机器学习纳米学位

毕业项目：自然语言处理-文档归类

白佳亮 优达学城

2018 年 8 月 24 日

I. 问题的定义

项目概述

文档分类是图书馆学、信息学和计算机科学中的一个问题。其任务是将一个文档分配到一个或者多个类别中。它可以是通过人工分类完成的，也可以是通过计算机算法实现的。多数通过人工的文档分类问题一直属于图书馆学的领域，而通过算法实现的文档分类问题则多属于信息学和计算机科学的领域。这些问题之间是有相同的部分的，所以有一些对文档分类的跨学科研究。需要被分类的文档有可能是纯文本，图片，音乐等等。每一种文档都有其独特分类问题。根据特殊的文档做研究，文档分类可以细分成文本分类，图片分类等等。文档自动分类的任务可以分为三类：监督式学习的文档分类，这需要人工反馈数据的一些外在机制。非监督式学习的文档分类（也被称作文档聚类），这类任务完全不依靠外在人工机制。和半监督式学习的文档分类，是前两类的结合，它其中有一部分的文档是由人工标注的。自动的文档分类可以使用的算法有：朴素贝叶斯分类器、支持向量机(SVM)、神经网络、决策树比如 ID3 或 C4.5、自然语言处理方法等。[1]

自然语言处理（英语：natural language processing，缩写作 NLP）是人工智能和语言学领域的分支学科。此领域探讨如何处理及运用自然语言；自然语言认知则是指让电脑“懂”人类的语言。自然语言生成系统把计算机数据转化为自然语言。自然语言理解系统把自然语言转化为计算机程序更易于处理的形式。[2]

随着技术的进步，选择使用深度学习方法配合自然语言处理(后面简称 NLP)，可以很好地本项目的文档分类问题。

问题陈述

近几年来，借助深度学习概念和性能强劲的硬件平台，一些顶尖学者深入开展了针对词向量的研究。并以词向量为基础，方便引入机器学习模型对文本进行分类、情感分析、预测、自动翻译等，也就是说，可以将这些词向量可以作为机器学习模型的输入数值向量。在认清了一些最简单的词向量(如独热编码 one-hot encoder)的缺点后，Mikolov、Socher 等人提出了 Word2Vec、GloVec 等词向量模型，很好的解决了用维数较少的向量表达词以及词之间关联性的问题。类似的，句子、段落以及文章也可以引入向量的概念进行表达，称之为 Doc2Vec。[3]

具体到此项目，这是一个文本多分类问题，需要基于词向量来进行建模。

可以将文档数据转化成为不同的词向量模型，进而将文档用向量(如 TF-IDF)的形式表示出来，并以此为输入数据，训练机器学习模型或深度学习模型，进而实现文档分类。

评价指标

通过训练模型和优化超参，将文档分类模型在整个测试集（全集）上的准确率提高到 85%以上。

说明：

此处有多种可以使用的多分类评估指标：如准确率(accuracy)、召回率(recall)、精确率(precision)、F1_score(准确率和召回率的调和平均)等

对于本项目而言，个人选择用准确率 accuracy 作为评估标准，因为它是我们最常见的多分类评价指标，如下所示:

$$accuracy = (TP + TN) / (P + N)$$

很直观、很容易理解，就是被分对的样本数除以所有的样本数，通常来说，正确率越高，分类器越好。

II. 分析

数据的探索

分类文本数据将首先项目描述文档中推荐使用经典的 20 类新闻包来训练词向量模型，里面大约有 20000 条新闻，比较均衡地分成了 20 类，是比较常用的文本数据之一。数据下载地址为：http://www.qwone.com/%7Ejason/20Newsgroups/

词向量训练完成之后，可以将文档数据表示成为加权词频 TF-IDF 向量(或直接使用 Word2Vec/Doc2Vec 模型等形式)，并以此为输入数据，训练机器学习模型或深度学习模型，进而实现文档分类。

本项目采用的数据集为 20news-bydate 类别下的所有文档。其中包括：训练集文档 20 个类目共计 11314 个，测试集文档为同样的 20 个类目共计 7532 个。训练集和测试集所有的文档已经被打好了 label (标明了类别)。

文档大致为包含了主题(Subject)、Email 地址(From)、组织机构(Organization) 、正文(main body)。一般情况下，主题和正文中的一些关键词语可以用来作为文档分类的依据。另外，文档中所含的各类标点符号，通常对分类文档没有作用，应该被忽略。

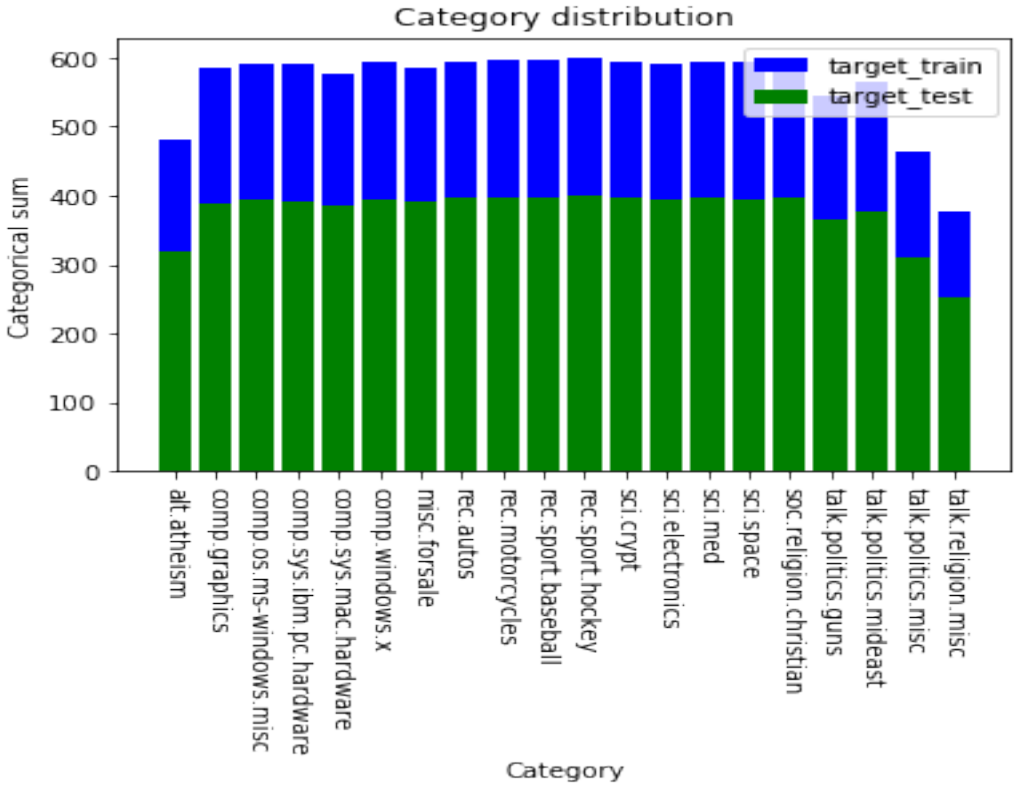
```
From: admiral@jhunix.hcf.jhu.edu (Steve C Liu)
Subject: Baseball Stats
Organization: Homewood Academic Computing, Johns Hopkins University, Baltimore, Md, USA
Lines: 17
Distribution: usa
Expires: 5/5/93
NNTP-Posting-Host: jhunix.hcf.jhu.edu
Summary: 1992 EWB II Stats wanted

        Hello, my friends and I are running the Homewood Fantasy Baseball
League (pure fantasy baseball teams). Unfortunely, we are running the league
using Earl Weaver Baseball II with the Comm. Disk II and we need the stats
for the 1992 season. (Preferably the 1992 Major League Stat Disk) We have
the '92 total stats but EWB2 needs the split stats otherwise we have 200
inning games because the Comm. Disk turns total stats into vs. L's stats
unless you know both right and left -handed stats.

        So, if anyone has the EWB2 '92 Stat Disk please e-mail me!

-------------------------------------------------------------------------------
|Admiral Steve C. Liu        Internet Address: admiral@jhunix.hcf.jhu.edu|
|"Committee for the Liberation and Intergration of Terrifying Organisms  |
|and their Rehabilitation Into Society" from Red Dwarf - "Polymorph"      |
|****The Bangles are the greatest female rock band that ever existed!****|
|   This sig has been brought to you by... Frungy! The Sport of Kings!    |
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

探索性可视化

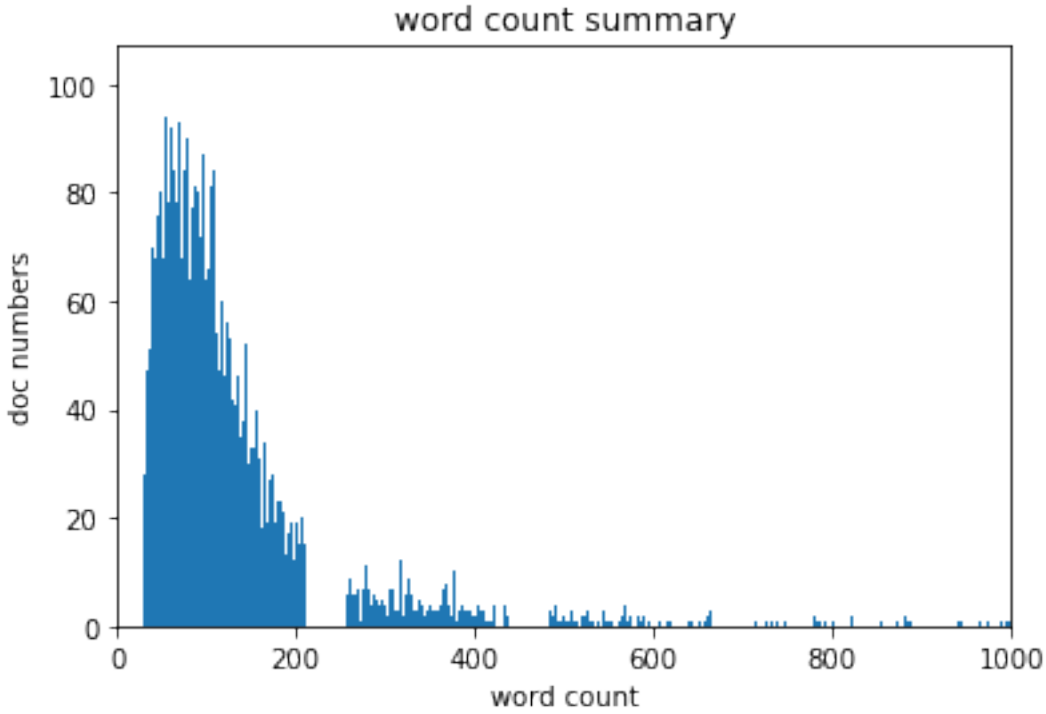
训练集和测试集中均包含以下 20 个类目：

