句子相似度匹配

（自然语言处理）

刘凯悦

2018 年 11 月

目录

1 问题的定义

1.1 项目概述

本项目是自然语言处理领域的问题。

Quora 是一个获取和分享知识的网站，这是一个提出问题并与提供独特见解和质量答案的人联系的平台。这使人们能够相互学习，更好地了解世界。每个月有超过 1 亿人访问 Quora，每一个用户都可以提出问题并寻求答案，因此很多人提出相似措辞的问题也就不足为奇了。导致的问题是，具有相同意图的多个问题可能会导致寻求者花更多时间找到问题的最佳答案，并使作者觉得他们需要回答同一问题的多个版本。本项目的出发点是为了给 Quora 作家、求职者和读者提供更好的体验，故需要将一个问题的多个不同措辞的版本识别为同一问题，并标注出来，不断减少重复的问题，读者能够更容易的找到问题的高质量答案，作者不必多次回答同一个主题的问题。

项目选择的数据集是 Kaggle 竞赛提供的数据，训练数据包括 404290 行已被标记的数据，测试数据包括 3563475 行未标记的数据。

1.2 问题陈述

模型需要判定测试数据的 question1 和 question2 是否具有基本相同的含义，如果是则目标变量 is_duplicate 设置为 1，否则设置为 0。训练集已经被标记，因此该问题是监督学习问题，而且标记类型是 0 和 1，是一个二分类问题。

解决这个问题实际是两个部分，通过训练数据进行学习并建立模型，第二部分是对测试数据进行二分类处理。

期望的结果是能正确将变量分类。

最后将该 CSV 文件上传到 Kaggle，Kaggle 根据该项目的判断指标来对预测结果进行评分。

1.3 评价指标

评估指标选择对数损失函数。对数损失函数的计算公式如下：

$$L(Y,P(Y|X))=-\log P(Y|X)=-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}y_{ij}\log(p_{ij})$$

Y 是输出变量，X 是输入变量，L 是损失函数，N 是输入样本量，M 是可能的类别数，yij 是一个二值指标，表示类别 j 是否是输入实例 xi 的真实类别，pij 是模型预测输入实例 xi 属于类别 j 的概率。

关于阈值。我选择 kaggle 上 Top10%水平作为基准阈值，Logloss 值低于 0.15632。

2 分析

2.1 数据的探索

2.1.1基本探索

数据集下载地址：https://www.kaggle.com/c/quora-question-pairs/data。

GoogleNews-vectors-negative300.bin 下载地址：https://s3.amazonaws.com/dl4j-distribution/GoogleNews-vectors-negative300.bin.gz。

有三个数据集：sample_submission.csv、test.csv、train.csv。

sample_submission.csv 数据集。2345796 行观测，2 个变量 test_id 和 is_duplicate。

train.csv 数据集。即训练集，404290 行观测，6 个变量 id、qid1、qid2、

question1、question2、is_duplicate。使用 DataFrame.head(5)函数，得到数据集前 5 行如下图，其中 255027 不重复样本，149263 重复样本。没有缺失值，但存在空值。

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0

- id：本条记录的 ID 号
- qid1：问题 1 的总的顺序 ID
- qid2：问题 1 的总的顺序 ID
- question1：问题 1
- question2：问题 2
- is_duplicate：本条记录中两个问题是否相似，1 代表重复，0 代表不重复

test.csv 数据集。即测试集，3563475 行观测，3 个变量 test_id、question1、question2。测试集的前 2345796 行比较结果已在 sample_submission.csv 数据集给出。

数据 size 总结如下表：

	Train	Test
Data Size	404290	2345796
Vocab Size	95603	101049

2.1.2测试集和训练集的异同

训练集的词表大小:8946784；测试集的词表大小:78568963。

test.csv 大小是 477.59MB，train.csv 大小是 63.4MB。值得注意的是，测试数据比训练数据多，一些测试数据是为了阻止手工标记而设计的虚拟数据，并且不包括在计

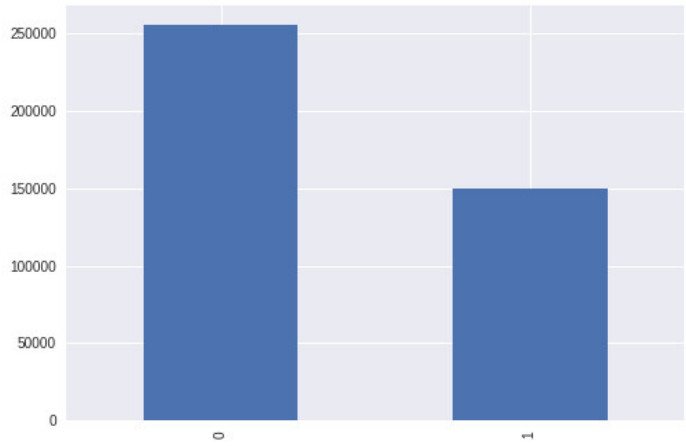
算中。如下图：

```
test.csv                            477.59MB
train.csv                           63.4MB
quora_duplicate_questions.tsv       58.18MB
sample_submission.csv               22.35MB
```

