知识工程作业一

董培伦1320221087

1 问题重述

基于北京大学计算语言学教育部重点实验室的文字数据集“现代汉语切分、标注、注音语料库-1998 年1 月份样例与规范”，我们需要选取合适的方法获得每条句子中代词所指代的先行词。为了使自然语言具有可计算性，我们需要先将句子转化成向量表示，对于本问题，我们采取独热编码思想，不同于去年的“会见”任务，此任务需要考虑的更多，且为了提高模型的泛化性、避免数据泄露问题，故不再考虑统计txt中高频作为特征向量，而是把词性作为是否是先行词的主要依据。我们认为该问题为两个并行的二分类问题，并对句子特征进行线性假设，采取广义线性模型进行实现。考虑到数据集分布的不均衡性，因此我们选择采用对非先行词的部分下采样的方法，首先对训练集和验证集进行了平衡。最终，采取了Accuracy、Recall、F1 score三个评价指标，测试集上分别对模型的分类预测能力和泛化能力进行了评价。

2 数据预处理

2.1 基于词性生成独热编码的映射词典

由于我们基于词性来生成特征向量，故不再特意对文段进行清洗，而是选择保留每个字词后的符号部分，用来确定该字词的词性。使用`LabelEncoder`对象将分类变量`pos_tags`映射到整数编码。然后，使用`OneHotEncoder`对象对整数编码进行独热编码，将每个整数编码转换为一个相应的独热向量。最后，通过创建映射字典`mapping_dict`，将原始类别与独热编码向量对应起来。

2.2 建立json文件与txt文本的联系

遍历训练集中的不同`task_id`的json文件，通过行内容的ID遍历数据总集，获得相应的行内容，并把对应的json文件中的先行词和代词位置一起记录在字典`id_lines_dict = {}`中。外层字典的键是任务ID(`task_id`)，对应着不同的任务。外层字典的值是一个内层字典，其中每个键是匹配到的先行词(`value_0`)，对应着在文件中匹配到的内容。内层字典的值是一个包含相关信息的字典，其中包括以下键值对：`"line_content"`：对应着匹配到的行内容。`"0_index_front"`：对应着先行词在行中的前索引位置。`"0_index_behind"`：对应着先行词在行中的后索引位置。`"pronoun_index_front"`：对应着代词在行中的前索引位置。`"pronoun_index_behind"`：对应着代词在行中的后索引位置。基于此字典，搭建了taskID—行内容id—行内容—先行词与代词的线性对应关系，后续提取特征向量时，即可遍历本字典，获得清晰且信息明确的文段特征。示例如下：

Figure 1: 字典示例

3 构造特征向量

3.1 当前字词的词性

为了提高模型的泛化性、避免数据泄露问题，故不再考虑统计txt中高频作为特征向量，而是把词性作为是否是先行词的主要依据。由于先行词是名词类词，如人名、地名、名词物品等，故合理认为，当前词的词性是判断是否为先行词的重要因素。故判断当前词词性，并映入映射字典中，得到当前字词词性的独热编码，是构造特征向量的第一步。首先用两层循环遍历上文提到的字典`id_lines_dict` ，外层循环：它遍历了一个名为`id_lines_dict`的字典中的每个元素。这个字典的键是任务的ID，而值是另一个字典`value_0_dict`。内层循环：内层循环遍历了`value_0_dict` 字典中的每个元素。该字典的键是行内容id，而值是另一个字典`line_dict`中的行内容与先行词、代词的位置。由于字典中的起始部分是编码id的词性“/m”，故为了获取第一个汉字的词性，代码使用`next` 函数和生成器表达式找到了给定字符串`line` 中第一个汉字的索引。这是通过遍历`line` 中的每个字符，并检查字符是否位于汉字的Unicode 范围内来实现的。接着由于词性是不定长度的字符串，且起始为斜杠（/），所以代码第二步根据第一个汉字的索引，在字符串`line` 中找到了斜杠字符（/）的索引。它使用了列表推导式来获取所有斜杠字符的索引。随后因为表示词性的字符串结束后用空格隔开下一个字词，代码通过跳过第一个斜杠后的字符，并遍历直到遇到空格或字符串结尾，生成了一个子字符串。这个子字符串存储在变量`substring` 中，并在映射字典查询，把获取到的值存储在变量`onehot_encoding` 中。