```
['alt.atheism',
 'comp.graphics',
 'comp.os.ms-windows.misc',
 'comp.sys.ibm.pc.hardware',
 'comp.sys.mac.hardware',
 'comp.windows.x',
 'misc.forsale',
 'rec.autos',
 'rec.motorcycles',
 'rec.sport.baseball',
 'rec.sport.hockey',
 'sci.crypt',
 'sci.electronics',
 'sci.med',
 'sci.space',
 'soc.religion.christian',
 'talk.politics.guns',
 'talk.politics.mideast',
 'talk.politics.misc',
 'talk.religion.misc']
```

对其中文档数量做了统计，每个类目(用横坐标表示)下的文档数（用纵坐标表示）大致比较均衡，且每个类目在测试集和训练集中的占比大致相当，即此问题为均匀分类问题。



训练集中，对每个文档的单词数量做了统计，横坐标表示文档的单词量，纵坐标表示某单词量对应的文档数量。经观察发现，字数最多的文档有 8066 个单词，字数最少的文档有 12 个单词，大多数文档都在 200 个单词以下。如下图所示:



算法和技术

将本项目分成“文档表示”阶段和“文档分类”阶段。

“文档表示”阶段将尝试以下几种解决方案：
方式一：使用词袋模型 BoW 来表示每篇文档。建立词典，每篇文档表示成特征词的频率向量或者加权词频 TF-IDF 向量，进而使用机器学习分类模型进行训练。
方式二：使用 Word2Vec 或 GloVe 即词向量模型表示每篇文档。其中，第一步需要将每个单词表示成词向量形式，第二步需要将文档中每个词的向量来表达整个文档(可以采取和平均池化相似的方案)

“文档分类”阶段将尝试使用卷积神经网络（如 TextCNN）或其他传统机器学习方法(如逻辑回归)来实现。

以下是一些关于相关技术的介绍:

TF-IDF 及其算法

TFIDF 的主要思想是：如果某个词或短语在一篇文章中出现的频率 TF 高，并且在其他文章中很少出现，则认为此词或者短语具有很好的类别区分能力，适合用来分类。TFIDF 实际上是：TF * IDF，TF 词频(Term Frequency)，IDF 反文档频率(Inverse Document Frequency)。[4]

在一份给定的文件里，词频（term frequency，TF）指的是某一个给定的词语在该文件中出现的频率。这个数字是对词数(term count)的归一化，以防止它偏向长的文件。（同一个词语在长文件里可能会比短文件有更高的词数，而不管该词语重要与否。）对于在某一特定文件里的词语 t_i 来说，它的重要性可表示为：

$$\mathrm{tf_{i,j}} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

以上式子中 $n_{i,j}$ 是该词t_i 在文件d_j中的出现次数，而分母则是在文件d_j中所有字词的出现次数之和。

逆向文件频率（inverse document frequency，IDF）是一个词语普遍重要性的度量。某一特定词语的 IDF，可以由总文件数目除以包含该词语之文件的数目，再将得到的商取对数得到：

$$\mathrm{idf_i} = \log \frac{|D|}{|\{j : t_i \in d_j\}|}$$

其中

|D|：语料库中的文件总数

$|\{j : t_i \in d_j\}|$：包含词语t_i的文件数目（即$n_{i,j} \neq 0$的文件数目）如果该词语不在语料库中，就会导致被除数为零，因此一般情况下使用$1 + |\{j : t_i \in d_j\}|$

然后

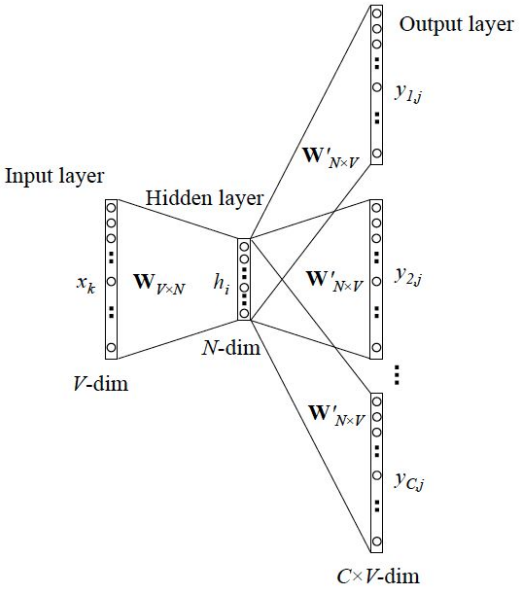
$$\mathrm{tfidf_{i,j}} = \mathrm{tf_{i,j}} \times \mathrm{idf_i}$$

某一特定文件内的高词语频率，以及该词语在整个文件集合中的低文件频率，可以产生出高权重的 TF-IDF。因此，TF-IDF 倾向于过滤掉常见的词语，保留重要的词语。

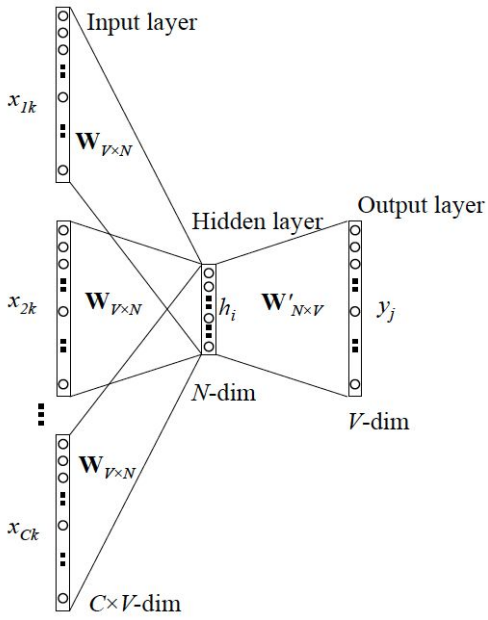
词向量模型 Word2Vec 及其算法

词向量模型中主要要介绍的是 word2vec 算法。可以理解为 word2vec 就是将词表征为实数值向量的一种高效的算法模型，其利用深度学习的思想，可以通过训练，把对文本内容的处理简化为 K 维向量空间中的向量运算，而向量空间上的相似度可以用来表示文本语义上的相似。word2vec 算法，在不断发展沉淀之后，得到两个机器学习模型：Skip Gram Model 和 CBOW(Continuous Bag of Words)。[5]

Skip-gram 的神经网络结构：



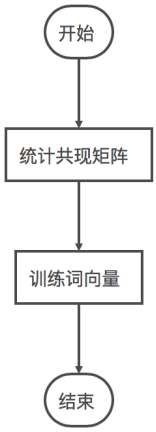
CBoW 的神经网络结构：



两种方法的区别是：Skip-gram 是预测一个词的上下文，而 CBOW 是用上下文预测这个词。隐层的激活函数其实是线性的，相当于没做任何处理（这也是 Word2vec 简化之前语言模型的独到之处），我们要训练这个神经网络，用反向传播算法，本质上是链式求导，当模型训练完后，最后得到的其实是神经网络的权重。需要提到一点的是，这个词向量的维度（与隐含层节点数一致）一般情况下要远远小于词语总数 V 的大小，所以 Word2vec 本质上是一种降维操作——把词语从 one-hot encoder 形式的表示降维到 Word2vec 形式的表示。

GloVe 模型

GloVe 模型的目标是：进行词的向量化表示，使得向量之间尽可能多地蕴含语义和语法的信息。使用的方法为：首先基于语料库构建词的共现矩阵，然后基于共现矩阵和 GloVe 模型学习词向量。如下图所示：[6]



逻辑回归(Logistic Regression, LR)模型

逻辑回归(Logistic Regression, LR)模型其实仅在线性回归的基础上，套用了一个逻辑函数，但也就由于这个逻辑函数，使得逻辑回归模型成为了机器学习领域一颗耀眼的明星，更是计算广告学的核心。 [7]

多项式朴素贝叶斯模型

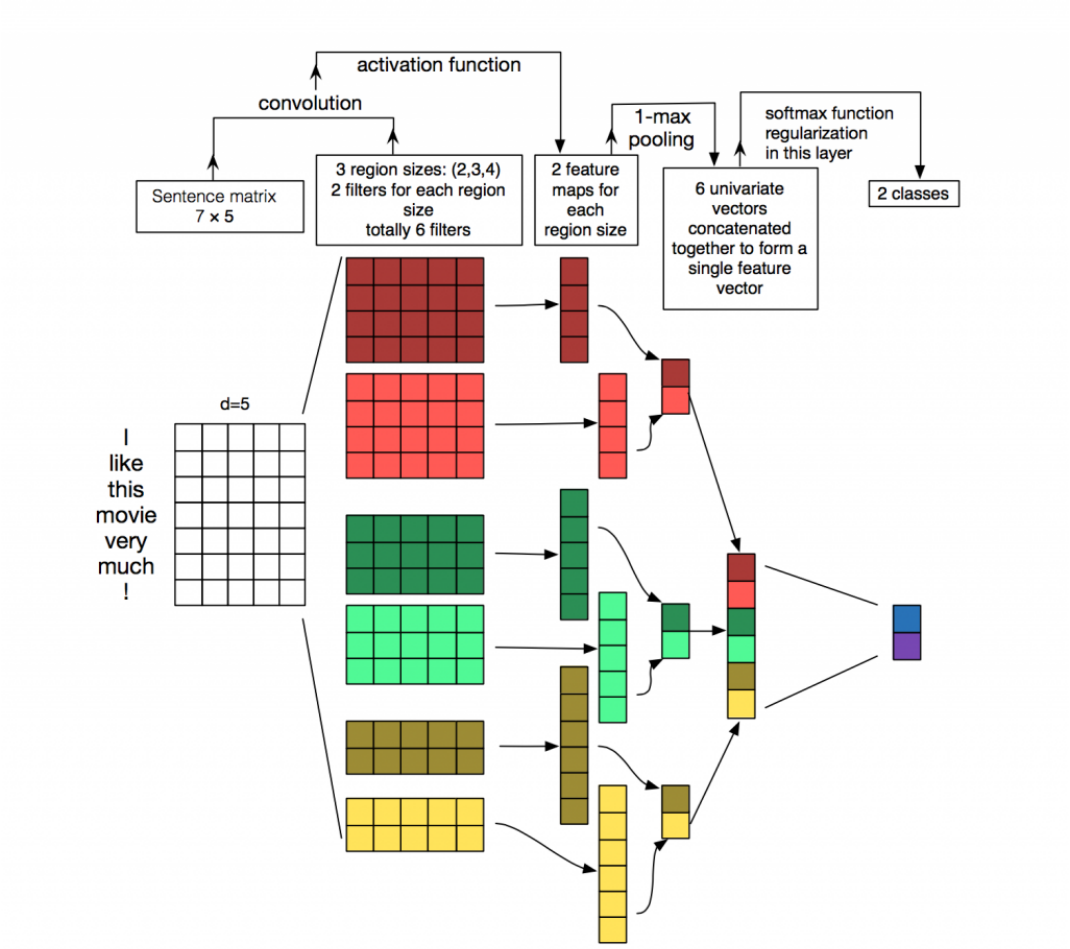
多项式朴素贝叶斯(Multinomial Naive Bayes),即所有特征都是离散型的随机变量(例如在做文本分类时所使用的词向量就是离散型的)。在此项目中，所有维度的特征都是离散型的随机变量，比较适合解决此类问题。

支持向量机 SVM 模型

支持向量机（SVM）算法被认为是文本分类中效果较为优秀的一种方法，它是一种建立在统计学习理论基础上的机器学习方法。该算法基于结构风险最小化原理，将数据集合压缩到支持向量集合，学习得到分类决策函数。这种技术解决了以往需要无穷大样本数量的问题，它只需要将一定数量的文本通过计算抽象成向量化的训练文本数据，提高了分类的精确率。

TextCNN

TextCNN 是利用卷积神经网络对文本进行分类的算法，由 Yoon Kim 在“Convolutional Neural Networks for Sentence Classification”一文中提出，是 2014 年的算法。模型结构如下图所示: [8]



卡方检验

卡方检验是一种用途很广的计数资料的假设检验方法。它属于非参数检验的范畴，主要是比较两个及两个以上样本率（构成比）以及两个分类变量的关联性分析。其根本思想就是在于比较理论频数和实际频数的吻合程度或拟合优度问题。它在分类资料统计推断中的应用，包括：两个率或两个构成比比较的卡方检验；多个率或多个构成比比较的卡方检验以及分类资料的相关分析等。卡方检验是以 χ2 分布为基础的一种常用假设检验方法，它的无效假设 H0 是：观察频数与期望频数没有差别。

该检验的基本思想是：首先假设 H0 成立，基于此前提计算出 χ2 值，它表示观察值与理论值之间的偏离程度。根据 χ2 分布及自由度可以确定在 H0 假设成立的情况下获得当前统计量及更极端情况的概率 P。如果 P 值很小，说明观察值与理论值偏离程度太大，应当拒绝无效假设，表示比较资料之间有显著差异；否则就不能拒绝无效假设，尚不能认为样本所代表的实际情况和理论假设有差别。

基准模型

「文档表示阶段使用 TF-IDF 模型，文档分类阶段使用逻辑回归模型」

可执行代码详见附件[A] Benchmark_Model_1_Original.py

Step1：使用 sklearn.datasets 中提供的 fetch_20newsgroups_vectorized 读取数据（训练集和测试集），并直接转化为向量化的样本数据(全 feature 的 TF-IDF 向量)
此处转换后的 TF-IDF 向量化样本数据，未进行文字预处理(如去掉标点符号、停词、词干化等)，可能会导致基准模型的准确率不高，后续步骤中将使用 GridSearch 进行优化。

Step2：使用 LogisticRegression 作为多分类模型，定义完整的模型超参，并对训练集进行训练 LogisticRegression 多分类的模型完整超参列表为：