2.2 探索性可视化

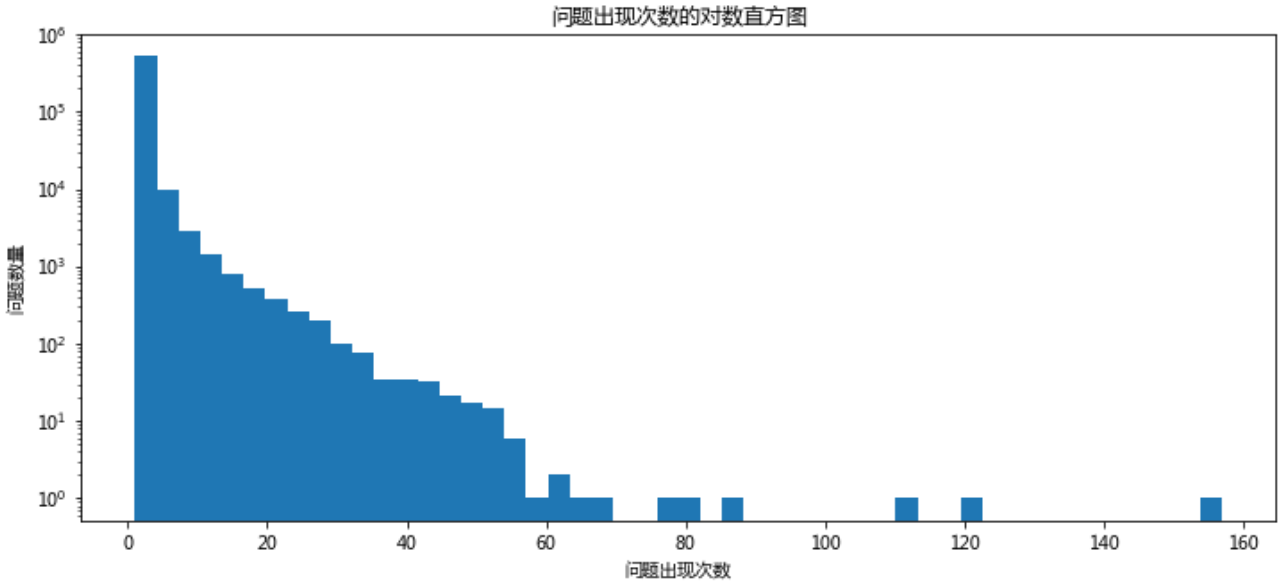
2.2.1train 数据集中问题对重复与不重复的分布

样本分布如下：

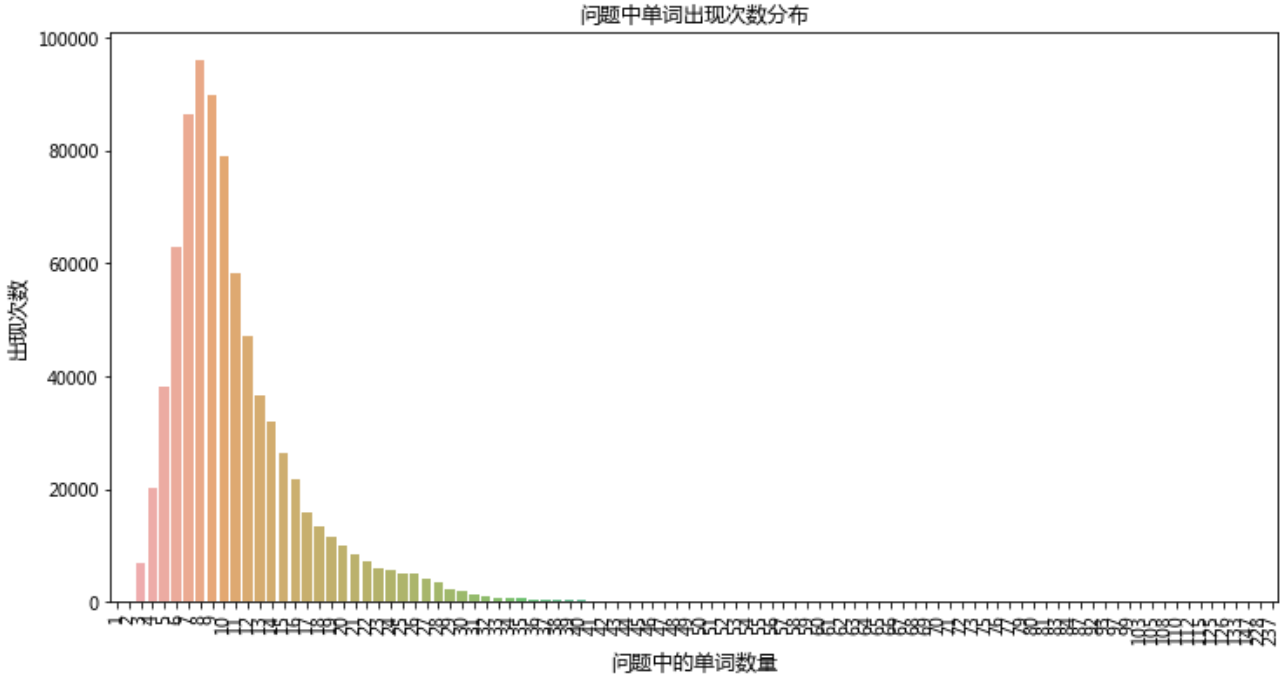


2.2.2train 数据集，句子出现的频次

训练数据中的问题总数: 537929；出现多次的问题的数量: 111778。下图是同一个问题在 train 数据集中出现次数的统计。可以看出大多数问题只出现几次，很少有问题出现多次(少数问题出现多次)，虽然有一个问题出现了 160 多次，但这是异常值。



