进的方案是，在参考了一篇论文后认为一般的模型主要的检测思想都是训练一个分类器，也就是另一个模型，但是如果采用一个叫做DetectGPT的方法有可能比分类器更好的检测出是否是机器生成还是人写的文本。因为一般分类模型的弊端是可能出现对数据的过拟合以及对于不同类型的文本，要重新生成不同的模型[7]。但是DetectGPT不会有这样的问题，它不需要大量数据的认知。它只需要与自己的源文本进行对比，而且可以适用于不同的领域。

DetectGPT的核心思想是基于实验发现的对于AI生成的文本和人类写的文本有一个很明显的区别，就是人类写的文本的扰动性更小，意思就是相对于人类重写的文本，让模型重新生成的一样意思的文本后求解与源文本的对数函数会有明显位于负曲率区域的趋势，但是，人类写的文本不会有这样明显的趋势，这个对数概率可能大也可能小。

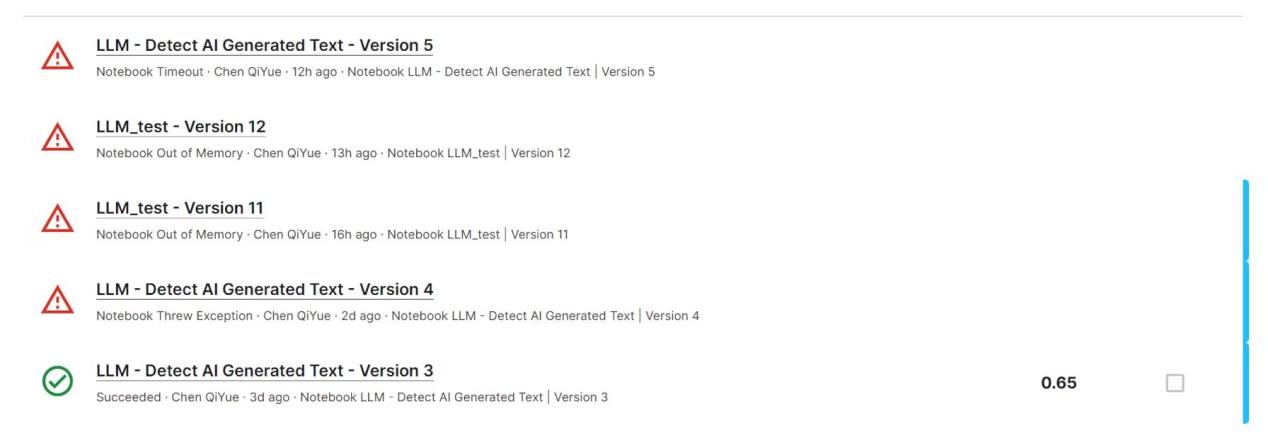
一般来说，对于一定的文本，让人或者机器去重新写不太现实，所以这个模型的基本思想就是使用T5这个模型，将文本进行一部分信息的掩盖，然后与原本文本进行对数概率的对比，如果有明显的更低的趋势，那么他有很大的可能是AI生成的文本，而且经过测试，这种模型的效果很不错。因为时间关系，这个模型我们目前还没有办法复现，所以如果时间充裕，这将是接下来对于检测AL生成和人类写的文本主要设计方向。

## 4.3 模型最终得分（队伍排名）

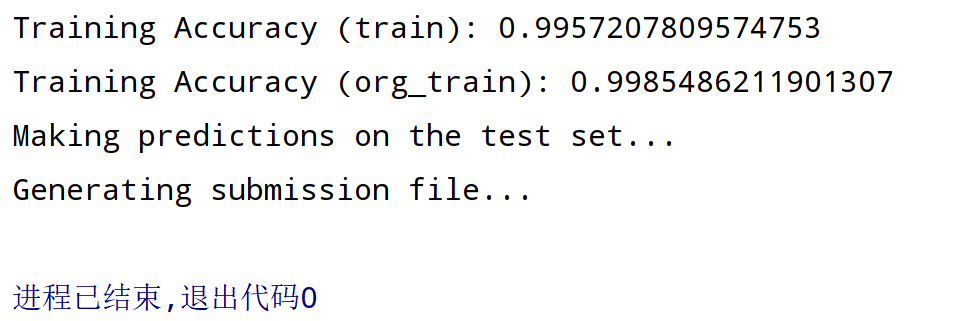
**队伍建立：**



**队伍得分：**



**备注：**目前新版本的代码在kaggle平台上运行12h后报错：Timeout。而在pycharm上运行时长约1h，现将pycharm运行结果展示如下。(代码文件见main4.0)



# 五、心得与体会

在选题时，我们看到本题就十分感兴趣。近年来，以ChatGPT为代表的生成式人工智能已经改变了我们生活的方方面面。但是这也随之而来引发了一些问题，比如当今论文的撰写、作文竞赛等领域十分注重文字的原创性，这就需要一种算法能够辨别AI写的内容和人类原创的内容。我们的课设科研都离不开chatGPT的协助，因此我们打算研究这个和我们很贴近的任务。