```
C=1, class_weight=None, dual=False, fit_intercept=True,intercept_scaling=1, max_iter=3, multi_class='multinomial',n_jobs=1,
penalty='l2', random_state=18, solver='sag',tol=0.0001, verbose=0, warm_start=False
```

需要作出的几点说明有：

1. 样本集数据量较大，因此 solver 选择'sag'可以使运算更快(小数据集可选择'liblinear')

2. 由于 solver 选择了'sag'，对应地，penalty 选择 L2;
3. multi_class 选择了 multinomial, 稍后将于使用 ovr 时的准确率做一些比较(使用 GridSearchCV)

Step3: 使用训练好的模型在测试集上进行测试并计算 accuracy

由此得出，基准模型的准确率为 0.72，模型表现一般，还有较大的调优空间。本报告稍后的部分中，通过超参调优，将基准模型的文档分类在测试集上的准确率提高到 85%以上。超参调优将主要使用 K 折交叉验证+网格搜索(KFold GridSearchCV)来实现，即通过将训练集中固定比例的样本循环用作验证集的方式，寻找出最优的超参组合。

III. 方法

数据预处理

对于基准模型(TF-IDF + LR)的改良过程，我对训练集和测试集文档使用了 NLTK 工具包的 stopwords 和 punkt 模块进行了去除标点符号并分词、去除词袋中停词和单词词干化三种预处理。随后使用 TF-IDF 模型（BoW）将所有文档表示成向量。需要提到的是，在使用 TF-IDF 模型表示预处理之后的文档时，训练集和测试集的单词总数及词典原本是不同的。为了方便后期 LR 模型训练和使用，且考虑到训练集的词典比测试集要大一些，所以在表示测试集文档时，强制使用和训练集一样的 TF-IDF 词典，以保证输入数据的维度一致。

对于使用基于词向量的深度学习模型（Word2Vec + TextCNN）的预处理过程，我使用 sklearn.datasets 中提供的工具方法对所有文档进行了和基准模型中相似的预处理流程。去除了标点符号并分词，去除了停词，但考虑到词向量模型的特点以及和 TF-IDF 模型的区别，此处没有进行单词词干化的预处理流程，因为词向量本身就可以表示类似单词之间的细微差别。

执行过程

对于改良后的基准模型(词干化后的 TF-IDF + LR)的训练，我依然选择了逻辑回归模型作为多分类器，使用准确率 Accuracy 作为评估指标。配合之前数据预处理部分所做的单词词干化，将训练集和测试集的文档转化为 118518 维（词干化后训练集的词典长度）的向量，作为逻辑回归模型的输入数据进行训练和测试。使用与原基准模型同样的超参组合的情况下(如下所示)，模型在测试集上的 Accuracy 由 0.72 提高到了 0.76，说明初步的调优还是有一定效果的，但与期望的 85%的准确率还有不小差距，需要进一步调优完善。

```
C=1, class_weight=None, dual=False, fit_intercept=True,intercept_scaling=1, max_iter=3, multi_class='multinomial',n_jobs=1,
penalty='l2', random_state=18, solver='sag',tol=0.0001, verbose=0, warm_start=False
```

可执行代码详见附件[B] Benchmark_Model_2_Stem.py

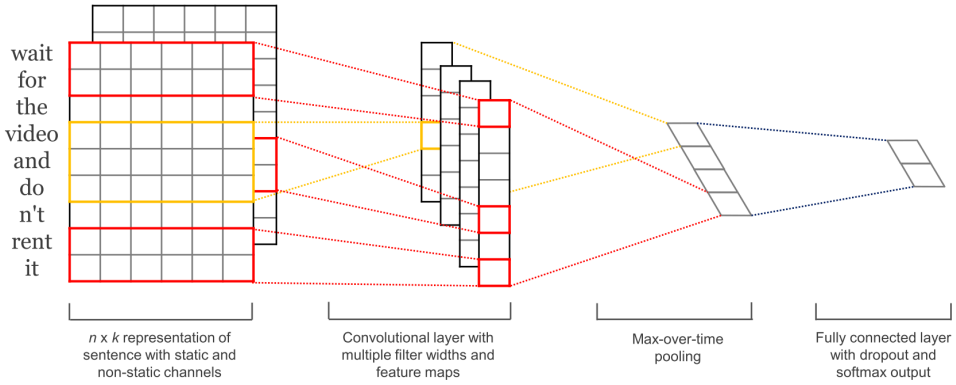
对于使用基于词向量的深度学习模型（Word2Vec + TextCNN）的训练执行过程，可以分为词向量训练部分和 TextCNN 训练部分。

在 Word2Vec 词向量的训练中，我选择使用训练集的文档按照 Skip-gram 模型来训练。将每个单词表示成为 300 维的词向量，其中单词窗口为 5(window=5)，且只统计训练集全部文档中出现 5 次以上的单词(min_count=5)。

使用词向量表示文档时，由于大部分文档的单词都在 200 以下，故将每个文档前 200 个单词(不足 200 时用 0.0 补齐，单词不在词向量列表时直接跳过该单词)堆叠成为 200 行 300 列的一个 200 * 300 的二维矩阵，即整个 TextCNN 的输入端的 shape 为(None, 200, 300)。

在 TextCNN 训练部分，我选择使用三个平行 Conv1D 层，filter_size 分别为[3, 4, 5]，每个 Conv1D 层的 filter 数量为 128 个，激活函数为 tanh。在每个 Conv1D 层之后放置一个 MaxPooling1D 层和一个 Flatten 层。然后通过 concatenate 方法将之前三个 Flatten 层的输出聚合成 1 列。然后加入一个 128 个神经元的隐藏层（使用 tanh 激活），最后的输出层是 20 个神经元(最终分类)，隐藏层和输出层之间为全连接层(Softmax)。

此结构与由 Yoon Kim 在 “Convolutional Neural Networks for Sentence Classification“一文中提出的算法结构(如下图所示)比较相似，稍有不同的是在输出层之前加了一层 128 个神经元的隐藏层。[9]



使用此 TextCNN 模型进行训练，mini_batch 设置为 50，validation_split 设置为 0.10，epochs 设置为 50，得到训练集的 Accuracy 在 90%以上，而测试集的 Accuracy 在 50～60%之间，效果十分不理想，存在很大的过拟合(overfitting)，并且词向量的准确性也存疑，需要进一步调优。

可执行代码详见附件[C] TextCNN_Model_1_Original.py

完善

对于改良后的基准模型(词干化后的 TF-IDF + LR)，继续使用 KFold GridSearchCV 来选择逻辑回归分类器的最优参数，并尝试使用其他 sklearn 提供的文本预处理方式处理数据，以提高模型整体表现。

选择 LogisticRegression 多分类器的以下参数进行尝试:

对于 GridSearch，尝试的 params 为

{'C': [0.1, 1, 10], 'multi_class':('ovr', 'multinomial'), 'max_iter':[1, 3, 7]}

对于 KFold，设置 n_splits 为 5，shuffle 为 True

在经过文档预处理和 K_Fold GridSearchCV 后，并使用 chi2 和 SelectKBest 进行特征选择后(先算出每个特征和标签的相关性，选择相关性最高的 6 万个特征，这样筛选后的特征就非常利于分类了)，模型准确率从 72%提高到了 86%，效果比较令人满意！最终模型参数

