Table of Contents

句子相似度匹配项目

问题定义

项目概述

Quora Question Pairs 是 Quora 于 2017 年公开的句子匹配数据集，其通过给定两个句子的一致性标签标注，从而来判断句子是否一致。

该问题来自于 Quora 问答网站，在该网站上，用户可以在上面提问，并有其他用户给出高质量的回答或是独特的想法。这能鼓励人们互相学习，了解更多的知识。

每个月，超过一亿的用户会访问 Quora。在这么大的访问量下，经常会有用户提出相似

的问题。相似的问题会导致用户花费更多时间去寻找最佳答案，来回答他们想问的问题。而且也会让回答问题的用户觉得，对于同一个问题，他们需要回答多次。因此在 Quora 上，一个问题的最典型形式是很有价值的。它能给提问者和回答这提供最好的用户体验。

目前，Quora 使用的是一个随机森林模型来寻找类似的问题。而本次项目就是要探索使用更先进的技术来判断给定的两个问题是否是重复问题。高准确率的判断能够帮助网站找到每个问题最佳的回答，从而提高提问者，回答者以及网站浏览者的用户体验。这就是该项目的出发点。

而在技术层面上，该问题属于二分类问题。给定数据后，我们需要训练模型要将数据划分到“相同问题”和“不同问题”两个类别中。同时，给定的训练数据中包含了人工分类后的标签结果，所以该分类任务属于监督学习中的问题。最后，由于我们处理的数据属于自然语言，所以该问题也涉及到自然语言处理。

Quora 数据集训练集共包含 40K 的句子对，且其完全来自于 Quora 网站自身。句子对中每个句子以字符串的形式存储，每个字符串即句子的自然语言表达。

问题陈述

这个项目中要解决的问题是判断两个问题是否表达相同的意思。

使用的训练数据集主要包含三列数据：第一列为问题 1，数据类型为字符串吗，包含了问题对中第一个问题的自然语言表示形式，使用语言为英语。第二列数据为问题 2，数据类型和内容与第一列数据类似，包含的是问题对中第二个问题的自然语言表达。第三列是表示问题对中两个问题是否相同的一个标签，该标签为数值类型，只包含 0,1 两个值。0 代表两个问题意思不同，1 代表两个问题意思相同。数据样例如下：

	question1	question2	is_duplicate
0	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0
5	Astrology: I am a Capricorn Sun Cap moon and c...	I'm a triple Capricorn (Sun, Moon and ascendan...	1

	question1	question2	is_duplicate
6	Should I buy tiago?	What keeps childern active and far from phone ...	0
7	How can I be a good geologist?	What should I do to be a great geologist?	1

解决方案

这个问题是分类问题，所以使用分类模型来解决该问题。方案的基本思路按照分类模型的训练过程进行：首先从原始数据中提取特征，然后选择并训练模型，并进行模型调参。最后，选择效果最好的模型，并用该模型解决问题。具体细节如下

1. 特征提取
   (1) 原始数据为自然语言，无法直接作为分类模型的输入，因此要在自然语言的基础上生成一些特征，例如字符串的长度，包含单词数量等。
   (2) 由于原始数据为自然语言，因此可以使用自然语言处理的技术来生成特征。例如使用词袋模型，tf-idf 模型，句子模糊匹配等技术对自然语言处理，并生成与自然语言相关的特征。
2. 模型训练
   (1) 数据清洗：通过特征提取获得了特征数据，但该数据中往往包含一些异常数据。例如空值，或数据范围过大的数据。因此在训练模型前需要对数据进行清理
   (2) 分割训练数据集和测试数据集。由于后续涉及到模型选择，还需要检验数据集（validation data set）
   (3) 对选取的模型进行调参
3. 模型选择
   (1) 模型训练过程中会选取多个分类模型进行训练。在最后比较模型间性能差异，通过测试数据集选择最好的模型来解决问题

以上为解决问题的过程。最终期望的结果是能够获得效果明显的模型。通过该模型判断给定的两个句子是否含有相同得到意思。

评价指标

我们需要训练的是一个分类模型，因此可以考虑使用分类模型的评价标准。本次我们选取 f1-score 和 accuracy score 来进行评价。

accuracy score 是计算模型预测准确性的指标，其定义为：

$$accuracy\left(y,\hat{y}\right)=\frac{1}{n_{samples}}\sum_{i=0}^{n_{samples}-1}1\left(\hat{y}=y_i\right).$$

代表了预测模型中，准确预测的结果数量与总预测结果数量的比值。

f1-score 的定义为：

$$F_1 = \left(\frac{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}.$$

其中 precision 和 recall 的定义为：

$$\mathrm{Precision} = \frac{tp}{tp + fp}$$

$$\mathrm{Recall} = \frac{tp}{tp + fn}$$

Precision 代表了模型预测准确的阳性结果与所有预测为阳性结果的数量比，而 Recall 代表所有阳性结果中，被模型预测为阳性结果的比值。而 f1-score 则是 Precision 和 Recall 的调和平均数。能够综合反映模型预测的 Precision 和 Recall。

这两个平均数能够反映出模型预测结果的准确性，因此用来做模型的评价标准。

分析

数据的探索

使用的数据为 Quora Question Pairs, 该数据集中使用的训练数据集包含三个列：question1， question2， is_duplicate。这三个列分别代表了句子对中的第一个句子，第二个句子和句子对是否表示相同意思的标签。该标签为数值类型，只包含 0 和 1 两个数值。其中 0 表示句子对意思不同，1 表示句子对意思相同。为了方便后面的分析，在后文中，我将意思相同的数据成为正样本，意思不同的数据称为负样本，而 is_duplicate 字段可以被称为我们的目标字段

数据集基本参数如下：

数据量	404290	列数	3
句子最大长度	1169	句子最小长度	1
包含单词最大值	237	包含单词最小值	1

表 1

句子的长度和单词数量在一定程度上反映了句子包含的语义信息的量，而相同含义的句子在大部分情况下会含有相同或相近的单词数量或长度。所以可以使用句子长度和单词数量作为判断句子对意思是否相同的特征。