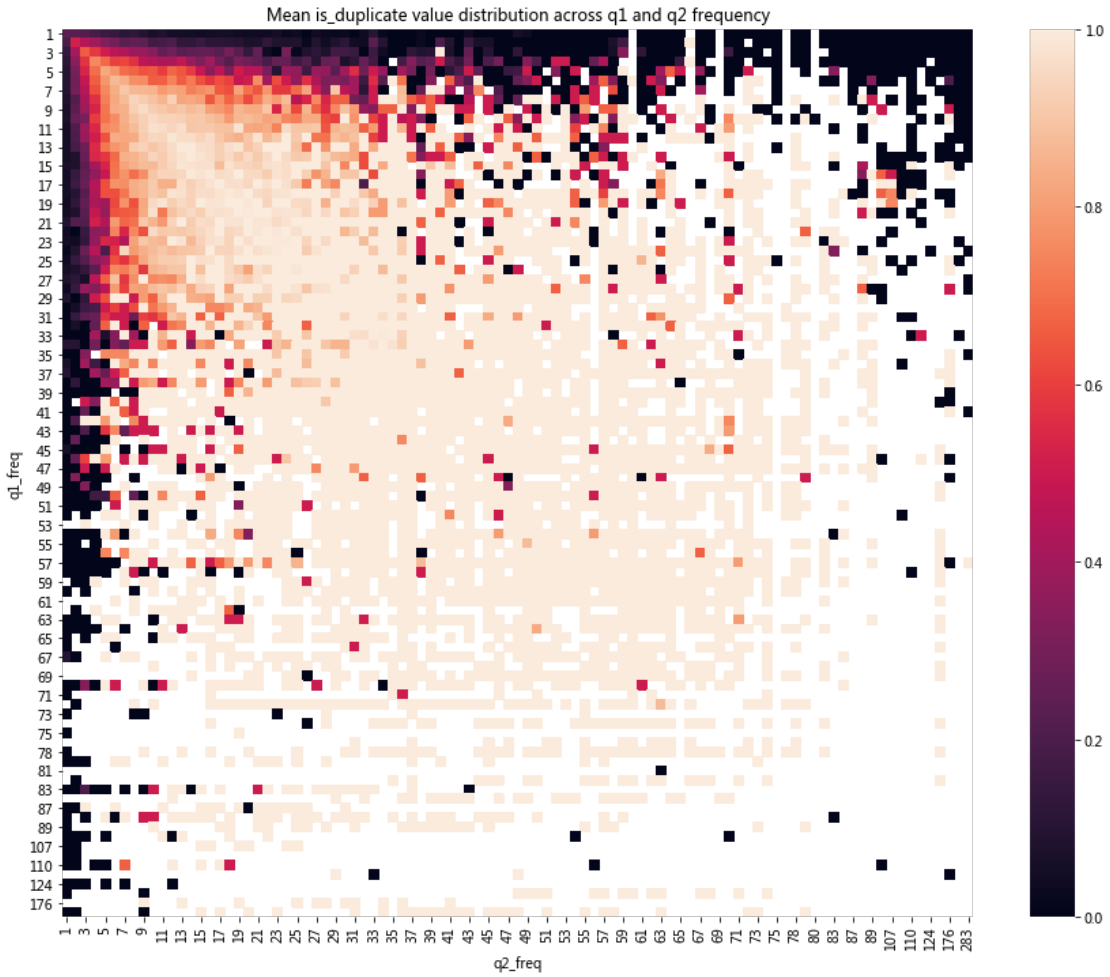
2.2.3train 数据集，问题中单词出现的数量分布



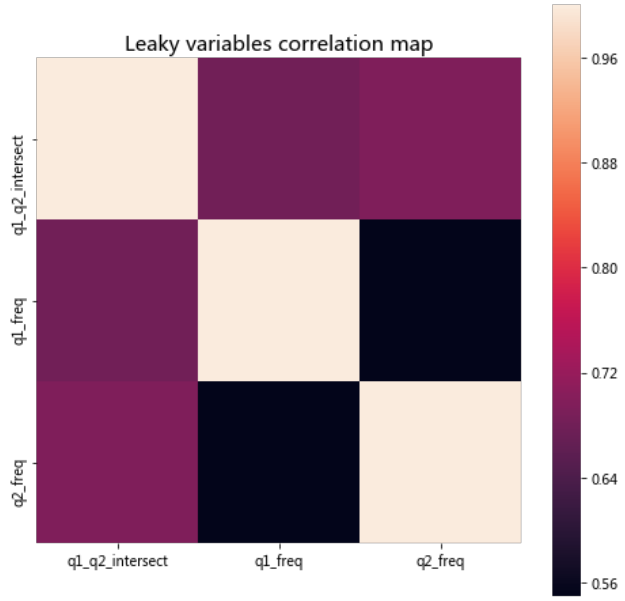
由上图看出，含有最多单词的问题由 237 个单词组成。也有一些问题与 1 或 2 个单词组成，大部分的问题单词数集中在 30 以内。