结合我们在本课程《机器学习》的学习内容，我们首先锁定两种方法：以K-Means为代表的非监督学习聚类算法、以TextCNN为代表的监督学习词分类算法。K-Means聚类算法可以采用我们在鸢尾花分类的实验的方法，首先采用PCA算法或者其他算法将文本数据降维，接着采用K均值或者其他方法将数据聚集成两类，输出最终判断结果。词分类算法可以采用我们TextCNN实验的方法，首先采用如word2Vec等模型获取预训练的词向量。接着，利用torchtext库进行词向量随机初始化。使用textCNN、SVM、逻辑回归模型或者其它监督学习机器学习模型进行文本分类，最后输出预测结果，对模型进行评估和优化。  
 接着我们在CSDN和GitHub等平台上了解了应用更广泛的文本分类方法，最后锁定了词分类方法。接下来我们就进入了代码编写阶段，第一步，我们使用Byte-Pair Encoding进行分词处理，并创建了一个用于处理文本的Tokenizer。第二步，我们选择了TF-IDF向量化器作为特征向量生成器，使用逻辑回归、向量机、贝叶斯分类器作为模型进行词分类，最后使用投票分类器进行模型集成。最后，我们通过交叉验证模型的性能，在整个训练集上拟合了模型，并在测试集上进行了预测。  
 经过这次的项目经历，我们最大的收获是更好的适应了团队合作模式，通过合理分配任务，实时沟通，来避免发生成员都只会自己负责的部分，导致代码不匹配。我们还充分运用了本课程的理论知识，不同于只是找到别人写的代码然后拼拼凑凑，学习了机器学习这门课程之后，我们可以清晰的理解别人的机器学习代码的逻辑，然后在此基础上改进，把别人的代码变成自己的代码，实现了机器学习入门。未来如果研究生或者工作选择了AI方向，接下来的机器学习进阶，一定会有更大的挑战等待着我们，但是我们有信心可以像这门课一样一步一个脚印，到达我们的目的地。

# 六、附加题

在调查了不同分类模型的应用后我们发现，K-Means算法适用于无监督学习的聚类任务，投票分类器集成的模型适用于需要更高准确率的监督学习分类任务，尤其是在数据关系较为复杂时。开始我们使用的是投票分类器集成的模型，这次我们采用K-Means算法进行聚类。

首先，我们通过加载训练集、测试集和相关提示数据，建立了数据框。接着，进行了文本预处理，包括将文章文本转换为小写、去除标点、词干提取等等。随后，我们使用 TF-IDF 向量化技术将文本数据转换为数值特征表示。通过轮廓系数确定了 KMeans 聚类的最佳簇数量，并在训练集上应用 KMeans 算法进行聚类。通过 PCA 将 TF-IDF 特征降维到 2 维，以便可视化聚类结果。最后，评估了在训练集和测试集上的聚类准确率，并使用训练好的 KMeans 模型对测试集进行了聚类，生成了提交文件。(代码文件见main3.0)