```
LogisticRegression(C=10, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, max_iter=7, multi_class='ovr', n_jobs=1,
                   penalty='l2', random_state=18, solver='sag', tol=0.0001,
                   verbose=0, warm_start=False)
```

可执行代码详见附件[D] Benchmark_Model_3_GridSearch_Best.py

在尝试逻辑回归模型的同时，还尝试了 MultinomialNB 和 SVM.SVC 两种机器学习模型。

MultinomialNB 的调参主要集中在 alpha 这个参数上，最终效果比较好的 alpha=0.01 可以是准确率达到 84%左右，略逊于逻辑回归。

而 SVC 的调参主要集中在 kernel(rbf/linear)和 C 的取值上，效果最好的一组参数'linear'和 C=1.0 可以使准确率达到 85，仍然略逊于逻辑回归模型。

对于使用基于词向量的深度学习模型（Word2Vec + TextCNN），主要考虑三点优化：

1. 使用训练集文档训练出来的 Word2Vec 模型不够准确，尝试使用 GoogleNews 预训练词向量[10]和 GloVe 预训练词向量[11]进行尝试(下面分别用训练集文档训练出来的词向量和 GloVe 预训练词向量来列出词意上与'college', 'school', 'university'最接近的 10 个单词)。

2. 对于之前模型出现的过拟合，尝试在模型中增加一些 Dropout 层来消除部分过拟合。

3. 在前两点优化的基础上对 TextCNN 模型进行微调，如尝试 filter_size 为(3, 4, 5, 6)，尝试调整 Conv1D 层的 filter 数量为 100 或 64 等取值，尝试调整隐藏层的神经元数量为 64 或 32 等值，尝试使用 relu 作为隐藏层和 Conv1D 层的激活函数等等。

使用训练集训练出来的 Word2Vec 词向量

```
model.most_similar('college')
```

```
[('claremont', 0.9366360902786255),
 ('youngstown', 0.9363923072814941),
 ('allegheny', 0.9166382551193237),
 ('ia', 0.916509747505188),
 ('univ', 0.9112268686294556),
 ('dayton', 0.9092824459075928),
 ('ohio', 0.9073096513748169),
 ('dept', 0.9036411046981812),
 ('nh', 0.9012987613677979),
 ('boulder', 0.8967545628547668)]
```

```
model.most_similar('school')
```

```
[('medicine', 0.8892966508865356),
 ('western', 0.8450724482536316),
 ('texas', 0.834946870803833),
 ('graduate', 0.8286827206611633),
 ('department', 0.823249101638794),
 ('reserve', 0.8055322170257568),
 ('engineering', 0.8053178787231445),
 ('pitzer', 0.8041761517524719),
 ('iowa', 0.7966450452804565),
 ('california', 0.7906061410903931)]
```

```
model.most_similar('university')
```

```
[('college', 0.8373095989227295),
 ('univelectronic', 0.7734301090240479),
 ('ohio', 0.7725263833999634),
 ('math', 0.7474826574325562),
 ('univ', 0.7408721446990967),
 ('claremont', 0.7388310432434082),
 ('dept', 0.7383532524108887),
 ('pitzer', 0.735273003578186),
 ('ia', 0.7349210977554321),
 ('walla', 0.7339256405830383)]
```

GloVe 预训练词向量

```
glove_model.most_similar('college')
```

```
[('university', 0.794602632522583),
 ('school', 0.779818058013916),
 ('student', 0.7411624193191528),
 ('colleges', 0.7254163026809692),
 ('graduate', 0.7153186202049255),
 ('College', 0.6926790475845337),
 ('undergraduate', 0.670303225517273),
 ('students', 0.6647510528564453),
 ('graduation', 0.6607996225357056),
 ('campus', 0.6606385111808777)]
```

```
glove_model.most_similar('school')
```

```
[('schools', 0.8140378594398499),
 ('college', 0.779818058013916),
 ('elementary', 0.7691876888275146),
 ('teacher', 0.7578344345092773),
 ('students', 0.7422181963920593),
 ('student', 0.7404062747955322),
 ('kindergarten', 0.7338491678237915),
 ('School', 0.7163903713226318),
 ('teachers', 0.7163441181182861),
 ('education', 0.7038593292236328)]
```