探索性可视化

句子长度相关特征

从基本统计数据的分析中可以看到，句子对中的句子长度和包含的词数等信息能够用来判断句子对是否能够表示相同含义。因此我选取了一些能够表示句子长度的度量，并对其进行可视化，这些特征包括：句子长度，句子字符长度，句子单词数量。

句子长度：句子字符串的长度

句子字符长度：句子中去除空格等空白符号后的字符长度

句子单词数量：用空格分割句子后得到的列表的长度。该长度为句子中单词和特殊符号的数量和。

进一步数据处理：直接获取到的长度特征为一个二元组，分别包含了第一个和第二个句子的长度特征。这个二元组不方便可视化，所以我将二元组的两个数据做比值并进一步处理来获取最终方便可视化的数据。以句子长度为例，处理过程如下：

句子长度的原始数据为二元组：（len_q1, len_q2）

将数据做比值，获得一个数据值：len_q_ratio = len_q1 / len_q2

为了反映该比值对判断的作用，我根据该比值将数据划分为较小的区间，并计算每个小区间中的数据中 is_duplicate 段的平均值，平均值越大的区间表示该区间包含越多的正样本，而平均值越小，表示该区间包含的正样本越少。

但在可视化之前，发现 len_q_ratio 比值的分布空间为[0.006711409395973154, 117.0]。因为是比值，数据应该以 1 为中心分布，因此数据分布不均匀，不利于可视化，所以考虑使用对数操作调整数据分布。

比值对数：len_q_ratio_ln = ln(len_q1 / len_q2)

该数值的分布大约在[-4，4]上。以 0.04 为小区间的划分范围，获得最终可视化分布如下：

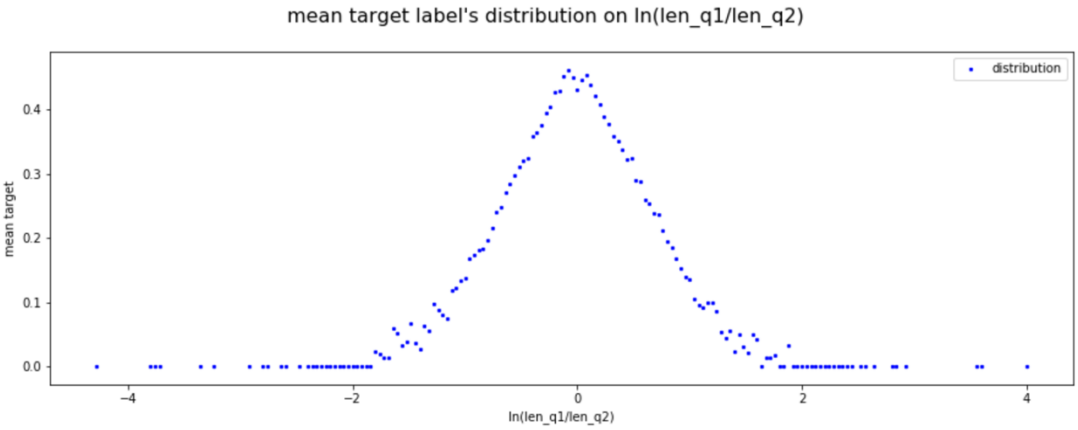

图 1

上图中和坐标为 ln(len_q1/len_q2)，是我最终选定的比值对数特征。纵坐标为每个小区间中数据 is_duplicate 字段的均值。

从图中可见，当比值对数在[-4, -2]和[2,4]之间时目标字段均值为 0，说明该句子对长度差异过大时，基本不可能是相同含义的句子。而比值对数越接近 0，目标字段均值越大，说明句子长度月相近，句子对的含义越肯能相同。因此该特征值对于模型的预测越有效果。

同上，我也准备了句子字符长度和句子单词数量对应的比值对数分布，如下：

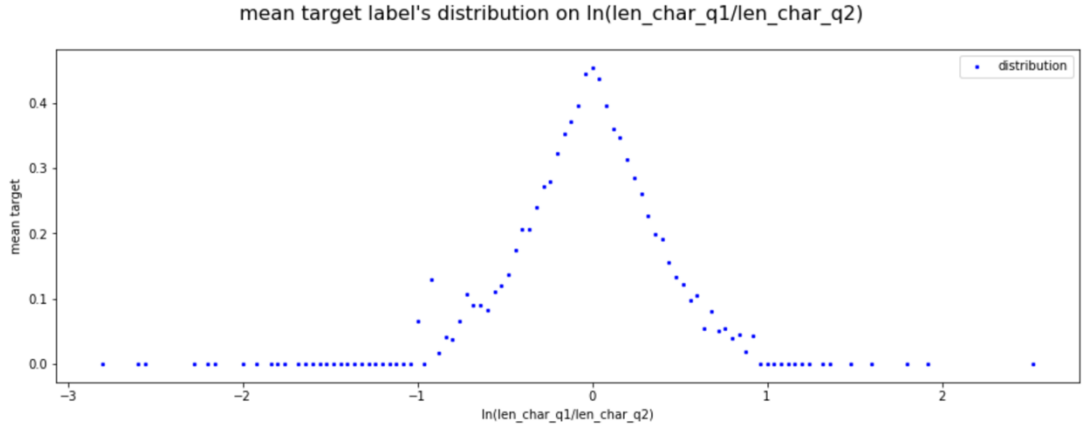

图 2

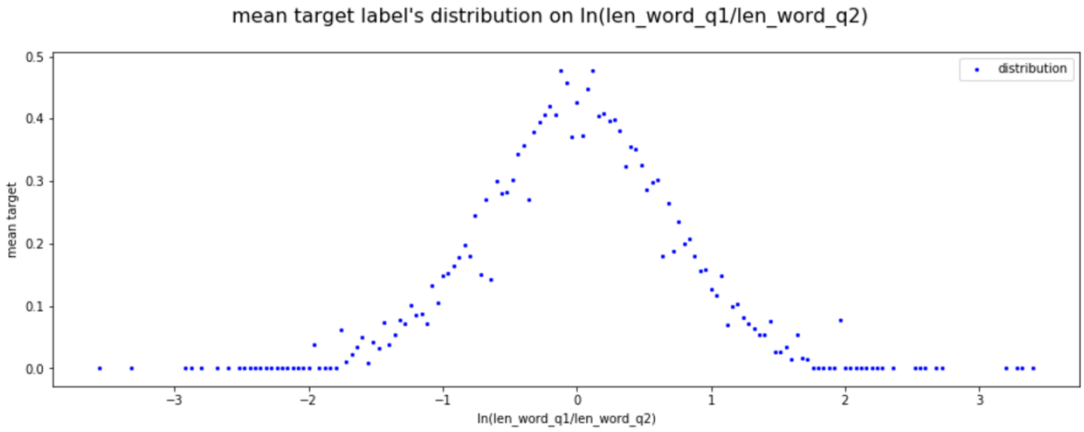

图 3

发现分布与字符串长度的分布类似。

WMD 相关特征

WMD

词移距离（WMD）是基于 word2vec 用来表示两个文本间语义距离的技术。其中 Word2Vec 技术将词映射为一个词向量，在这个向量空间中，语义相似的词之间距离会比较小。