2.2.4句子出现频次和句子 is_duplicate 的相关性

如下图，查看问题 1 和问题 2 出现频率与 is_duplicate 值得分布关系：

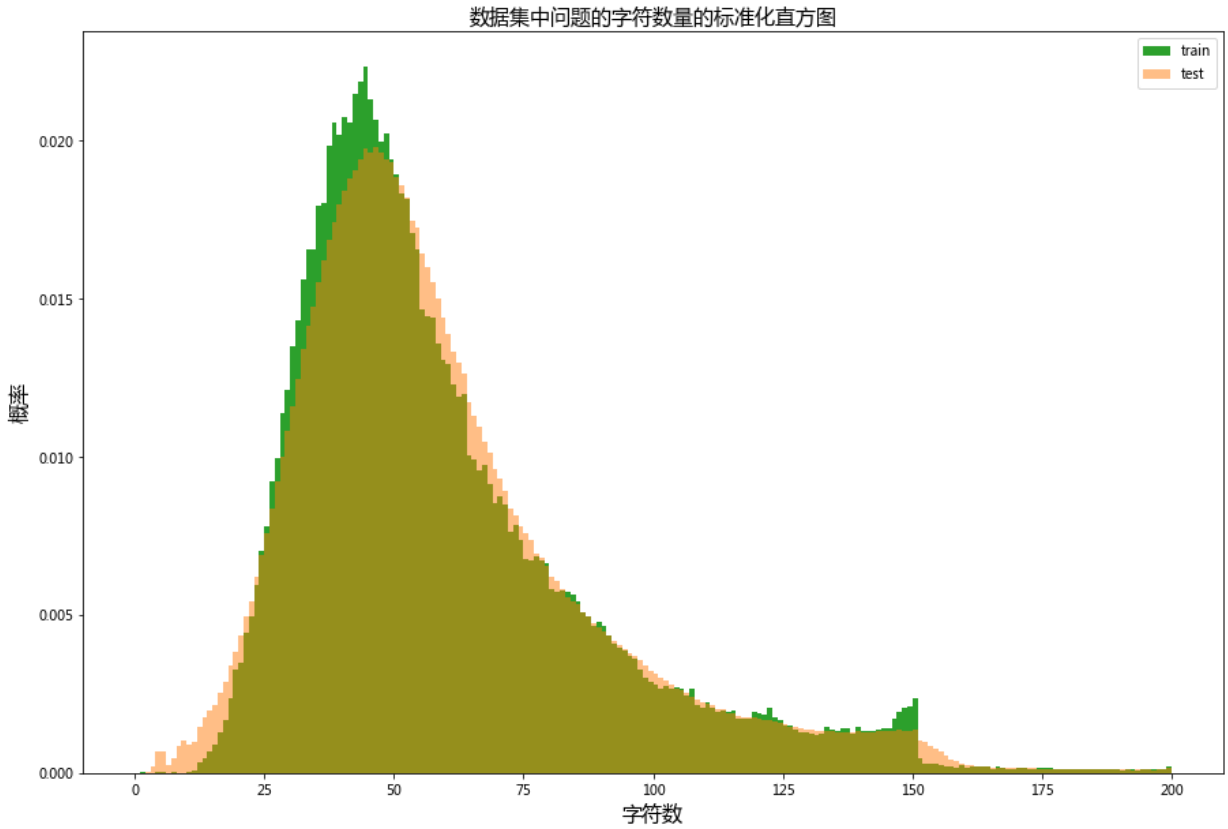


下图是问题 1 出现频率、问题 2 出现频率、一个问题对相同部分三者之间的相关关系程度图：

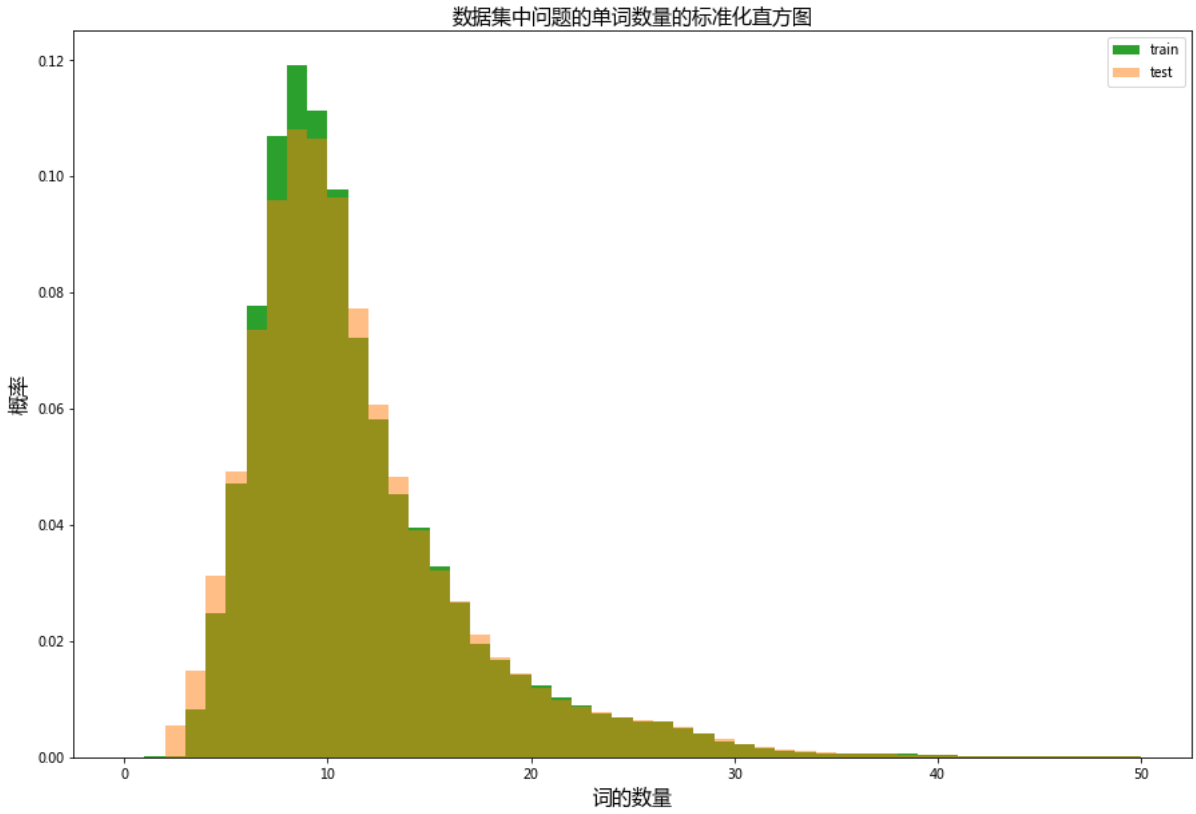


2.2.5词表

训练集和测试集中问题的字符数量的标准化直方图如下：



训练集和测试集中问题的单词数量的标准化直方图如下：



其中，训练集平均值为 11.06；训练集标准差为 5.89；测试集平均值为 11.02；测试集标准差为 5.84；训练集中最大值为 237.00；测试集中最大值为 238.00。

训练集的词表大小:8946784；测试集的词表大小:78568963。

2.3 算法和技术

解决本问题的总体思路是特征提取和使用监督学习对问题对二分类。

2.3.1特征算法

Fuzzywuzzy 是一个 Python 库，使用编辑距离（Levenshtein Distance）来计算序列之间的差异。在信息论，语言学和计算机科学中，Levenshtein 距离是用于测量两个序列之间差异的字符串度量。非正式地，两个单词之间的 Levenshtein 距离是将一个单词更改为另一个单词所需的单字符编辑（插入，删除或替换）的最小数量。它以苏联数学家弗拉基米尔·莱文斯坦（Vladimir Levenshtein）的名字命名，后者在 1965 年考虑过

这个距离[9]。距离计算公式如下：

$$\mathrm{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} \mathrm{lev}_{a,b}(i-1,j) + 1 \\ \mathrm{lev}_{a,b}(i,j-1) + 1 \\ \mathrm{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