```
glove_model.most_similar('university')
```

```
[('college', 0.794602632522583),
 ('universities', 0.7863306403160095),
 ('colleges', 0.7419571876525879),
 ('campus', 0.7325456142425537),
 ('undergraduate', 0.7250710725784302),
 ('student', 0.7238409519195557),
 ('academic', 0.7130117416381836),
 ('graduate', 0.7076259851455688),
 ('faculty', 0.7044827938079834),
 ('institute', 0.6965475082397461)]
```

很明显可以看出，根据训练集得到的 Word2Vec 词向量并不可靠，GloVe 预训练词向量很好的表达了词语的现实含义。

最优的模型综合考虑了以上三点优化，最终在使用 GloVe 词向量以及加入了一些 Dropout 层之后，将测试集的整体 Accuracy 达到了 86%以上。任务完成！

可执行代码详见附件[E] TextCNN_Model_2_GoogleNews.py

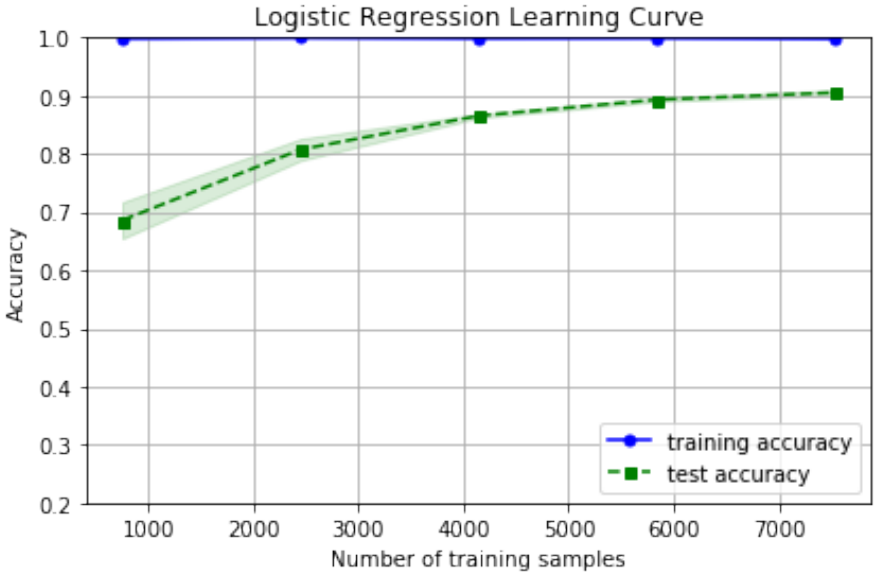
可执行代码详见附件[F] TextCNN_Model_3_GloVe.py

IV. 结果

模型的评价与验证

逻辑回归模型(LogisticRegression)

我画出了最终逻辑回归模型的学习曲线，可以从图中看到该模型在不同训练样本的情况下有着怎样的表现。



由图可知，样本点较少时(1000)，模型验证准确率已经达到了 0.70 左右，随着样本量的增加， 在 2500 样本时突破了 0.80，并在 7000 样本点时达到了 0.90。

综上，此逻辑回归多分类器模型的训练集准确率非常接近 100%，而且测试准确率也在 90%以上，过拟合非常小，因此我们可以断定，该多分类器对于不同规模的样本集来说是足够健壮可靠的。

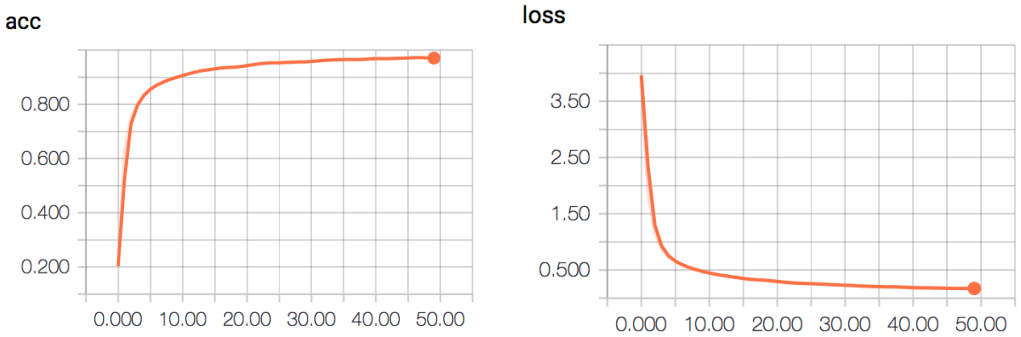
Word2Vec + TextCNN 模型

为了方便观察和比较模型训练过程中的 Accuracy 和 loss 的变化过程，我将最终模型(GloVe)相关数据放进了下面的表格：

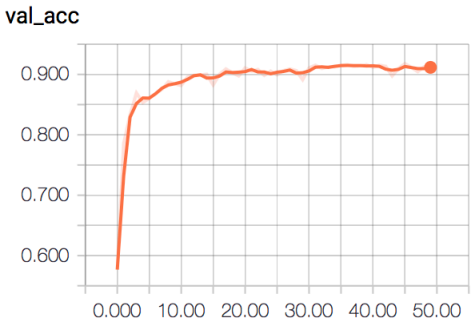
TextCNN 模型表现(With GloVe)

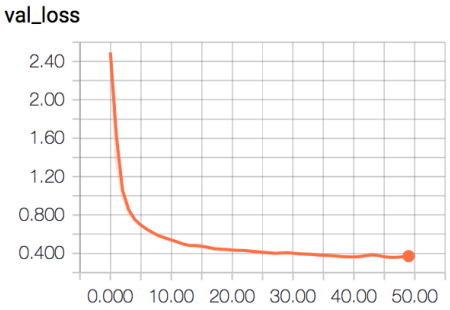
Epochs	0	10	20	30	40	50
Training Accuracy	0.2035	0.9078	0.9415	0.9573	0.9702	0.9703
Training Loss	3.964	0.4437	0.2977	0.2291	0.1830	0.1728
Validation Accuracy	0.5769	0.8816	0.9108	0.9108	0.9117	0.9117
Validation Loss	2.494	0.5490	0.4336	0.4030	0.3646	0.3714

模型表现在训练集上的变化过程(横坐标为训练的 epochs)：



模型表现在验证集的上变化过程(横坐标为训练的 epochs)：





可以看到，在模型训练到第 10 个 epoch 的时候(每个 epoch 的训练集数量为 10000 左右)，模型在验证集上的表现已经比较不错了，但仍然可以看到有一些过拟合(对比测试集的 84%的准确率)。出现过拟合的原因，可能是因为可作为训练集的文档还是相对较少，如果训练集再丰富一些，此模型应该会表现的更好。

合理性分析