Word2Vec 得到的词向量可以反映词与词之间的语义差别，那么如果我们希望有一个距离能够反映文档和文档之间的相似度，可以将文档距离建模成两个文档中词的语义距离的一个组合，比如说对两个文档中的任意两个词所对应的词向量求欧氏距离然后再加权求和，形式如下：

$$\sum_{i,j=1}^{n} T_{ij}c(i,j)$$

其中 c(i,j)为 i,j 两个词所对应的词向量的欧氏距离。而其中的加权矩阵 T 会保证由文档 1 中的某个词 i 移动到文档 2 中的各个词的权重之和应该与文档 1 中的这个词 i 的权重相等。同样，文档 2 中的某个词 j 所接受到由文档 1 中的各个词所流入的权重之和应该等于词 j 在文档 2 中的权重。加权代价求和后在经过线性规划求得下界之后，即可作为文档 a 中单词转移到文档 b 中单词的最短总距离，代表两个文档之间的相似度。

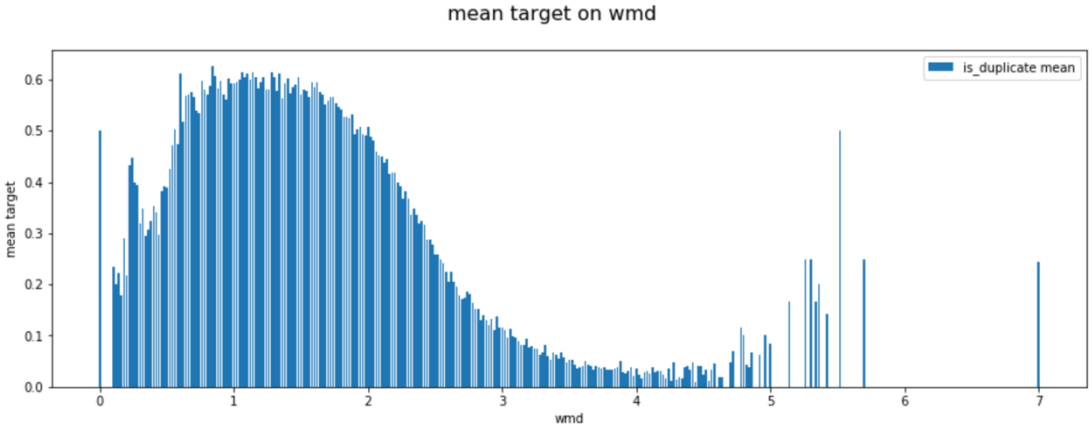

图 4

从可视化结果中可以看到在[0, 2.5]区间中，目标字段的取值较高，包含的正样本较多，而在[2.5, 5.5]区间中，取值较低，包含正样本数量较少。在[5, 6]区间中有一些目标字段均值较高的区间，但对比了 wmd 直方图，发现该区间内分布的数据量很少，所以可以忽略。

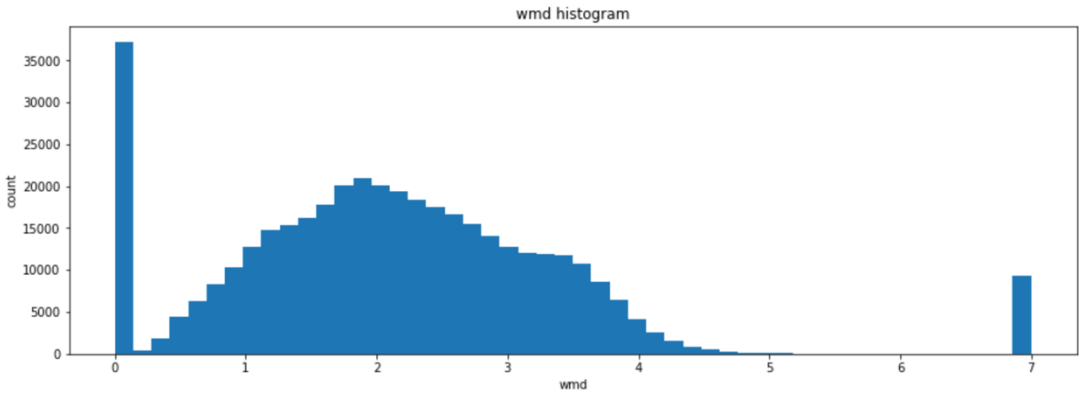

图 5

同样，对于 Normal WMD，使用同样的可视化方法得到如下可视化图片。同样可以看到，以 1.0 为划分点，可以将数据划分为目标字段均值较高和均值较低的区间。可见，Normal WMD 同样对数据有划分作用。