3.2 下一位相邻字词的词性

考虑到先行词后一般情况下为谓语动词，如“彭/nrf　楚政/nrg　扶贫帮困/jv ”，所以我们特征向量的第二部分为下一个词的词性的独热编码。用同样的方法继续遍历下一个斜杠（/），直到遇到空格停止，把这之间的字符串对应在映射字典中，记录其独热编码。

3.3 相距代词的距离

我们猜测先行词距离代词的距离是有限制的，不可能无限大，也不可能过小，故把这段距离作为特征向量的最后一部分。最终把这三部分记录在result矩阵中，并把矩阵转化为DataFrame形式，输出csv文件，以便后续模型训练。

4 模型训练

本任务我们选择广义线性模型，基于是否为先行词，对特征向量进行二分类，01为标签。由于标签01数量对比悬殊，故对标签0的部分进行下采样，经过调试，观察模糊矩阵，发现在训练集中，01比例为1.5：1时效果最好，故对三个数据集进行同样的下采样调整，使得获得均衡有效的标签。

4.1 参数调节

通过手动调节，发现学习率在0.002，训练轮数为8000轮，threshold为0.44时，模型表现更好，故最终参数如上。（图片放在screen中）

4.2 模型思路

- 1、`sigmoid` 方法：计算给定输入z 的`sigmoid` 函数值。通过应用S 形函数，将线性模型的输出转换为0 到1 之间的概率值。

- 2、`predict` 方法：对给定的输入数据X 进行预测。首先，计算线性模型的输出（通过将输入数据X 与权重weights 相乘并加上偏置bias）。然后，将线性模型的输出通过`sigmoid` 函数转换为概率值。根据阈值threshold，将概率值转换为二进制类别标签，返回预测的类别标签列表。

- 3、`fit` 方法：使用梯度下降算法拟合Logistic 回归模型。在训练过程中，根据输入数据X 和标签y，通过迭代优化权重weights 和偏置bias。在每次迭代中，计算线性模型的输出，并使用`sigmoid` 函数将其转换为概率值。然后，计算梯度，即对权重和偏置的偏导数。根据梯度和学习率，更新权重和偏置的值。同时，计算训练数据和验证数据的交叉熵损

失（cross_loss），并将其存储在相应的列表中。最后，返回训练数据和验证数据的损失列表。

4.3 模型效果

模型效果如下，Figure2为loss函数图像，从图中可见，随着训练轮数地增多，函数逐渐下降且趋于平缓，在验证集也表现出同样为此趋势，可见模型地泛化性能还比较好。Figure3为测试集的模糊矩阵，可见对于标签01的判断很准确且较为均衡，反映出模型对测试集的验证效果较好。 最终结果：