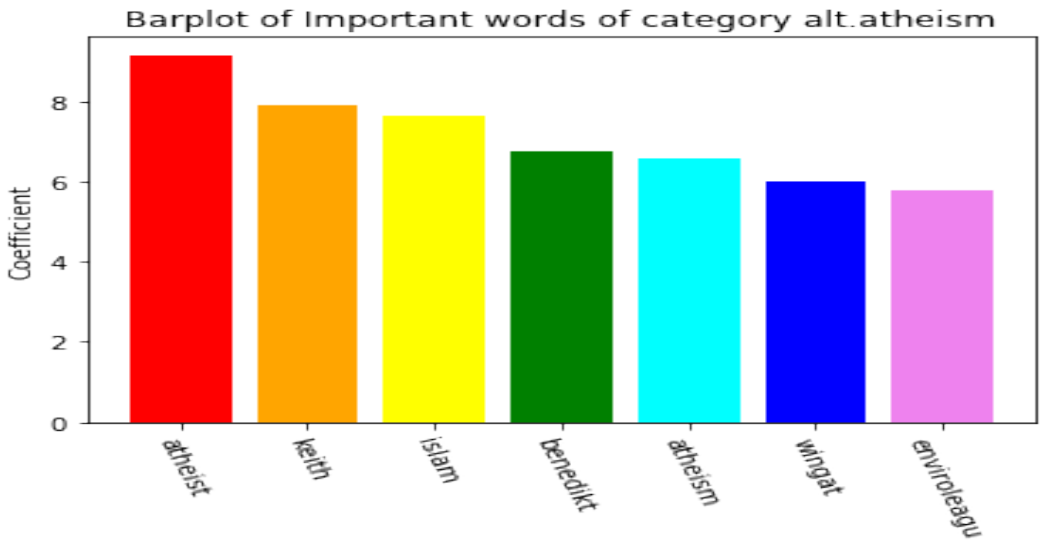
调优后的逻辑回归和 TextCNN 两个多分类器均比基准模型有着较大的提升，TextCNN 准确率达到了 84%，而逻辑回归甚至达到了 86%，分类效果上使用逻辑回归更佳。所以针对此数据集(20newsgroup)，使用逻辑回归做为最终的选择。

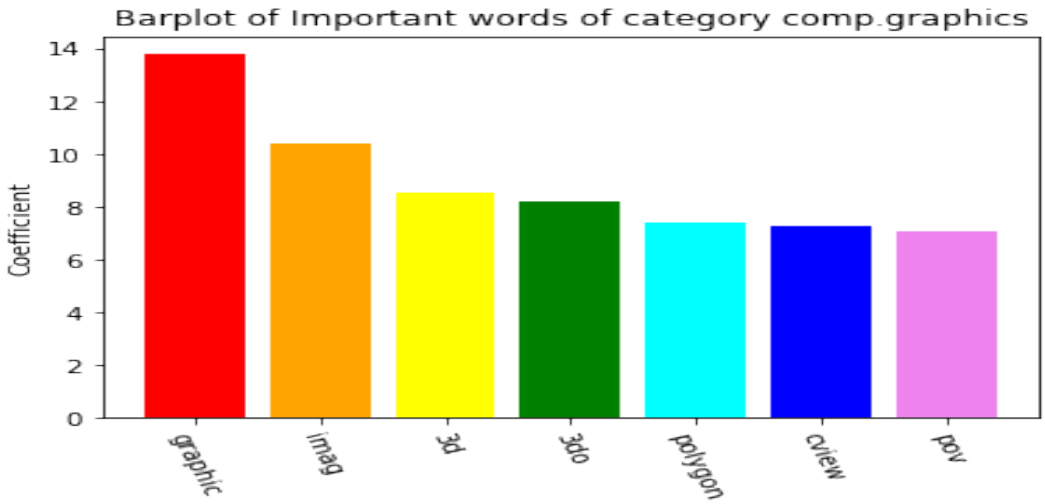
逻辑回归模型如果用于训练集减小的其他数据集时，效果会更好；TextCNN 如果用于训练集增加的其他数据集时，过拟合会进一步降低，效果更佳。

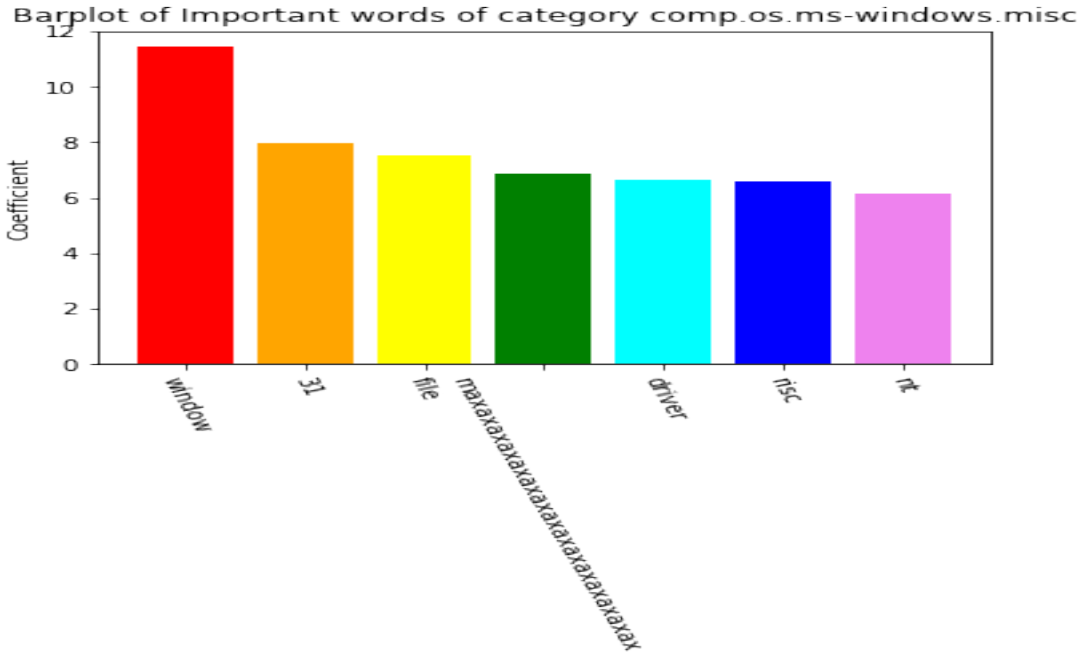
V. 项目结论

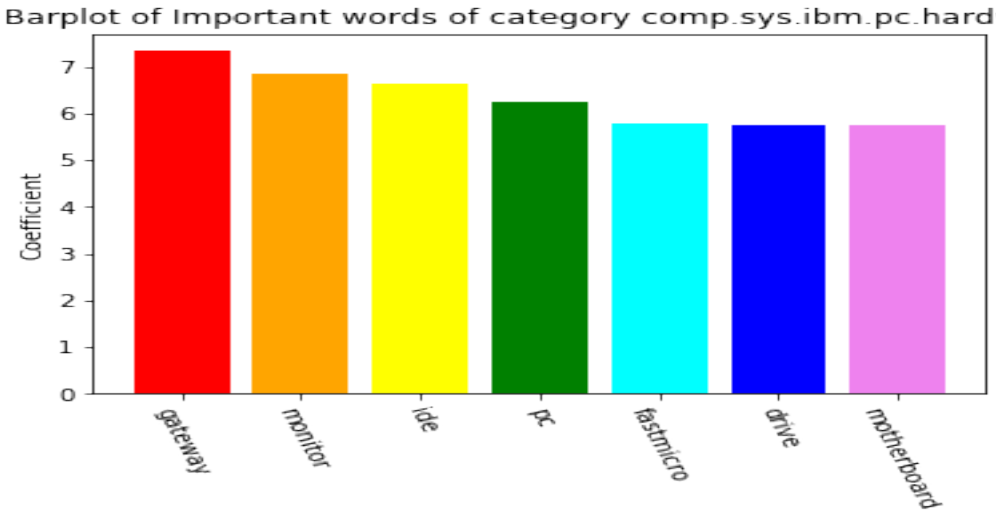
结果可视化

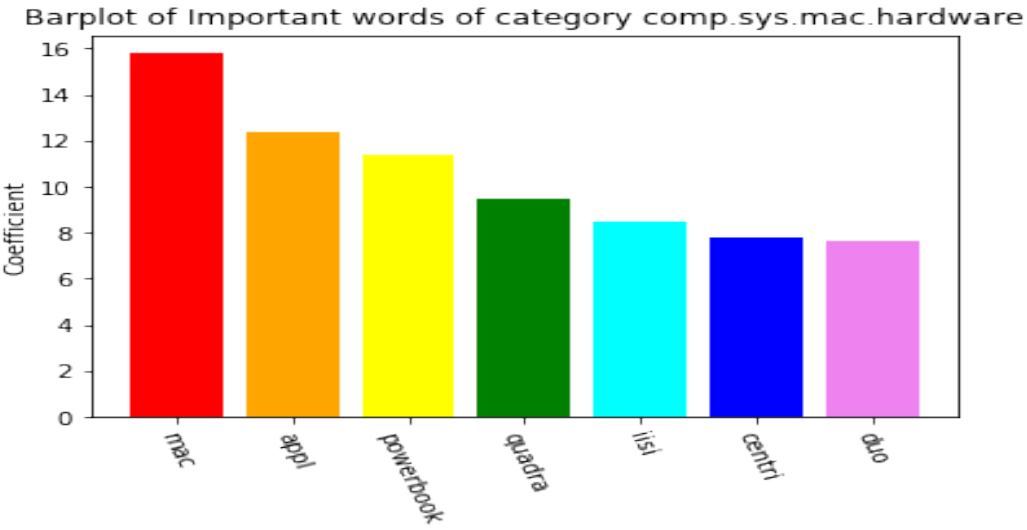
从最终的逻辑回归模型中，我们可以通过模型的 coef_属性提炼出对于每个类目/主题最重要的一些单词(有些是已经经过词干化处理的词干)。如下列图形所示：

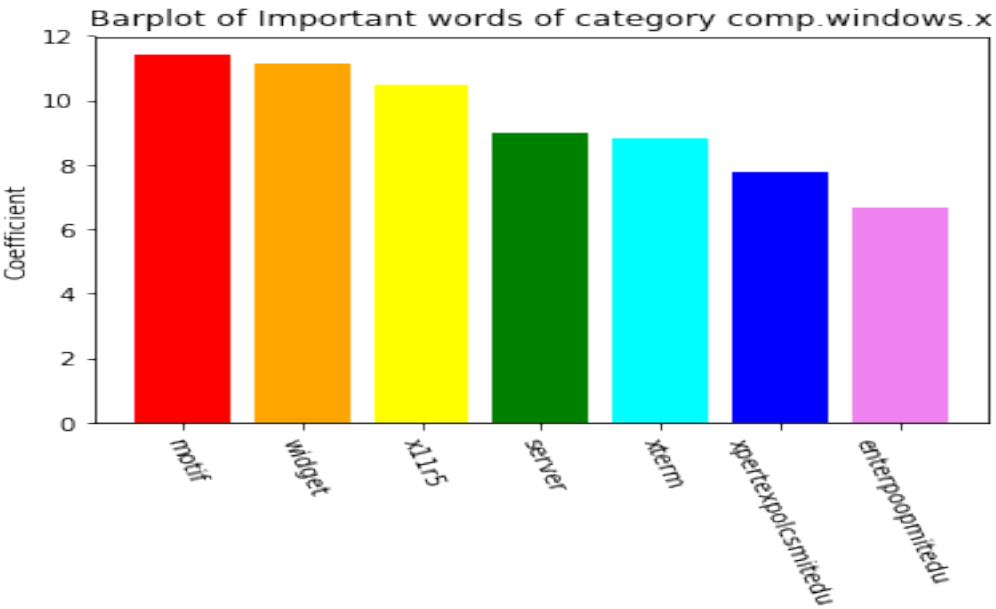

Barplot of Important words of category alt.atheism
Coefficient
0
2
4
6
8
atheist
keith
islam
benedikt
atheism
wingat
environleagu

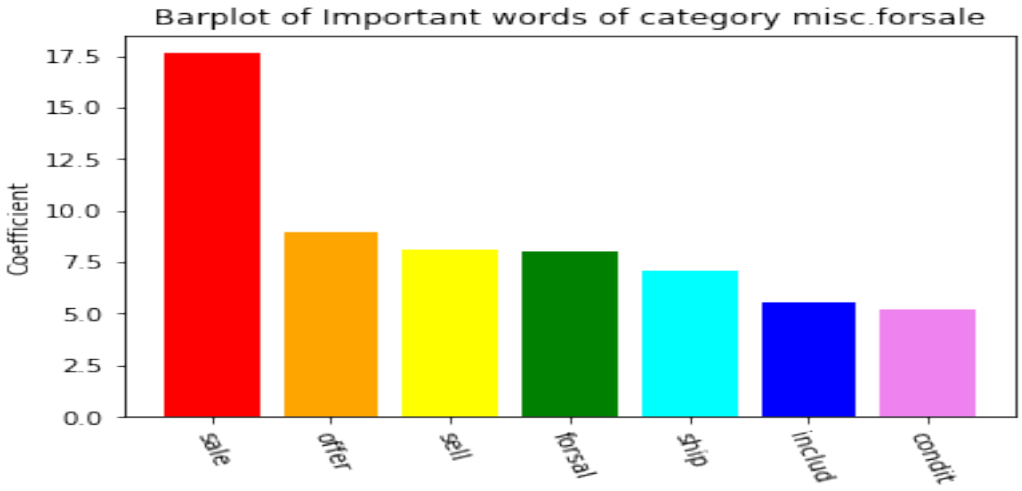

Barplot of Important words of category comp.graphics
Coefficient
0
2
4
6
8
10
12
14
graphic
imag
3d
3do
polygon
cview
pov

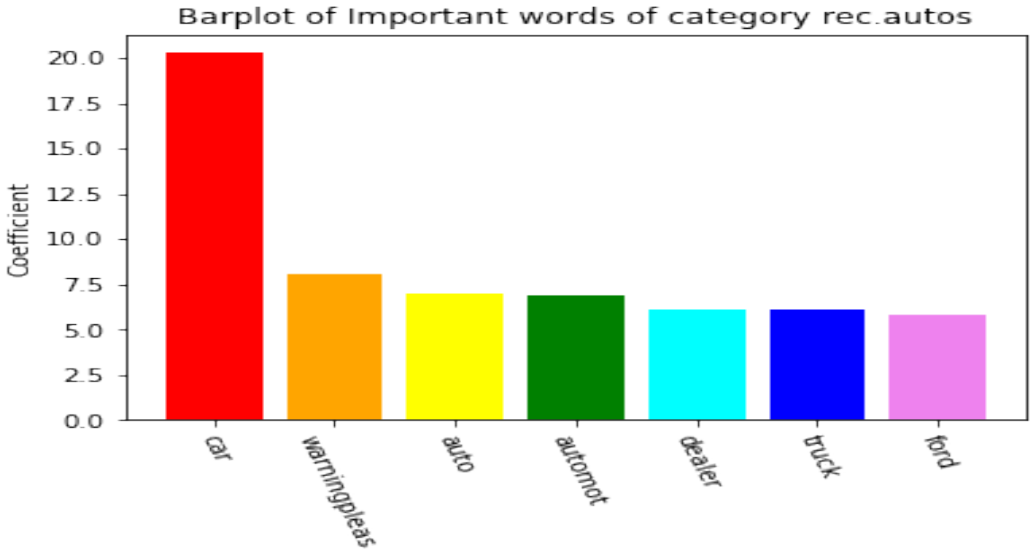

Barplot of Important words of category comp.os.ms-windows.misc
Coefficient
0
2
4
6
8
10
12
window
31
file
maxaxaxaxaxaxaxaxaxaxaxaxax
driver
risc
nt

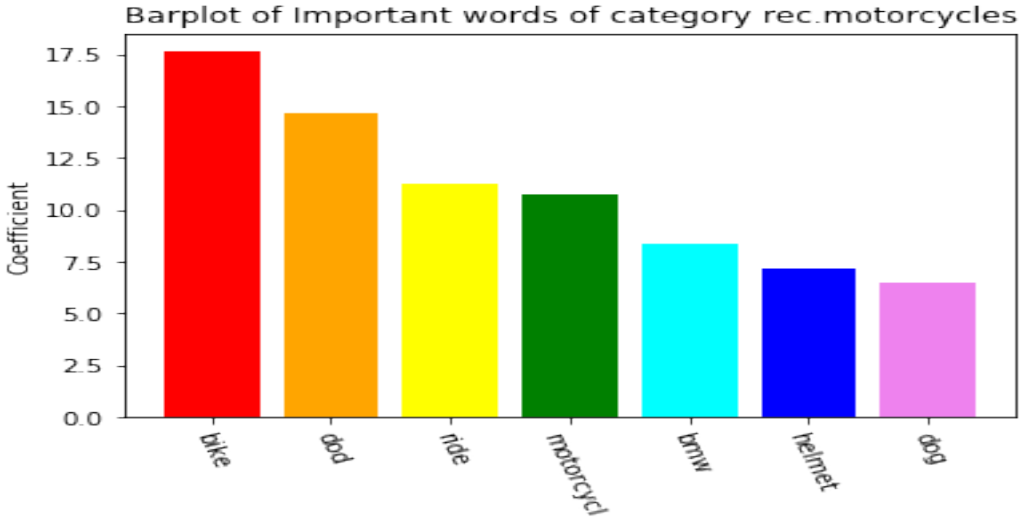

Barplot of Important words of category comp.sys.ibm.pc.hardware
Coefficient
0
1
2
3
4
5
6
7
gateway
monitor
ide
pc
fastmicro
drive
motherboard

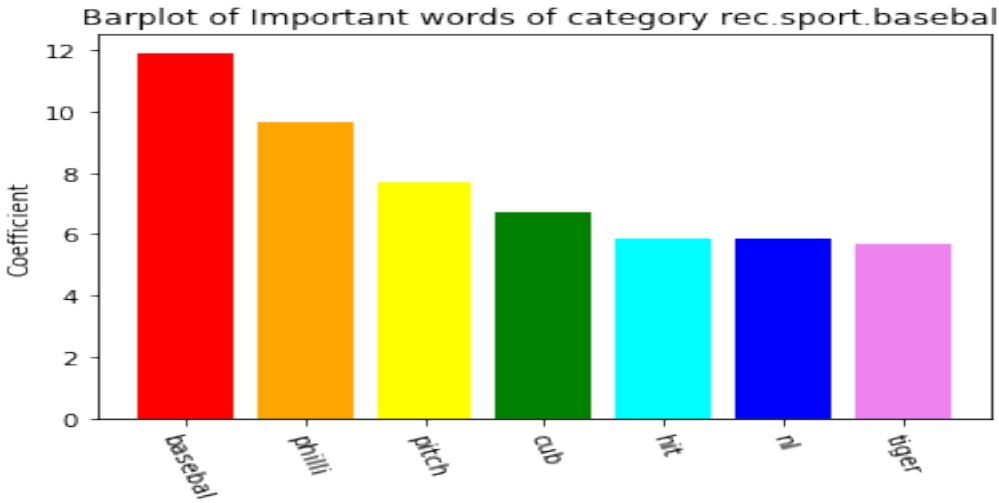

Barplot of Important words of category comp.sys.mac.hardware
Coefficient
0
2
4
6
8
10
12
14
16
mac
appl
powerbook
quadra
iisi
centri
duo

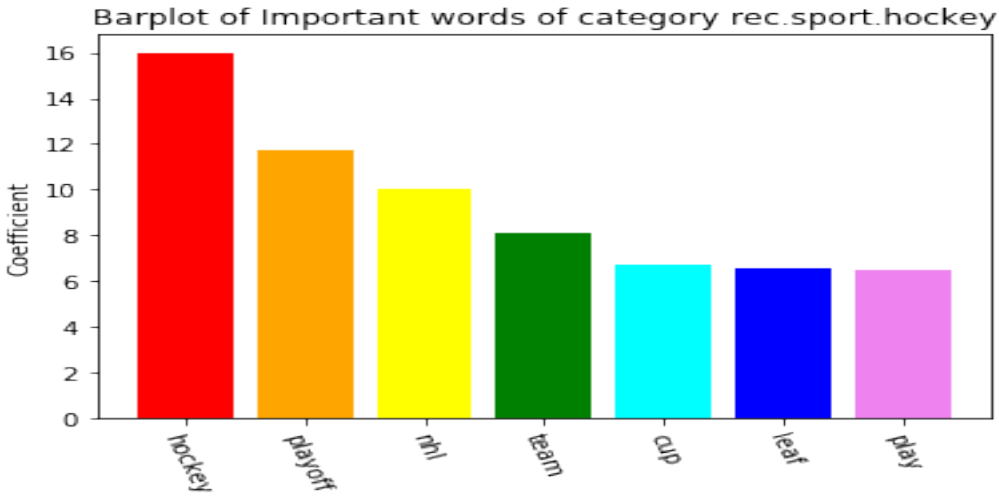

Barplot of Important words of category comp.windows.x
Coefficient
0
2
4
6
8
10
12
motif
widget
x11r5
server
xterm
xpertexpolcsmitedu
enterpoopmitedu

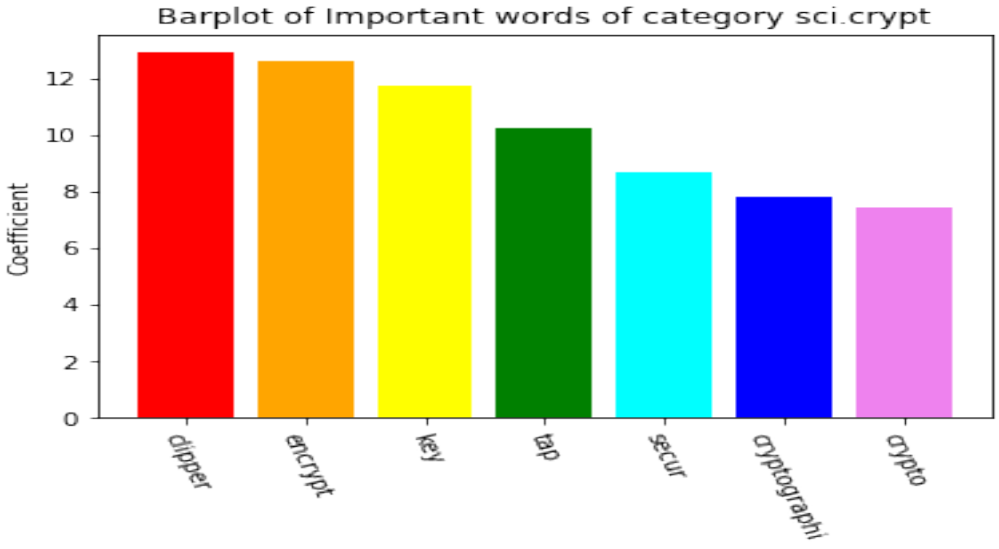

Barplot of Important words of category misc.forsale
Coefficient
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