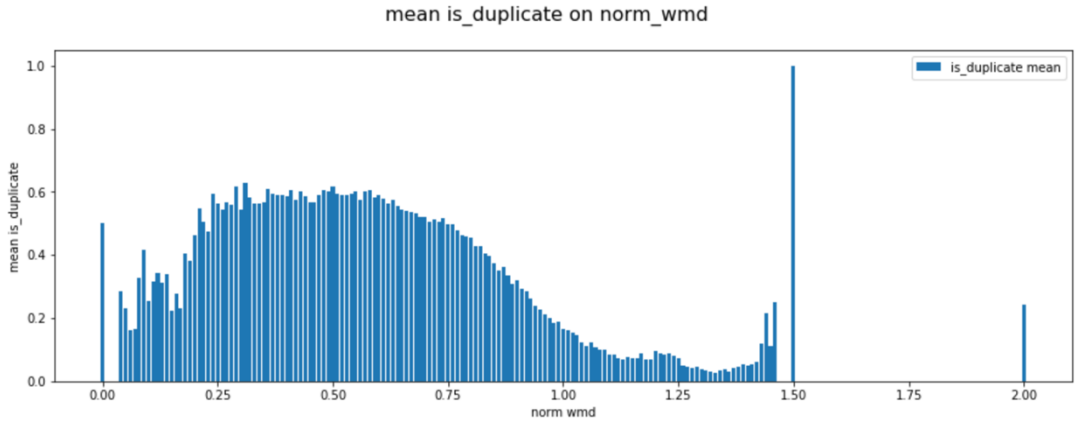

图 6

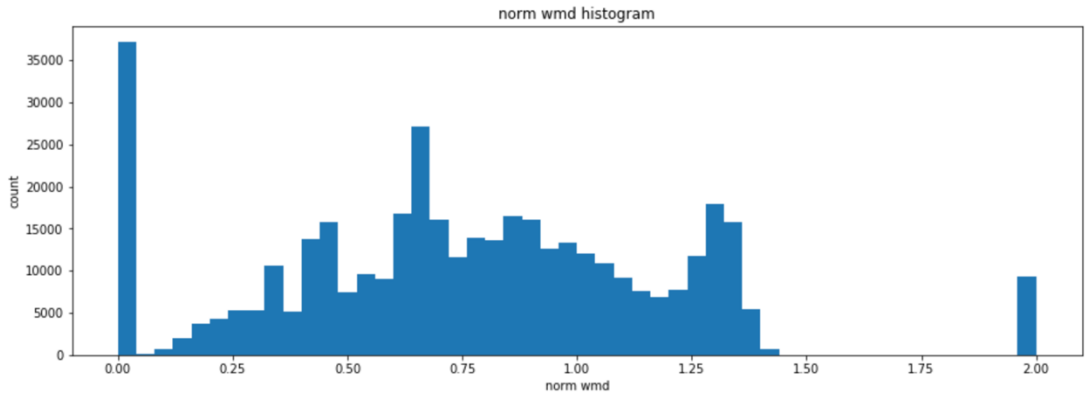

图 7

相似度相关特征

项目中使用了距离相关的特征来描述句子对之间的差异。在文档中将可视化较为典型的两个距离分布。

以余弦相似度为例，距离相关特征的计算方法如下:

以句子对包含的两个句子为语料库，计算句子中每个单词的 tf-idf 值，并以单词对应的 tf-idf 值将两个句子转化为两个向量。获取向量后计算两个向量的余弦相似度。

余弦相似性通过测量两个向量的夹角的余弦值来度量它们之间的相似性。0 度角的余弦值是 1，而其他任何角度的余弦值都不大于 1；并且其最小值是-1。

Cityblock distiance 又被称为曼哈顿距离，是向量中各个维度数值查的绝对值的和。例如在平面上，坐标（x1, y1）的点 P1 与坐标（x2, y2）的点 P2 的曼哈顿距离为

$$|x_1 - x_2| + |y_1 - y_2|.$$

Jaccard distance（雅卡尔距离）用于量度样本集之间的不相似度，其定义为 1 减去雅卡尔系数。而雅卡尔指数是用于比较样本集的相似性与多样性的统计量。雅卡尔系数能够量度有限样本集合的相似度，其定义为两个集合交集大小与并集大小之间的比例：

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

因而雅卡尔距离的定义为：

$$d_J(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}.$$

Canberra Distance(堪培拉距离)被用来衡量向量空间中两个点之间的距离，它是曼哈顿距离的加权版本。其定义公式为：

$$d(\mathbf{p},\mathbf{q}) = \sum_{i=1}^{n} \frac{|p_i - q_i|}{|p_i| + |q_i|}$$

其中 p,q 为作比较的两个向量。通常 Canberra distance 对于接近于 0（大于等于 0）的值的变化非常敏感。

Euclidean distance 既几何距离，其计算公式如下：

$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

Bray-Curtis distance 与曼哈顿距离相似，其计算公式为：

$$d_{ij} = \frac{\sum_{k=1}^{n} |x_{ik} - x_{jk}|}{\sum_{k=1}^{n} (x_{ik} + x_{jk})}$$

与曼哈顿距离相比，该距离的特点是：当比较的两个向量的所有维度数据都是正值时，距离的结果始终在[0, 1)之间，这样不用再做数据的归一化。

鉴于选用的距离特征较多，且实际可视化过程中发现大部分数据的分布有相似性，在此只展示余弦相似度和曼哈顿距离的可视化结果，如下图。

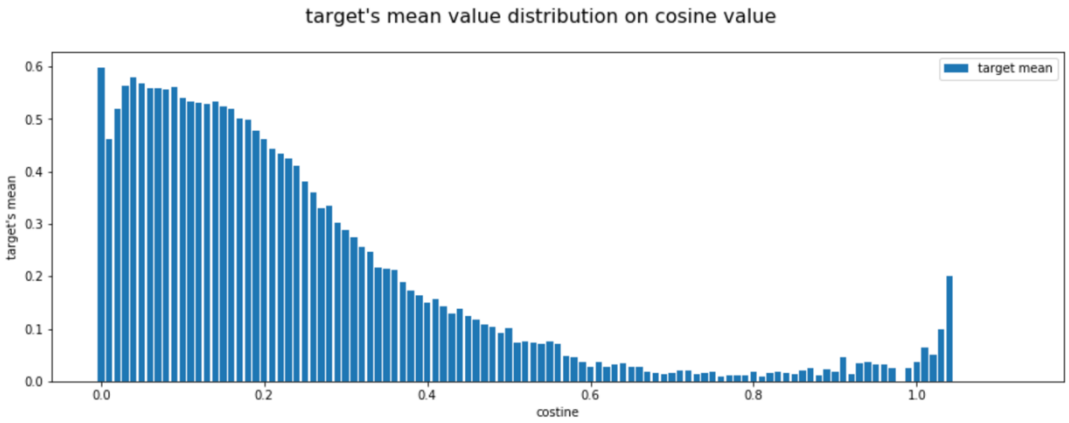
target's mean value distribution on cosine value
target mean
target's mean
costine
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.0
0.2
0.4
0.6
0.8
1.0