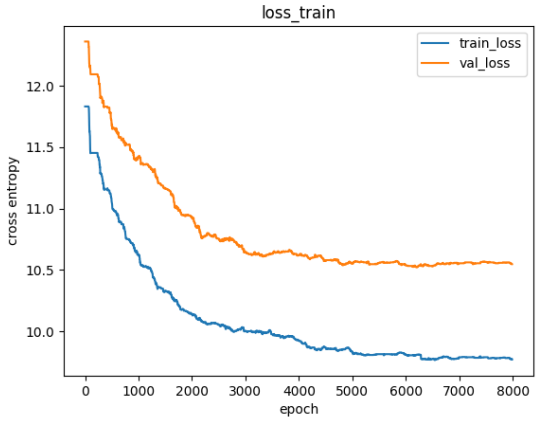


Figure 2: loss图像

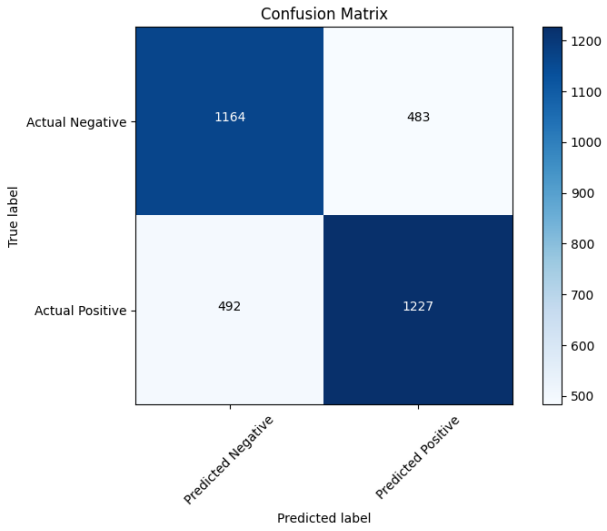


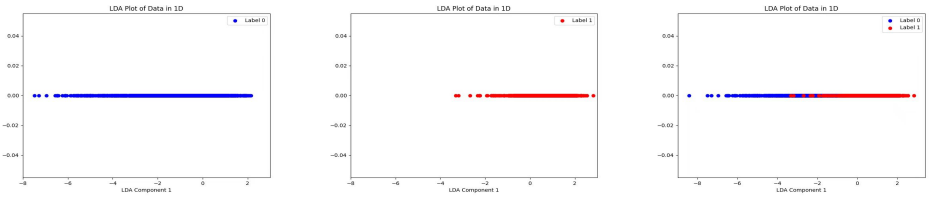
Figure 3: 模糊矩阵

训练集：`Accuracy:70.74% Precision: 57.26% Recall:73.37% F1 score:64.32%`;测试集`Accuracy:71.51% Precision: 71.04% Recall:74.64% F1 score:72.79%`

5 问题及后续

5.1 验证线性模型可否继续优化

经历了参数优化后，F1 score最终保持在`72.79%`，并不是很高。对于自然语言处理，猜测线性模型过于简单，不太适合此任务，于是我对下采样做了LDA（线性判别分析）来验证。



(a) label 0 (b) label 1 (c) label 0 and label 1

Figure 4: LAD

线性模型要适用于分类任务，它应该是线性可分的，做LDA之后，它应该明显地分为两簇，但是在LDA分析当中，我们发现label1和label0有很大的重合，反映出现在分类准确率不高的主要原因是线性模型本身不适用，想要达到更好的效果还需进一步修改。

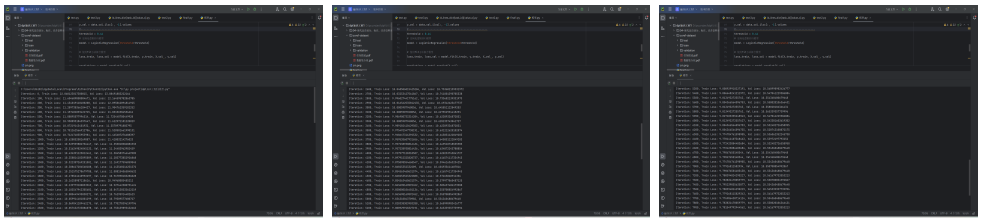
5.2 两个优化角度

为了尝试继续提高模型准确度，我们可以思考两种思路。1、猜测对于自然语言处理，线性模型过于简单，可以尝试更改为决策树模型。2、特征向量不够丰富，可以尝试继续从其他角度丰富特征向量。

5.3 尝试验证

相较于更改特征向量，尝试决策树模型更为简单，所以我继续尝试了用决策树模型训练，以下是最终结果。准确率：0.7564270152505447,由此可见，虽然有小部分提升，但仍效果不明显。所以为了更好地提升模型效果，需要在后期尝试丰富特征向量。

6 补充迭代图



(a) image 1 (b) image 2 (c) image 3

Figure 5: 迭代图