部分核心代码如下图所示：

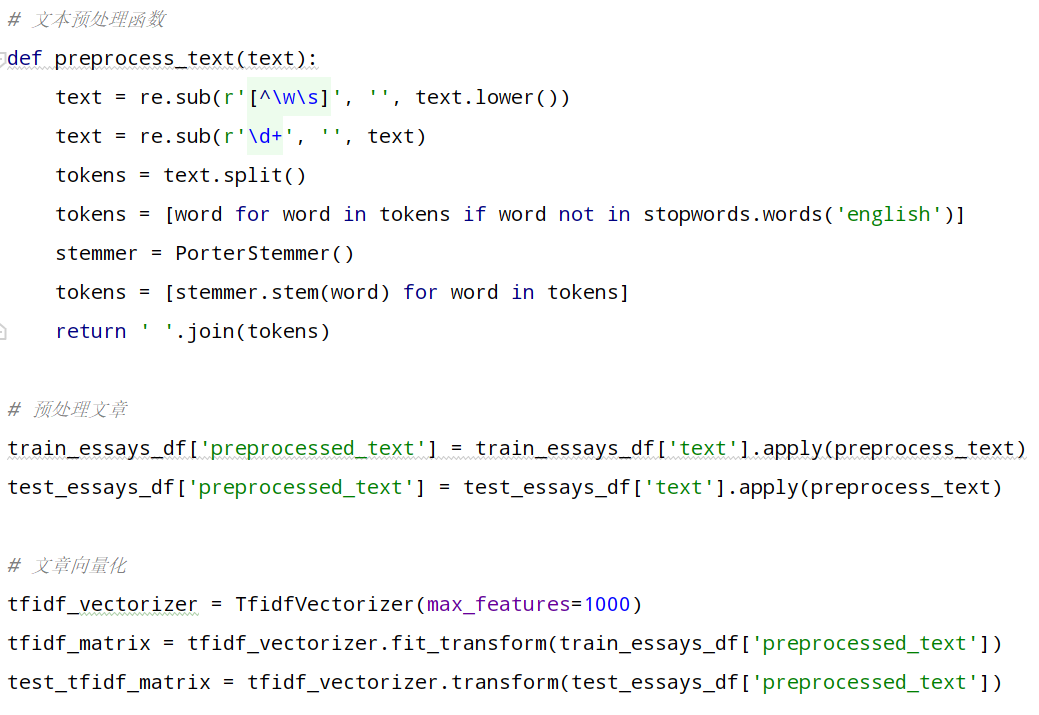


图6.1 文本预处理代码

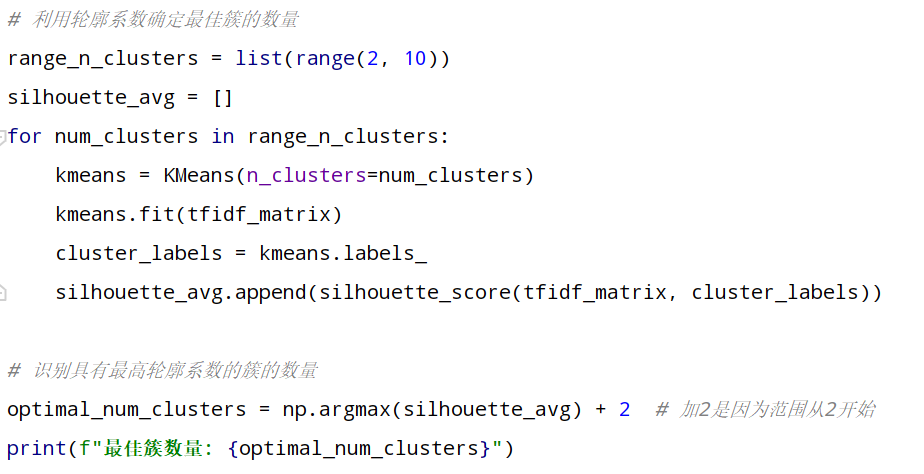


图6.2 确定最佳簇数代码

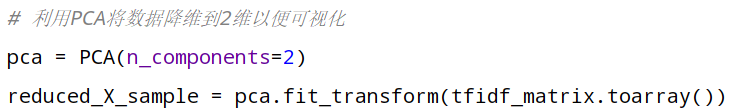


图6.3 使用PCA进行降维

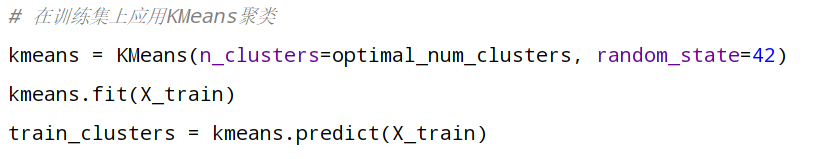


图6.4 使用KMeans进行聚类

我们还在代码中进行了必要的可视化输出，我们绘制了不同簇数量的轮廓系数图、文章的PCA降维图，最后载训练集和测试集上评估模型准确率。最终，代码运行结果如下图所示：

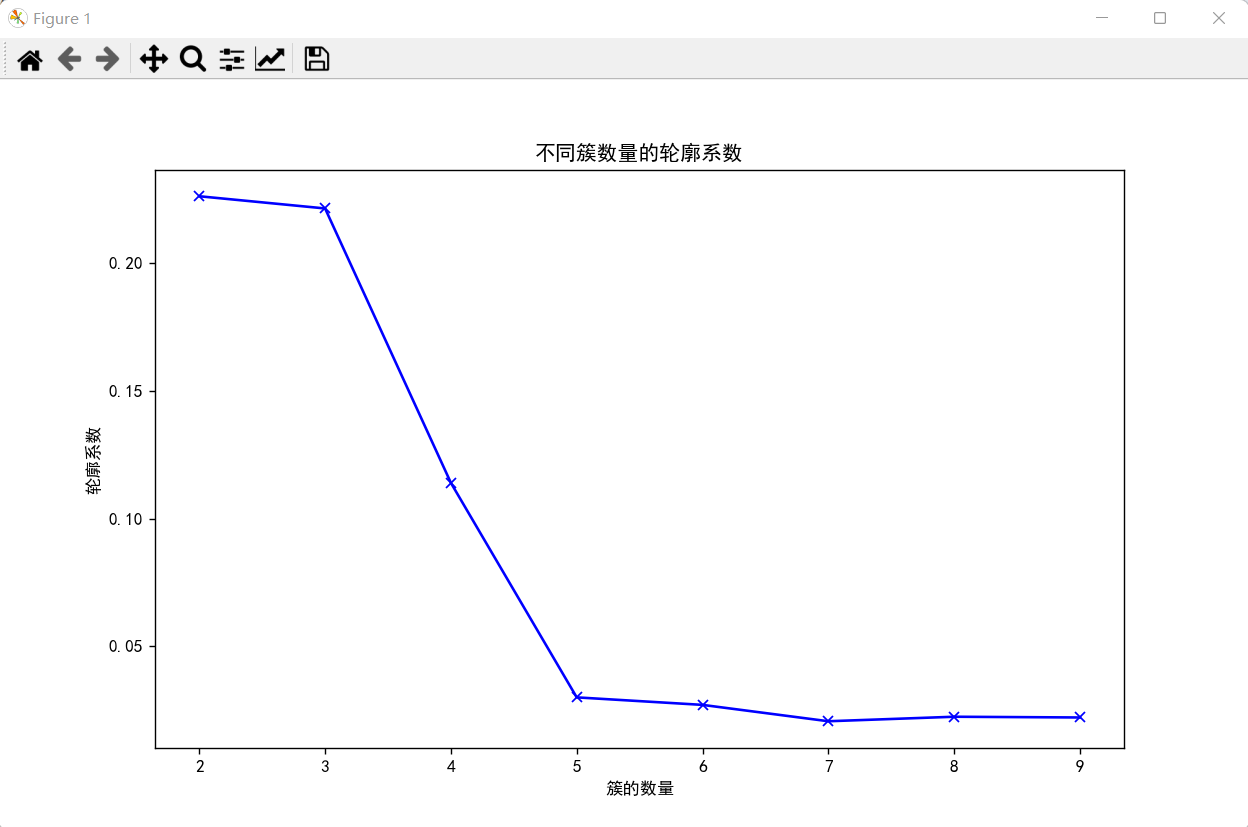


图6.5 不同簇数量的轮廓系数图

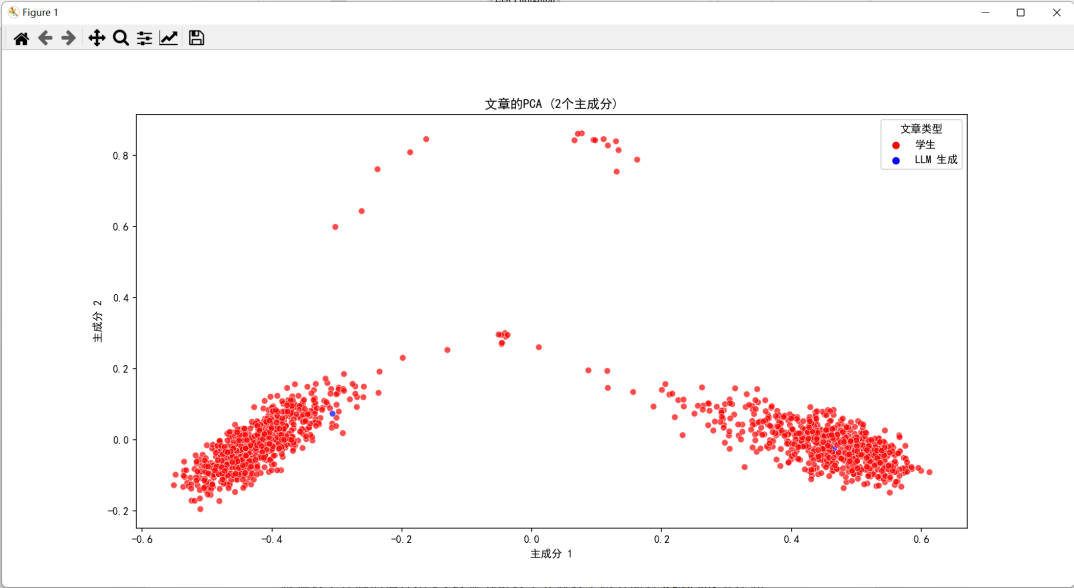


图6.6 训练集数据的PCA

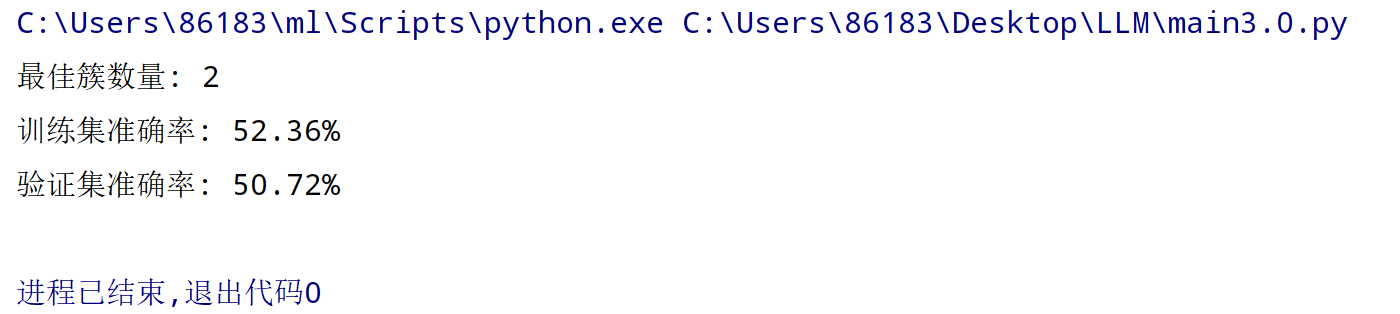


图6.7 模型准确率

根据以上代码的运行结果，我认为可以主要就以下两方面对K-Means算法和监督学习算法进行比较：