图 8

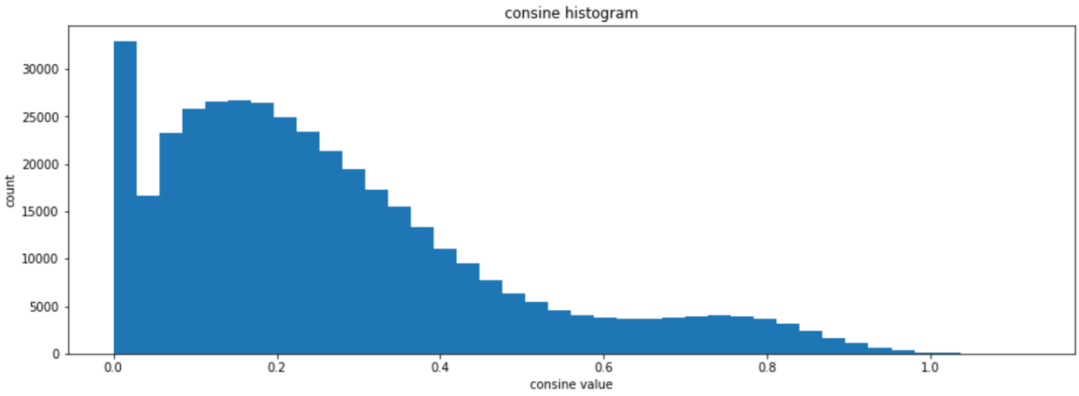
consine histogram
count
consine value
0
5000
10000
15000
20000
25000
30000
0.0
0.2
0.4
0.6
0.8
1.0

图 9

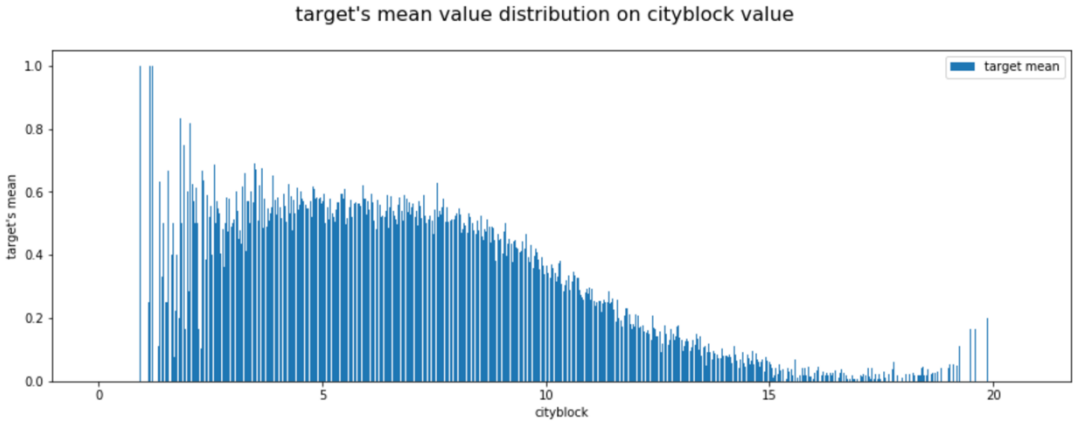
target's mean value distribution on cityblock value
target mean
target's mean
cityblock
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20

图 10

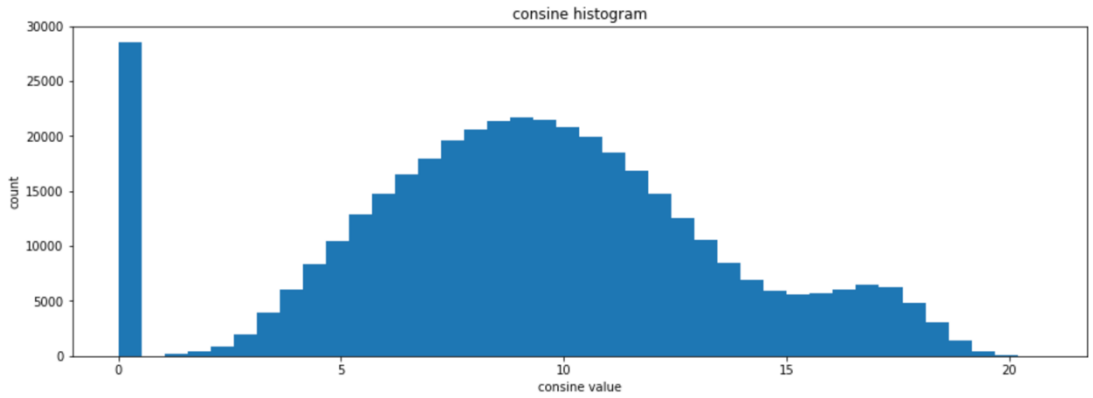

图 11

图 8 和 10 分别是目标字段的均值在余弦相似度和曼哈顿距离上的分布。两者分布大致类似，在分布区间的前半部分，目标字段均值较高，后半部分均值较低。可见特征对于分类还是有帮助的。

而通过图 9 和 11，可以看到余弦相似度和曼哈顿距离本身的分布情况。

同时发现，余弦相似度中，相似度数值较高（大于 0.4）的部分中，数据分布稀少。而目标字段均值的区分点也在0.4 左右。这说明通过余弦相似度能够找出的负样本的数量过少，根据此距离进行分类的效果肯能不会太好。而图 11 中发现，通过曼哈顿距离分布的前半部分和后半部分的数据量相近，这样通过曼哈顿距离区分出的负样本数量较多，分类效果要更好。

算法和技术

变量说明：

使用的变量是我们从原始的句子对数据中根据自然语言处理技术，获取到的变量，包括：

len_q1, len_q2, diff_len, len_char_q1, len_char_q2, len_word_q1, len_word_q2, common_words, fuzz_qratio, fuzz_WRatio, fuzz_partial_ratio, fuzz_partial_token_set_ratio, fuzz_partial_token_sort_ratio, fuzz_token_set_ratio, fuzz_token_sort_ratio, wmd, norm_wmd, cosine_distance, cityblock_distance, jaccard_distance, canberra_distance, euclidean_distance, minkowski_distance, braycurtis_distance, skew_q1vec, skew_q2vec, kur_q1vec, kur_q2vec