sale
offer
sell
forsal
ship
includ
condit

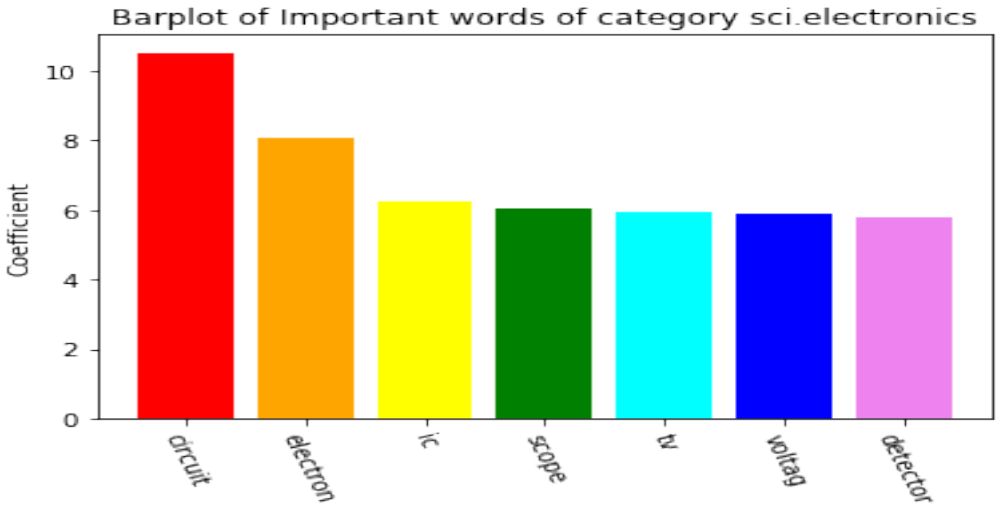

Barplot of Important words of category rec.autos
Coefficient
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
car
warningpleas
auto
automot
dealer
truck
ford

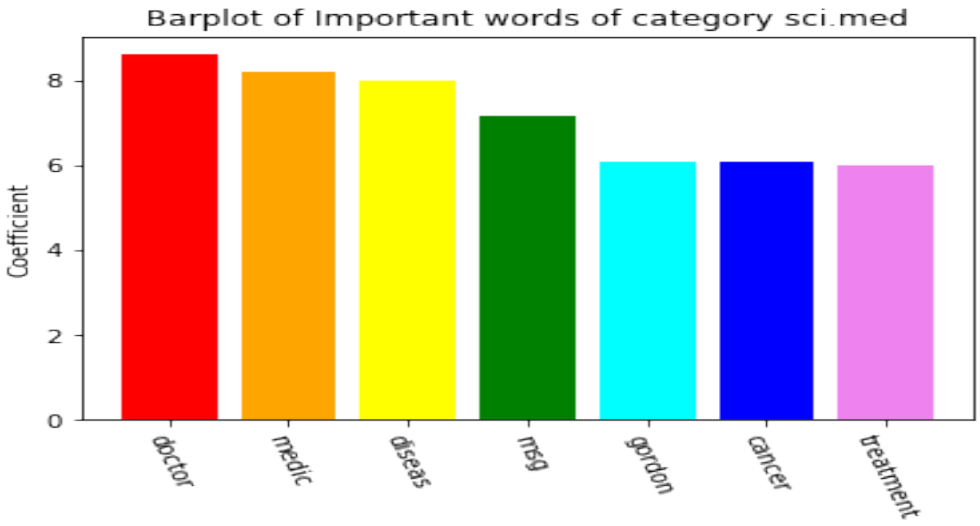

Barplot of Important words of category rec.motorcycles
Coefficient
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
bike
dod
ride
motorcycl
bmw
helmet
dog

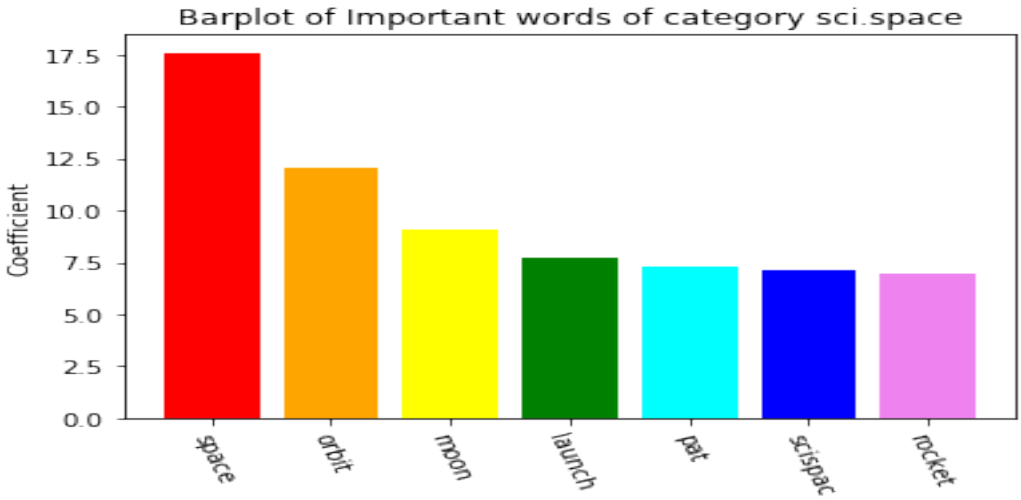

Barplot of Important words of category rec.sport.baseball
Coefficient
0
2
4
6
8
10
12
baseball
phillli
pitch
cub
hit
nl
tiger

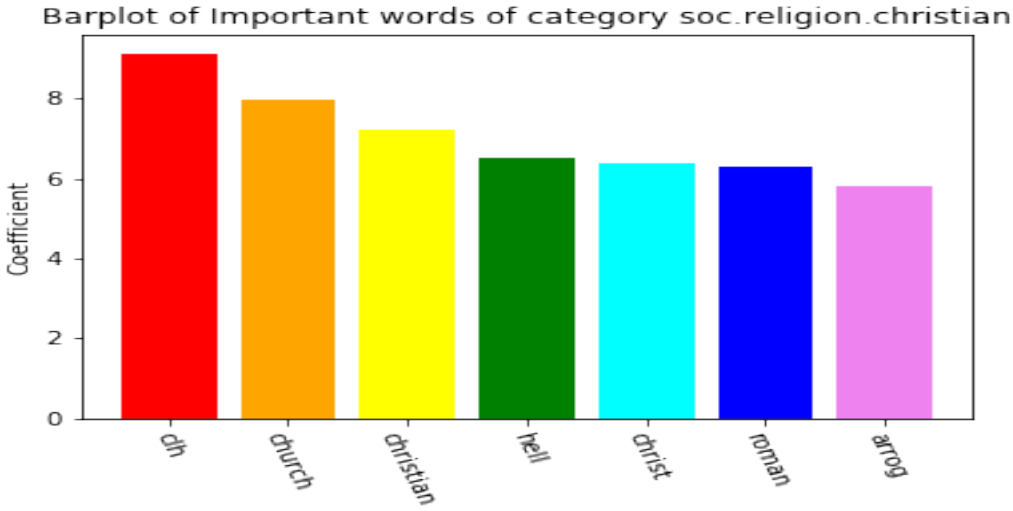

Barplot of Important words of category rec.sport.hockey
Coefficient
0
2
4
6
8
10
12
14
16
hockey
playoff
nhl
team
cup
leaf
play

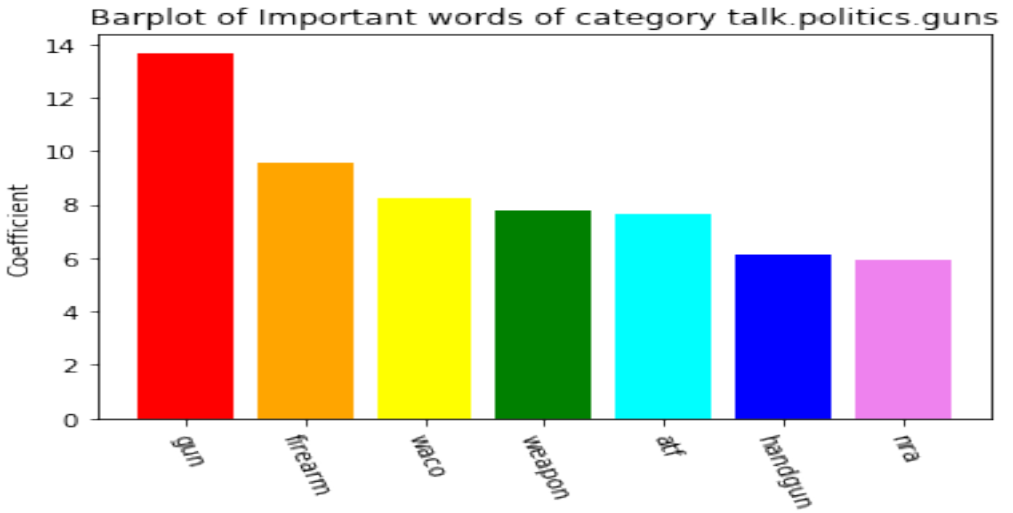

Barplot of Important words of category sci.crypt
Coefficient
0
2
4
6
8
10
12
clipper
encrypt
key
bap
secur
cryptographi
crypto

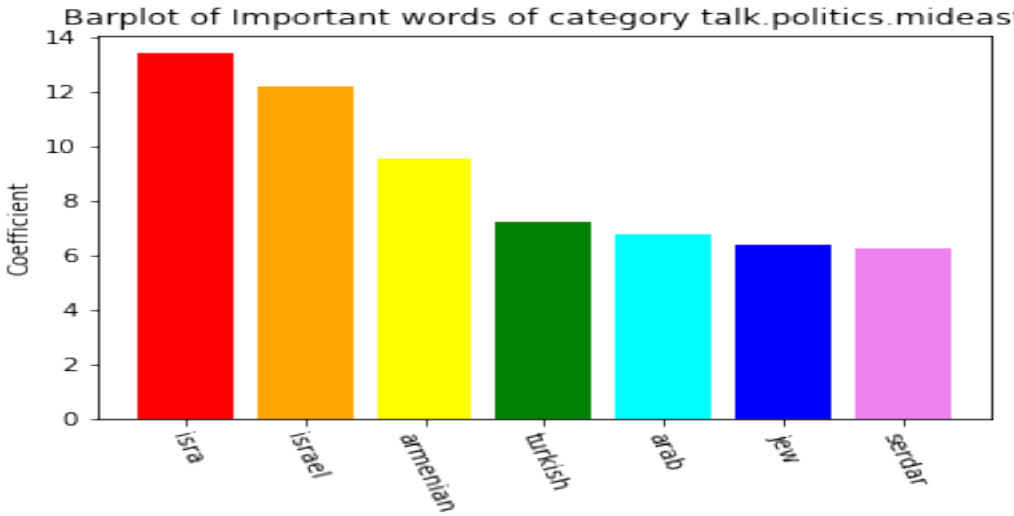

Barplot of Important words of category sci.electronics
Coefficient
0
2
4
6
8
10
circuit
electron
ic
scope
tv
voltag
detector

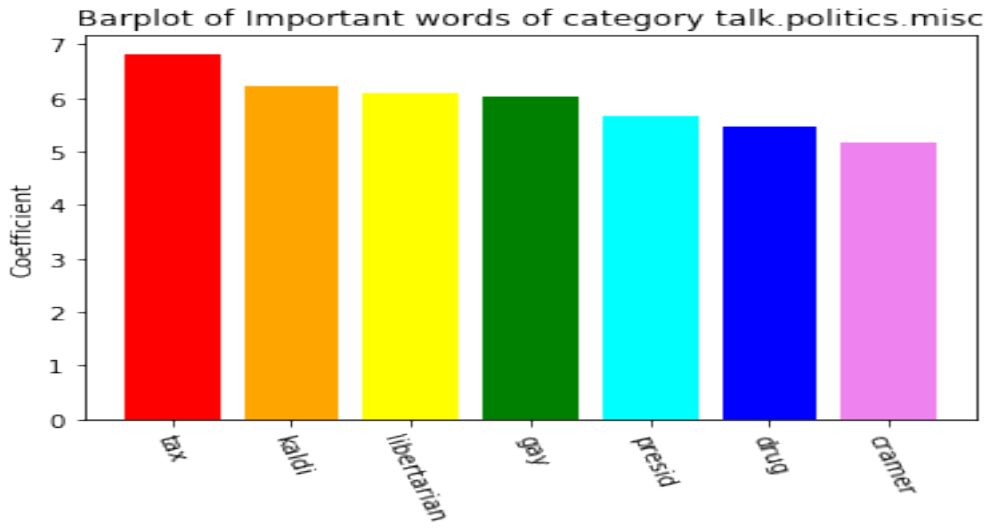

Barplot of Important words of category sci.med
Coefficient
0
2
4
6
8
doctor
medic
diseas
msg
gordon
cancer
treatment

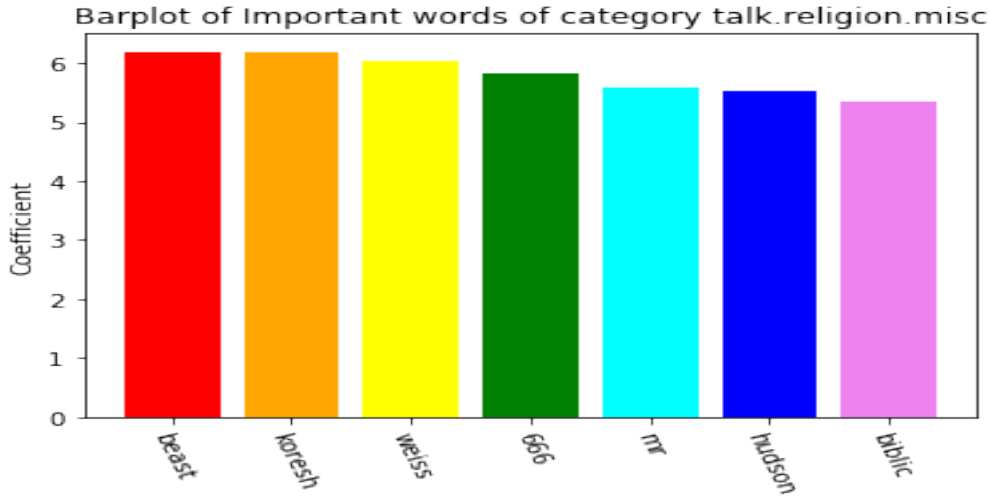

Barplot of Important words of category sci.space
Coefficient
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
space
orbit
moon
launch
pat
scispac
rocket


Barplot of Important words of category soc.religion.christian
Coefficient
0
2
4
6
8
dh
church
christian
hell
christ
roman
arrog


Barplot of Important words of category talk.politics.guns
Coefficient