TF-IDF 是一种文本特征提取技术。TF-IDF 是 Term Frequency - Inverse Document Frequency 的缩写，即“词频-逆文本频率”。它由 TF 和 IDF 两部分组成。一个词 x 的 IDF 的基本公式如下：$\mathrm{IDF(X)} = \log\frac{N}{N(X)}$ ，N 代表语料库中文本的总数，而 N(x)代表语料库中包含词 x 的文本总数。当某一个生僻词在语料库中没有时，N(x)为 0，IDF 值变的没有意义。因此对 IDF 做一些平滑，使语料库中没有出现的词也可以得到一个合适的 IDF 值。平滑的方法有很多种，最常见的 IDF 平滑后的公式之一为：$\mathrm{IDF(X)} = \log\frac{N+1}{N(X)+1} + 1$，某一个词的 TF-IDF 值为 TF−IDF(x)=TF(x)∗IDF(x)。

奇异值分解(Singular Value Decomposition，以下简称 SVD)是在机器学习领域广泛应用的算法，它不光可以用于降维算法中的特征分解，还可以用于推荐系统，以及自然语言处理等领域。SVD 也是对矩阵进行分解，但是和特征分解不同，SVD 并不要求要分解的矩阵为方阵。假设我们的矩阵 A 是一个 m×n 的矩阵，那么我们定义矩阵 A 的 SVD 为：$A = U\Sigma V^T$。其中 U 是一个 m×m 的矩阵，Σ是一个 m×n 的矩阵，除了主对角线上的元素以外全为 0，主对角线上的每个元素都称为奇异值，V 是一个 n ×n 的矩阵。U 和 V 都是酉矩阵，即满足 $U^TU=I, V^TV=I$。

自己定义了一个 sent2vec 函数，通过这个函数来生成简单的问题 1 向量和问题 2 向量，再计算两个向量的各种距离。

关于 wmd 特征。Word2vec，也叫 word embeddings，中文名“词向量”，将自然语

言中的字词转为计算机可以理解的稠密向量，该算法有两种重要模型 Skip-gram(Continuous Skip-gram Model)和 CBOW(Continuous Bag-of-Words Model)，二者的区别是，Skip-gram 利用一个单词预测其上下文单词，而 CBOW 正相反，是从词语已知上下文中预测这一词语。

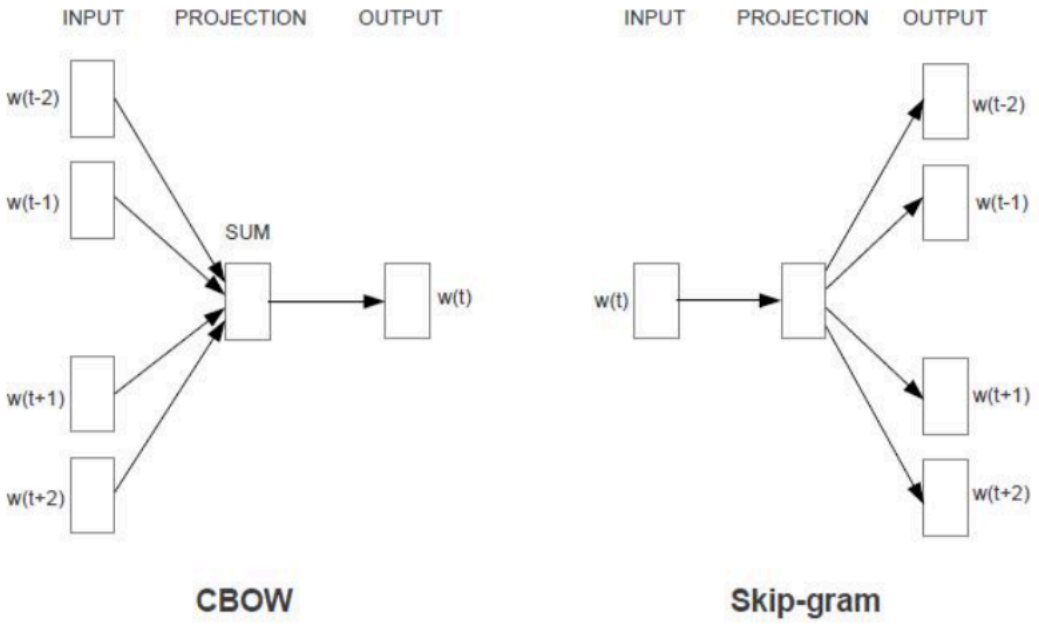
Skip-gram 模型输入输出正相反，其目标函数为：

$$\mathcal{L} = \sum \log p(Context(\omega) \mid \omega)$$

CBOW 模型，已知词语上下文内容(w)与词语 w,目标函数是对数似然函数：

$$\mathcal{L} = \sum \log p(\omega \mid Context(\omega))$$

CBOW 和 Skip-gram 结构图如下：



散列算法，是把任意长度的输入（又叫做预映射 pre-image）通过散列算法变换成固定长度的输出，该输出就是散列值。本次使用 train 和 test 数据集的所有问题构造成能够被映射的字典，将问题 1 和 2 进行映射得到散列特征，再通过这些特征得到问题

和 2 的频率特征。

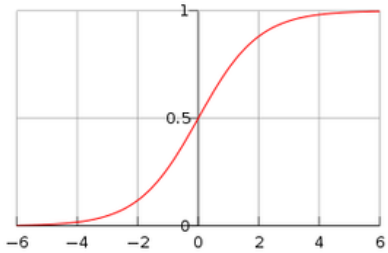
本项目中，在计算问题 1 和问题 2 相同部分时，还用到了 Python 的 intersection() 方法它用于返回两个或更多集合中都包含的元素，就是计算他们的交集。

Pagerank 算法[11]。PageRank 是由谷歌创始人拉里·佩奇和谢尔盖·布林开发的一种巧妙的算法。Pagerank 是根据节点之间的链接，对图结构中的节点进行排序。与重要节点共享的边也会使该节点增加权重，变得重要，与垃圾节点链接会使该节点本身权重降低，排名下滑。但为什么在这里使用该算法呢？在文本的上下文中，每个节点都是数据集中的一个问题，一条边代表一个问题对。通常情况下，相关问题(主题)共享一条边，但语义上可能不相等——但是这条边对于可视化主题集群和某些节点的重要性非常有用。因此，如果一个问题与一个有较高的 pagerank 值得问题匹配，那么这两个问题本身就很相关。