其中，len_q1, len_q2, diff_len, len_char_q1, len_char_q2, len_word_q1, len_word_q2, common_words 等特征是基于字符串长度和单词数量长度生成的特征。

fuzz_qratio, fuzz_WRatio, fuzz_partial_ratio, fuzz_partial_token_set_ratio, fuzz_partial_token_sort_ratio, fuzz_token_set_ratio, fuzz_token_sort_ratio 等特征是根据编辑距离算法，计算两个字符串的相似度。

wmd, norm_wmd 变量是根据词移距离计算两个句子的语义距离变量。

cosine_distance, cityblock_distance, jaccard_distance, canberra_distance, euclidean_distance, minkowski_distance, braycurtis_distance 等变量是将句子转化成向量后计算出的向量间距离。

使用模型：

解决的问题为一个典型的分类问题，所以可以使用分类或回归模型。

该项目中我尝试使用了 RandomForest, GBDT, SGD 以及 LogisticRegression 算法。最终选

择了 GBDT 算法。

GBDT(Gradient Boosting Decision Tree) 又叫 MART（Multiple Additive Regression Tree)，是一种迭代的决策树算法，该算法由多棵决策树组成，所有树的结论累加起来做最终答案。它在被提出之初就和 SVM 一起被认为是泛化能力（generalization)较强的算法。我曾尝试使用 SVM，但由于训练数据量较大，SVM 训练时间非常长，最后没有尝试使用该算法。

参数说明：

在实验中使用了如下一些参数。

n_estimators: 也就是弱学习器的最大迭代次数，或者说最大的弱学习器的个数。一般来说 n_estimators 太小，容易欠拟合，n_estimators 太大，又容易过拟合，一般选择一个适中的数值。默认是 100。在实际调参的过程中，常常将 n_estimators 和 learning_rate 一起考虑。

learning_rate: 即每个弱学习器的权重缩减系数 ν，也称作步长，对于同样的训练集拟合效果，较小的 ν 意味着需要更多的弱学习器的迭代次数。一般来说，可以从一个小一点的 ν 开始调参，默认是 1。

subsample:子采样，取值为(0,1]。GBDT 的采样是不放回抽样。如果取值为 1，则全部样本都使用，等于没有使用子采样。如果取值小于 1，则只有一部分样本会去做 GBDT 的决策树拟合。选择小于 1 的比例可以减少方差，即防止过拟合，但是会增加样本拟合的偏差，因此取值不能太低。推荐在[0.5, 0.8]之间，默认是 1.0，即不使用子采样。

决策树最大深度 max_depth: 默认可以不输入，如果不输入的话，默认值是 3。一般来说，数据少或者特征少的时候可以不管这个值。如果模型样本量多，特征也多的情况下，推荐限制这个最大深度，具体的取值取决于数据的分布。常用的可以取值 10-100 之间。

内部节点再划分所需最小样本数 min_samples_split: 这个值限制了子树继续划分的条件，如果某节点的样本数少于 min_samples_split，则不会继续再尝试选择最优特征来进行划分。 默认是 2.如果样本量不大，不需要管这个值。如果样本量数量级非常大，则推荐增大这个值。

叶子节点最少样本数 min_samples_leaf: 这个值限制了叶子节点最少的样本数，如果某叶子节点数目小于样本数，则会和兄弟节点一起被剪枝。 默认是 1,可以输入最少的样本数的整数，或者最少样本数占样本总数的百分比。如果样本量不大，不需要管这个值。如果样本量数量级非常大，则推荐增大这个值。

基准模型：

选取了 RandomForest，GBDT，LogsitcRegression 和 SGD 四个分类模型对该问题进行预测。基准模型为这四个模型的默认参数，并给定一个固定的 random_state 以方便的进行后续的模型对比。

训练数据和测试数据生产过程如下：

Quora 给定的训练数据集中包含约 40 万条数据。通过我们之前介绍的方法生成变量后，又对数据进行了清洗。

清洗过程主要去除了数据中的空值，NaN 值以及无限大数值。最后对得到的数据集进行去重，获取到最终可以使用的数据。

最后对数据进行切割。以 1：4 的比例分割数据，获取到测试数据集和基本训练数据集，再对基本训练数据集以 1：4 的比例分割，得到最终的验证数据集合训练数据集。

得到的数据量如下:

Train Set	Test Set	Validation Set	All
251436	78575	62860	392871

表 2

训练基本模型时，使用训练数据集新型模型训练，使用测试数据集对模型进行测试。验证数据集将会在后面的模型调优过程中使用。

因为是基准模型，使用默认值参数进行训练。得到模型的效果如下结果如下：

model	ROC AUC Score		F1-Score		Accuracy	
	Train	Test	Train	Test	Train	Test
RandomForest	0.987073	0.701551	0.985816	0.618121	0.989503	0.732165
GBDT	0.735222	0.730302	0.674664	0.669399	0.734828	0.729203
LogsitcRegression	0.661427	0.662177	0.577043	0.578439	0.681540	0.681716
SGD	0.500000	0.500000	0.000000	0.000000	0.626821	0.626817

表 3

从表 3 可以看出，除 SGD 外，其他几个模型都有一定的预测效果。其中效果最好的是 GBDT 模型，训练和测试的准确率都很高。而 RandomForest 明显出现过拟合，而 SGD 则明显欠拟合。考虑是没有进行对应参数调节造成的。

方法

数据预处理

预处理过程在上一章已经介绍过，在此不再赘述。

没有进行特征变换，在之前的特征可视化过程中可以看到，使用到的主要特征包括：字符串长度相关特征，向量化距离特征，词移距离，模糊匹配特征。这些特征的分布范围基本都在（-10， 10）之间。模糊匹配由于使用的是百分比结果，结果分布也都在（0，100）间。我认为这个分布范围的区别不是特别大，因此不需要进行进一步的特征变换。

数据的异常和一些特殊值已经在数据处理过程中介绍过，主要是对 Nan，inf 等异常值进行了处理。由于其他特征分布范围不大，分布相对均匀，所以没有做额外的异常值或利群店的处理。

训练数据和测试数据生产过程如下：

Quora 给定的训练数据集中包含约 40 万条数据。通过我们之前介绍的方法生成变量后，又对数据进行了清洗。

清洗过程主要去除了数据中的空值，NaN 值以及无限大数值。最后对得到的数据集进行去重，获取到最终可以使用的数据。

最后对数据进行切割。以 1：4 的比例分割数据，获取到测试数据集和基本训练数据集，再对基本训练数据集以 1：4 的比例分割，得到最终的验证数据集合训练数据集。处理后得到的数据量可见表 2.