0
2
4
6
8
10
12
14
gun
firearm
waco
weapon
atf
handgun
nra


Barplot of Important words of category talk.politics.mideast
Coefficient
0
2
4
6
8
10
12
14
isra
israel
armenian
turkish
arab
jew
serdar





从上列柱状图可以看出，每个类目/主题中最重要的单词确实和此类目/主题有着非常强的关联性。这也证明了最终的逻辑回归模型是可信的。

对项目的思考

本项目解决的是一个有着超多特征维度文档类目/主题的多分类问题。我下载了 20newsgroup 数据集，并选择了全部类目进行多分类任务。

我以 TF-IDF 模型和逻辑回归多分类模型作为入手点，研究了 TF-IDF 词向量的产生原理，并将所有文档在预处理之后使用 TF-IDF 表示出来。此时的文档向量有 11 万以上的超多维度。之后，我使用了逻辑回归模型进行了多分类的尝试。起初的准确率只有 76%左右。后

来我应用了 KFold GridSearchCV 进行了超参选择，并成功的使用效果最好的一组超参，将模型准确率提升到了 86%左右。对于所有类目的多分类任务来说，这个结果已经非常不错了，而且也超过了之前给自己设定的 85%的目标。

随后我使用 Word2Vec 和 TextCNN 的组合，再次挑战全类别的分类任务。尝试期间，经历了很多的潜在问题，比如使用训练集来训练 Word2Vec 词向量会导致词向量很不可靠，比如基于 Word2Vec 词向量的模型在预处理阶段不适合进行单词词干化，比如 TextCNN 模型的搭建和参数调优，又比如重新尝试 GoogleNews 预训练词向量和 GloVe 词向量来提高模型准确率……经历的挫折和挑战真的是数不胜数，也让我知道了一个机器学习工程师的艰辛。还好经过了这些挑战，最终我将全类目多分类的准确率提高到了 86%，完成了任务！

需要作出的改进

「自然语言处理-文档分类」这个项目中，我还没有尝试过另外一些主流的模型，如 Doc2Vec 模型和 LSTM 模型，我将会在此报告之外继续尝试，力求对此类问题有一个全面的了解。

引用资料

[1] 文档分类 https://zh.wikipedia.org/wiki/文档分类

[2] 自然语言处理 https://zh.wikipedia.org/wiki/自然语言处理

[3] 本项目的描述文档
https://github.com/nd009/capstone/tree/master/document_classification

[4] TF-IDF 及其算法 https://blog.csdn.net/sangyongjia/article/details/52440063

[5] 词向量及语言模型 https://blog.csdn.net/EwellWang/article/details/80755797

[6] 理解 GloVe 模型 https://blog.csdn.net/coderTC/article/details/73864097

[7] 逻辑回归模型基础 http://www.cnblogs.com/sparkwen/p/3441197.html

[8] Convolutional Neural Networks for Sentence Classification – Yoon Kim
https://arxiv.org/pdf/1408.5882.pdf

[9] 同引用[8]

[10] Word2Vec https://code.google.com/archive/p/word2vec/

[11] GloVe: Global Vectors for Word Representation
https://nlp.stanford.edu/projects/glove/