2.3.2算法模型之 logistic 回归

logistic 回归，是一种广义的线性回归分析模型，主要是进行二分类预测，输出介于 0~1 之间的一个值，当概率大于 0.5 预测为 1，小于 0.5 预测为 0。其数学表达式 sigmoid=1/(1+exp(-inX)),该函数的曲线形状类似于 S 型，也称 S 函数，x=0 时，函数值为 0.5。图示如下：

$$f(z)=\frac{e^z}{e^z+1}=\frac{1}{1+e^{-z}},$$



Logistic 回归的参数一般使用最小二乘法来估计，同时使用梯度下降不断调试，

直至最终找到合适的参数值。

使用的 Python 库如下：

- Numpy[1]
- Pandas
- Sklearn
- Matplotlib
- Fuzzywuzzy
- Tqdm
- Nltk
- Xgbboost

优点

- 实现简单，易于理解和实现；
- 计算代价较低，效率高；
- 能够方便的观测到样本概率分数。

缺点

- 容易欠拟合，很多时候分类精度较低；
- 不能拟合非线性数据；
- 对模型中自变量多重共线性较为敏感，例如两个高度相关自变量同时放入模型，可能导致较弱的一个自变量回归符号不符合预期，符号被扭转。需要利用因子分析或者变量聚类分析等手段来选择代表性的自变量，以减少候选变量之间的相关性。

2.3.3算法模型之 XGBoost

优点：

- XGBoost 的代价函数里有正则项，能够较好的控制模型的复杂度；
- XGBoost 工具支持并行并且非常灵活，支持用户自定义目标函数和评价函数，前提是要求目标函数二阶可导；
- 对于特征的值有缺失的样本，XGBoost 可以自动学习出它的分裂方向；
- 相对于 GBM，XGBoost 先从顶到底建立所有可以建立的子树，再从底到顶反向进行剪枝。极大地降低了陷入局部最优解的概率；
- 相对于 GBM，XGBoost 允许在每一轮 Boosting 迭代中使用交叉验证。因此，可以方便地获得最优 Boosting 迭代次数，而 GBM 使用网格搜索，只能检测有限个值。

缺点：

- 算法参数过多，导致调参效率很低，在选择参数时计算效率；
- 只适合处理结构化数据；
- 不适合处理超高维特征数据。

2.4 基准模型

本项目是 Kaggle 上已经结束了的竞赛项目，在 Leaderboard 下，仍可以查看该项目上所有的参赛者的排名及其 logloss。Logloss 越小排名越高，最终的排名以 Private Leaderboard 为准。本项目的目标是，测试集的分类结果能排在 Private Leaderboard Top 20%。根据 kaggle 上 top20%的水平，项目设定的目标为分类对数损失分数小于 0.18267。

3 方法

3.1 数据预处理

数据预处理。去除缺失值、标点符号，将所有大写字母变成小写。

分词处理。使用 nltk 包对文本进行分词处理。行程两个字段“question1_spilt”和“question2_spilt”。

去除停用词。使用 nltk 包对文本进行去除停用词的处理，分别得出 question1 和 question2 去除停用词后的两个新列“question1_remove_swords”和“question2_remove_swords”。

3.2 执行过程

3.2.1特征工程

挖掘基础特征。question1 和 question2 的文本长度、Difference in the two lengths、Character length of question1 without spaces、Character length of question2 without spaces、Number of words in question1、Number of words in question2、Number of common words in question1 and question2、question1 词数和 question2 词数的比值、question1 和 question2 相同词的数量、question1 和 question2 分词并去掉符号后，二者按位与运算结果与二者词数量和的比值（这里称为魔术特征）[5][6]。

fuzzy[7]特征工程[8]。Fuzzy 是一个快速实现常见语音算法的 python 库。常用于处理字符串相似性问题。使用 fuzzy 的 QRatio 方法，生成一个特征列“fuzz_qratio”。

使用 TF-IDF 和 SVD 技术得出的特征 f3。

使用 Word2Vec 词向量技术，用 Google 预训练的 300 维新闻语料的词向量

googlenews-vecctors-negative300.bin，来得到 wmd 和 norm_wmd 以及各类距离的特征。

使用散列技术得出的散列特征和频率特征。

根据句子中单词相似特征，得出一个问题对相同单词数量的特征。

最后，为防止特征丢失同时方便后续使用，使用 CSV 模块的 to_csv 方法，存储这些特征到本地。

3.2.2构造训练数据

不同的模型对输入数据的要求不同，分别构造训练集合验证集。总体上 20%数据用于模型评估和验证，80%数据用来训练模型。

3.2.3训练模型

本项目使用了两个算法模型来完成分类任务，分别是 `Logistic` 回归模型和 `xgboost` 模型。方便效果对比。

使用 Logistic 回归模型。使用特征 f1、f2、f3_3、f3_4、f3_5、f4_1、f4_2、f5（详细特征查看代码），得到准确率为 0.754，logloss 为 0.462675968762。

训练模型 XGBoost。首先随机设置参数，参数设置如下，max_depth 是树的深度，越大越容易过拟合，这里设置为 4；参数 eta 取值范围[0,1]，单棵提升树的学习率，通过缩减特征的权重使提升计算过程更加保守，这里设置为 0.02；参数 objective，定义学习任务及相应的学习目标，因为这是一个二分类的回归问题，这里选择 binary:logistic；早停参数 early_stopping_rounds 设为 50。选择特征是 f1、f2、f4_1、f4_2、f5、f7，使用训练集训练模型，使用验证数据预测模型，准确率为 0.914，logloss 为 0.20964670224。显然这一特征组合和 xgboost 模型训练效果更好一