执行过程

由于该项目属于分类问题，开始选取了 4 中进行分类的模型。在模型训练过程中要进行两方面工作：1，分别对每个模型进行参数调优。2，从四个模型中选择最优的模型。

模型调优：

调优过程使用了 sklearn 库中的 GridSearch 工具进行。对四个模型，分别使用 GridSearch 尝试模型中各种参数组合的结果。这部分的相关代码在 data_check.ipynb 文件中。

调优过程中使用参数如下：

GBDT 模型调优所用的参数已经在上一部分的方法与技术小节中介绍过，不再赘述。

Random Forest 中调节的参数包括：n_estimators， max_depth， min_samples_split,， max_features，min_samples_leaf。其中 max_features 参数代表训练过程中每棵决策树训练时使用的特征数量。默认是"auto",意味着划分时最多考虑 N 平方根个特征；如果是"log2"意味着划分时最多考虑 log2N 个特征；如果是"sqrt"或者"auto"意味着划分时最多考虑 N 平方根个特征。如果是整数，代表考虑的特征绝对数。如果是浮点数，代表考虑特征百分比。其中 N 为样本总特征数。

对 SGD 模型使用的参数包括：max_iter，tol，penalty，loss。其中

Max_iter 表示最大的训练迭代次数。

tol 表示迭代停止标准，如果迭代过程中损失函数值缩小的数值小于 tol 时，即可终止迭代。

penalty 表示惩罚方式,字符串型；默认为’l2’;其余有’none’,‘l1’,‘elasticnet’。

loss 表示损失函数选择项，字符串型；默认为’hinge’即 SVM；'log’为逻辑回归。

对逻辑回归模型的训练参数包括：

penalty 参数可选择的值为"l1"和"l2".分别对应 L1 的正则化和 L2 的正则化，默认是 L2 的正则化。

solver 参数决定了我们对逻辑回归损失函数的优化方法，有 4 种算法可以选择，分别是

a) liblinear：使用了开源的 liblinear 库实现，内部使用了坐标轴下降法来迭代优化损失函数。

b) lbfgs：拟牛顿法的一种，利用损失函数二阶导数矩阵即海森矩阵来迭代优化损失函数。

c) newton-cg：也是牛顿法家族的一种，利用损失函数二阶导数矩阵即海森矩阵来迭代优化损失函数。

d) sag：即随机平均梯度下降，是梯度下降法的变种，和普通梯度下降法的区别是每次迭代仅仅用一部分的样本来计算梯度，适合于样本数据多的时候。

具体的训练代码可以参考 data_check.ipynb 文件。基本的步骤是对各个模型使用 GridSearch 以及训练数据进行最优参数的挑选。最后使用测试数据集比较四个模型得到的最优结果的性能。在最后模型性能比较的过程中我们使用了 accuracy 和 f1-score 来评价模型的性能。最终比较结果如图 12。每个小图中横坐标代表了每个模型使用的不同度量指标，包括准确率和 f1 score，并且按照训练数据和测试数据划分为训练的指标结果和测试的指标结果。具体指标划分可以参见图例中的说明。纵坐标为各个指标的分数。准确率和 f1 score 分布区间都是[0, 1]之间，所以两个指标可共用同一个纵坐标轴。

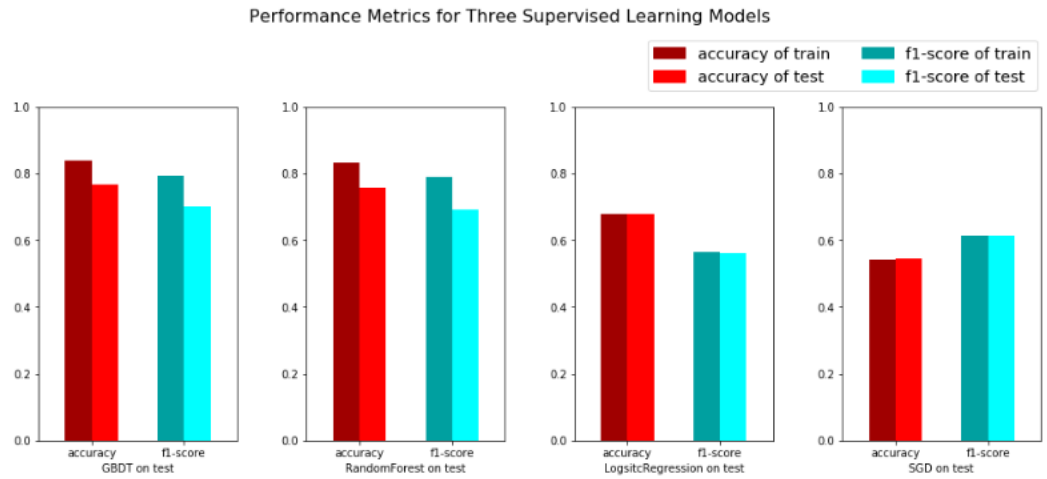

图 12

从最后结果的比价中可见，性能最好的还是 GBDT 经过调优后的结果。而最终该模型的调优参数如下：

learning_rate =0.1， n_estimators =140， max_depth =14，min_samples_split =550，max_features ='sqrt'， subsample=0.85

而其他模型的最终结果请参见 data_check.ipynb 文件。

结果

模型的评价与验证

最终模型是在项目开始时选取的几个模型算法进行参数调优后再选择最优的模型获得的。整个选择过程包括：模型调优和最终模型选择。初始模型选取了四个典型的用来进行分类的模型，包括：随机森林，GBDT，SGD 和逻辑回归分类模型。