1. 准确率。K-Means算法在训练集上的准确率较低，可能由于K-Means主要用于无监督学习中的聚类任务，而在分类任务上表现一般。综合使用逻辑回归、支持向量机和贝叶斯分类器集成的模型在训练集上表现更好，我们认为这可能是因为这些模型能够更好地捕捉数据的复杂关系。
2. 训练时间。K-Means算法的训练时间较短。由于K-Means是一种无监督学习方法，其训练过程主要涉及数据点之间的距离计算和簇中心的更新，相对较快。综合使用逻辑回归、支持向量机和贝叶斯分类器集成的模型的训练时间较长[8]。这是因为这些模型可能需要更多的迭代和参数调整，尤其是在集成模型中，需要对多个模型进行训练和协同工作。

由于K-Means算法的准确率相对较低，而综合使用逻辑回归、支持向量机和贝叶斯分类器集成的模型在准确率上表现更好，我们最终选择了投票分类器，以在牺牲一些训练时间和模型复杂度的情况下获得更高的训练集准确率。

# 七、参考文献

[1]张重毅,牛欣悦,孙君艳等.ChatGPT探析：AI大型语言模型下学术出版的机遇与挑战[J].中国科技期刊研究,2023,34(04):446-453.

[2]沈锡宾,王立磊.人工智能生成学术期刊文本的检测研究[J].科技与出版,2023,(08):56-62.DOI:10.16510/j.cnki.kjycb.2023.08.002

[3]苏金树,张博锋,徐昕.基于机器学习的文本分类技术研究进展[J].软件学报,2006,(09):1848-1859.

[4]崔建明,刘建明,廖周宇.基于SVM算法的文本分类技术研究[J].计算机仿真,2013,30(02):299-302+368.

[5] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. arXiv preprint arXiv:1408.5882.

[6] Joulin, A., Grave, E., Bojanowski, P., Mikolov, T., Chen, K., & Mikolov, T. (2017). Bag of Tricks for Efficient Text Classification. arXiv preprint arXiv:1607.01759.

[7] 王亮, 李晓明, 张晨, 等. 基于深度学习的文本分类方法[J]. 计算机研究与发展, 2016, 53(7): 1512-1520.

[8]王千,王成,冯振元等.K-means聚类算法研究综述[J].电子设计工程,2012,20(07):21-24.DOI:10.14022/j.cnki.dzsjgc.2012.07.034