些。

3.3 完善

在模型执行过程中，不断通过特征组合来尝试得出更高的准确率和更低的logloss。最终确定了如 3.2 节模型训练中提到的特征组合。

在使用 xgboost 时，调节参数能够优化模型的表现效果，这里我手动调参，调节参数'max_depth'，最初设置为 4，不断尝试使用 2、6、8、10，最终测试值为 10 时的效果最好。

4 结果

4.1 模型的评价与验证

logistic 回归得到的准确率是 75.4%，logloss 值是 0.462675968762。

RandomForest 模型得到准确率 84.76%，logloss 值是 0.32563975926。

XGBoost 调优后，准确率为 92.9%，logloss 值是 0.179185508139。

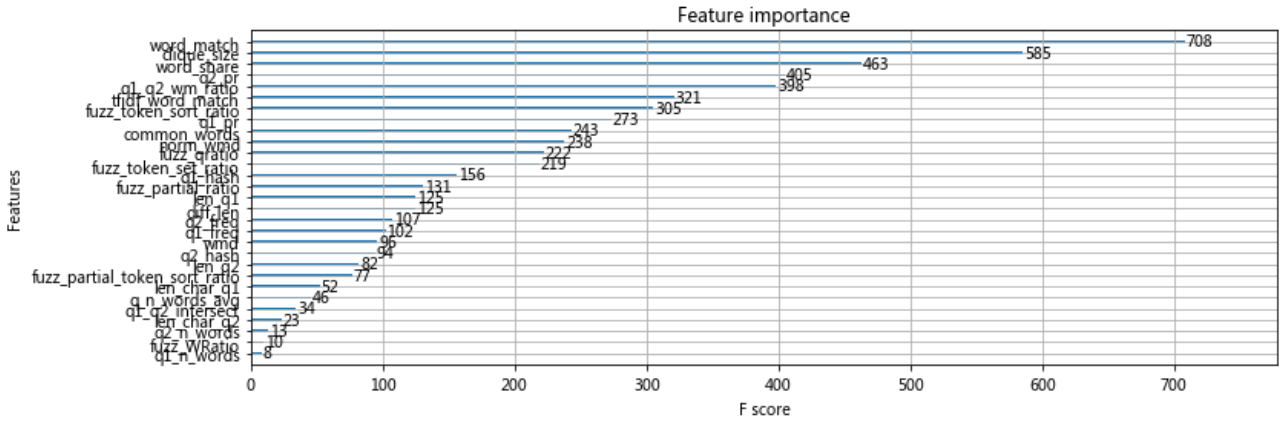
4.2 合理性分析

使用 logstic 回归作为基准模型得出的 logloss 为 0.462675968762，准确率是 75.4%。使用 xgboost 模型，准确率是 92.9%，logloss 值是 0.179185508139，比基准模型表现更好。92.9%的准确率是能够接受的，因此该模型能够解决该分类问题。

5 项目结论

5.1 结果可视化

以下是不同特征的重要性可视化：



模型参数呈现：'max_depth':4、'eta':0.02、early_stopping_rounds=50

模型树显示：

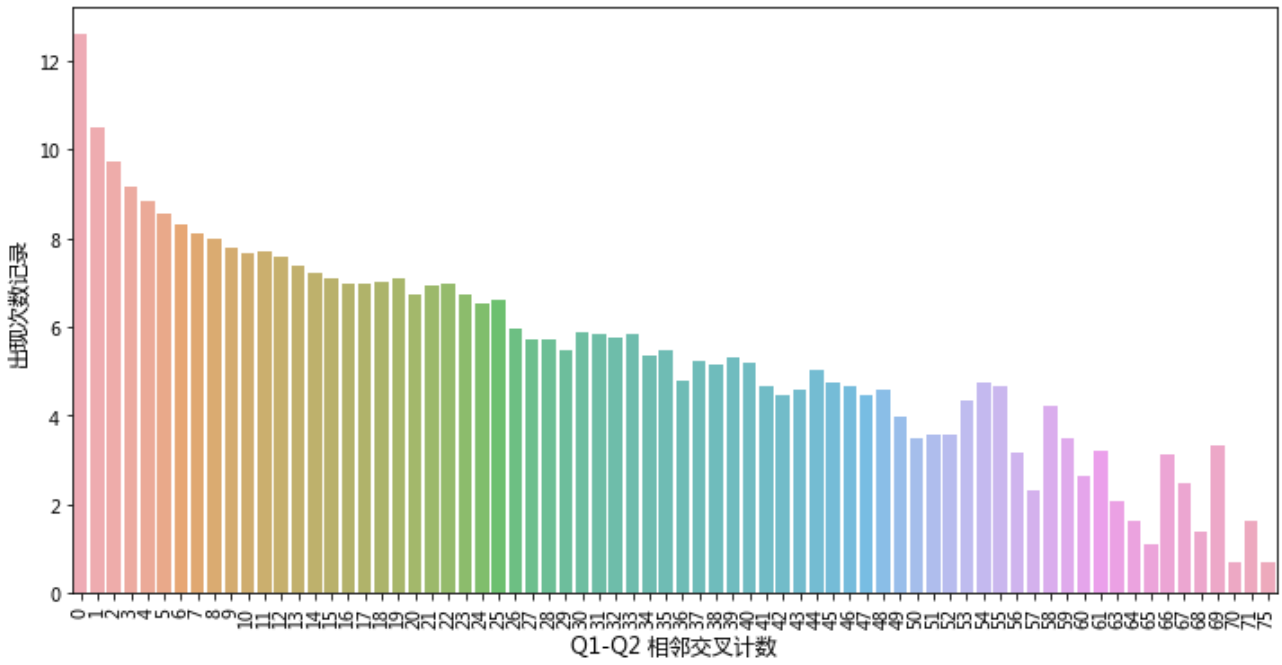
5.2 对项目的思考

本项目是一个较为基础的自然语言处理项目，看似简单，实则需要很多的耐心和细心才能做好。本项目主要任务有两个方面，特征工程和模型选择及训练，其中特征工程的主要任务就是使用现有的自然语言技术进行特征提取，这些自然语言处理技术包括词袋模型、TF-IDF 模型、停词过滤、分词等等。