然后需要对每个模型进行参数调优，以获得每个模型中最优的模型结果。在这个过程中，我们使用 sklearn 中的 Grid Search 调试每个模型的参数，最后得到每个模型的最优结果。具体的调试过程和使用的参数可以参考方法部分的执行方法的小节，具体代码在 data_check.ipynb 文件中。

最后，在几个模型中使用 accuracy score 和 f1 score 作为评价标准，并可视化各个最优模型所得到的对应指标结果。可视化结果的可视化如图 12 所示。具体的代码参考 data_check.ipynb 文件。

最终获得了用来分类的模型结果。最终模型使用的是 GBDT 算法，其参数调优结果如下：

learning_rate =0.1， n_estimators =140， max_depth =14，min_samples_split =550，max_features ='sqrt'， subsample=0.85

而模型的性能结果为：

model	F1-Score		Accuracy	
	Train	Test	Train	Test
GBDT	0.793530	0.700332	0.840546	0.768013

表 4

我认为模型结果比较合理，与项目开始时定义的基本模型（表 3）相比，最终模型的性

能有明显提高。说明模型的调优和选择是合理的。

合理性分析

从上一章节中已经描述了模型调优和最终模型选择。初始模型选取了四个典型的用来进行分类的模型，包括：随机森林，GBDT，SGD 和逻辑回归分类模型。

然后需要对每个模型进行参数调优，这个过程中使用了 Grid Search 来筛选最优参数组合。在最后比较四个模型，得到了最终的模型。

而比较了基准模型和最终得到的模型的结果，可以看到最终模型比基准模型的效果要好，选定的三个对比指标都有所提高。

V. 项目结论

结果可视化

总结本次项目，总共包含这几个步骤：数据特征分析与可视化，初始模型生成，模型训练以及模型选择。在这几个步骤中，运用了训练模型的相关技术，包括：数据准备与清理，变量生成，参数调优，最终模型选择这几个步骤。

数据的准备和清理过程主要是利用 pandas 库，对读入的多维数据进行异常值处理。主要是去除了数据中 none 值或无限大的数值。

变量生成的过程中主要采用了可视化方法，将变量的分布情况，以及目标字段的均值分布在各个变量上展示了出来，分析了使用的变量对于分类是否有明显的贡献。而生成的变量包括：句子字符串长度相关变量，句子向量化后向量间距离的相关变量，句子词移距离相关变量，以及句子模糊匹配产生的相关变量。具体变量和可视化结果分析请参见前文中可视化的章节。在这一部分中，使用了 matplotlib 进行了数据的可视化，代码参见 data_analyse.ipynb 文件中代码。同时，为了产生相关特征，使用了自然语言处理的相关工具，包括词移距离计算工具 gensim，模糊匹配工具包 fuzzywuzzy，自然语言处理工具包 nltk 等，具体代码请参考 features.py 文件。

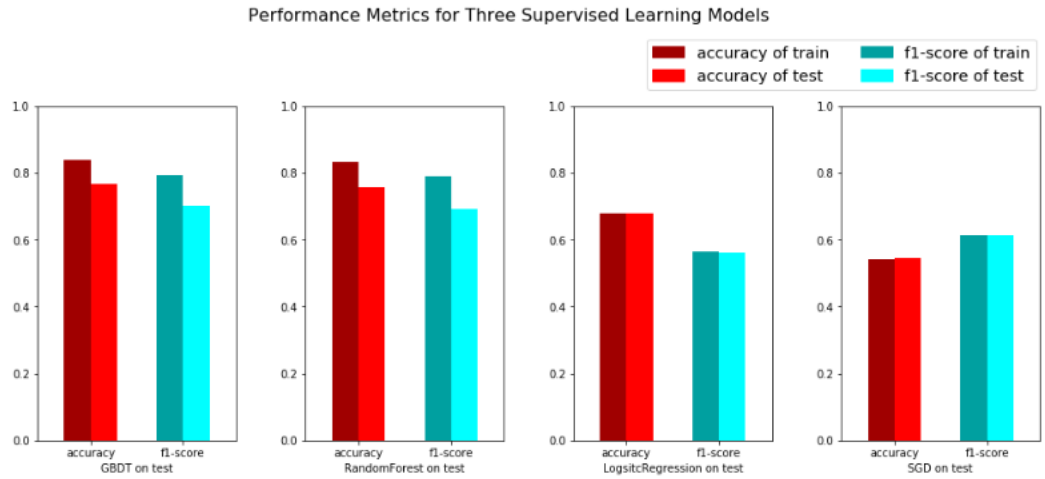

图 13

参数调优过程使用了 Grid Search 工具，针对各个模型参数，对模型进行了调整，得到了各个算法模型的较好结果。最后对结果进行了可视化，如图 13 所示。从结果中可以看到，对比基准模型数据（表 3），调试后各个模型的性能指标都有所提高。

最优模型的选取，通过图 13 的可视化分析，可以看到，比较四个模型，GBDT 的效果是最好的，该模型在测试数据上的结果是最好的，而且训练数据集的结果与测试数据集的结果差异不大，没有明显的过拟合现象。因此选用该模型作为最终模型。

对项目的思考

整个项目的流程在上一章节中已经介绍。

个人觉得项目中比较有趣的部分包括以下两点：

1. 各个变量可视化过程。在该过程中需要考虑如何可视化才能反映变量对目标字段是否有区分作用，因而使用了目标字段的平均值在变量上的分布情况来反映其区分效果。同时，还需要考虑变量本身的分布情况，以及如何转化才能有明显的可视化效果，因而需要对变量进行各种转化。例如句子长度比例的分布范围不对称，需要进行对数操作后才能获得较好的分布范围，也更有助于可视化。最后，可视化的过程其实也是一个变量提取的过程，例如句子长度比例的变量，在找到比例对数的表现形式后，该比例对数其实也是能够反映比例特征的最好的变量。
2. 寻找特征的过程。在这一过程中可以了解各种自然语言处理相关的技术，并能够不断尝试新的算法和技术。有助于了解更多的机器学习的相关算法和技术。

需要作出的改进

改进方向：

1. 尝试使用更多的自然语言处理相关的特征。
2. 对字符串进行多样化的分割，例如以空格分割；以空格加特殊字符分割；分割后去除停用词；分割后保留标点符号，标点符号视作独立单词等等