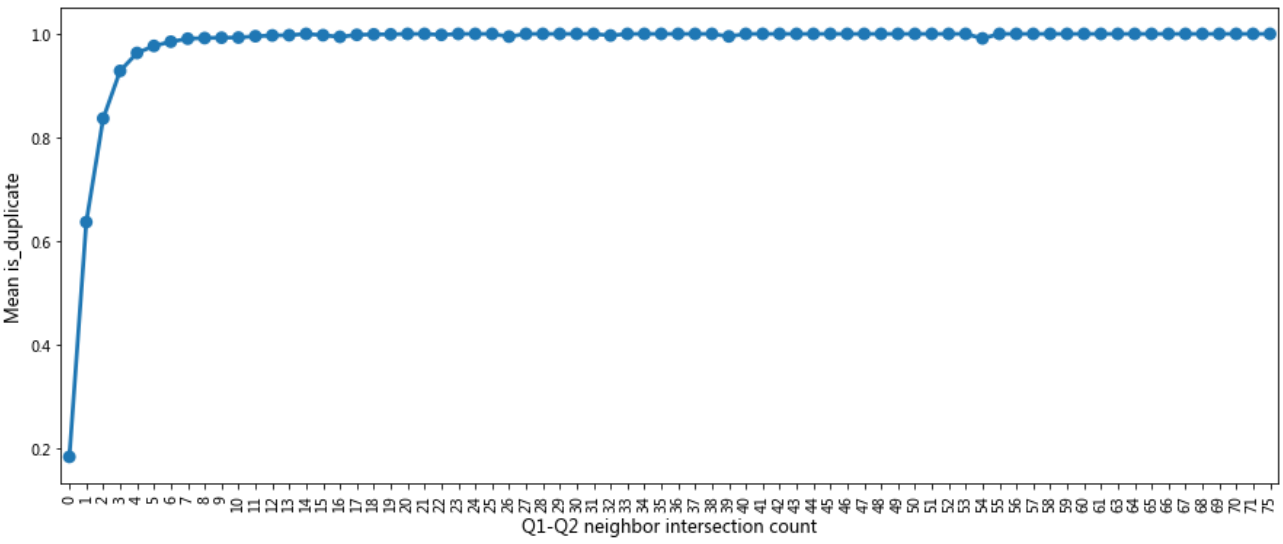
本项目的最大挑战就在于文本特征的提取，虽然自然语言处理技术已经较为成

熟，但是对于一个新的文本，使用这些技术提取特征仍然需要开阔的思路和丰富的经验。完成本项目，就对自然自然语言处理有了较初级的认识，为以后处理文本问题打下基础。

一个有趣的特征，Q1-Q2 neighbor intersection count 特征。第一步，先做简单的计数图，看到分布情况，如下：



第二步，查看 Q1-Q2 交叉部分的关于 mean is_duplicate 的分布图，如下：



可以看出交叉部分越多，is_duplicate 越接近于 1，这就解释了为什么这个变量具

有超强的预测性。

5.3 需要作出的改进

不断挖掘新特征。特征工程是提升本项目完成质量最主要的手段之一（我认为提升本项目完成质量最主要的两个手段是特征工程和模型选择）。因此增加更多的特征无疑会不断降低logloss。毕竟数据和特征决定了机器学习的上限，而模型和算法只是逼近这个上限而已。在挖掘新特征方面，我最初使用了f1、f2、f4_1、f4_2、f5、f3_3、f3_4、f3_5（具体特征详见代码），并使用逻辑回归和随机森林来训练，得到的logloss的最好结果只有0.32563975926。手动调参之后并没有很好的表现，于是，继续寻找特征，增加了f6、f7、f8、f9、f10等特征，由于数据类型限制，则使用xgboost来训练模型，主要参数是：'max_depth':4、'eta':0.02、early_stopping_rounds=50，得到的logloss是0.191257647193。

特征的选择与组合优化，也可以在一定程度上提升项目完成质量。

调参。本项目主要使用 XGBoost 完成，最初使用 GridSearchCV 来调节参数，通过对 max_depth、learning_rate、min_child_weight、n_estimators、gamma、subsample、colsample_bytree 等 7 个参数调节来寻找最优参数值，最终得到 logloss 值为 0.19634983942908477，还不如开始使用 xgboost 表现效果好，至此调参遇到很大困难。但我开始手动调节 xgboost 的参数将'max_depth':4 不断尝试提高和降低，最终确定'max_depth':10 时，得到最佳结果 logloss 值为 0.179185508139，好于 kaggle top20%表现结果。

但是在调参时，受限于硬件，为了兼顾效率，只进行了较少参数以及参数范围的调节，因此，测试更多的参数和更大的参数范围是一个改进方向。

参考文献

[1] TF-IDF: en.wikipedia.org/wiki/Tf%E2%80%93idf.
[2] scikit-learn userguide: scikit-learndevelopers, Release0.16.1.
[3] 有故事. XGboost 数据比赛实战之调参篇(完整流程).https://segmentfault.com/a/1190000014040317. Published: 2018-03-28.
[4] wgwyanfs.https://www.cnblogs.com/wgwyanfs/p/7151446.html.Published:2017-07-11
[5] Jared Turkewitz.https://www.kaggle.com/jturkewitz/magic-features-0-03-gain#
[6] Philipp Schmidt. Quora EDA & Model selection (ROC, PR plots).https://www.kaggle.com/philschmidt/quora-eda-model-selection-roc-pr-plots.
[7] https://pypi.org/project/Fuzzy/.
[8] Abhishek Thakur.Is That a Duplicate Quora Question?.https://www.linkedin.com/pulse/duplicate-quora-question-abhishek-thakur/.Published:2017-02-27
[9] https://en.wikipedia.org/wiki/Levenshtein_distance
[10] https://www.kaggle.com/c/quora-question-pairs/leaderboard
[11] https://www.kaggle.com/shubh24/pagerank-on-quora-a-basic-implementation