贪心机器学习项目实验手册

豆瓣电影评分预测

项目设计和编写: 刘畅

后期编辑: 李文哲

单 位: 贪心科技

2021 年 2 月 25 日

目 录

1 项目描述与目标

豆瓣电影是一个中文网站，允许互联网用户分享他们对电影的评论和观点。用户可以在电影上发表简短或长期的评论并给他们打分。本项目基于豆瓣电影平台上用户对电影的评论与打分的数据集，构建一个豆瓣电影评分预测系统。

通过本项目的练习，你能通晓文本处理的各个流程：

- **中文分词**：这是处理中文文本数据的第一步
- **文本预处理**：这是所有 NLP 项目的前提，或多或少都会用到相关的技术，主要包括去停用词、去低频词等技术。
- **文本特征提取**：任何建模环节都需要特征提取的过程，你将会学到如何使用 `tfidf`、word2vec、`bert embedding` 等技术来设计文本特征。
- **模型搭建和训练**：在这里你将会使用前面学到一些机器学习分类模型来搭建算法。

同时，通过本项目

- 1. 熟练掌握分词, 过滤停止词等技术
- 2. 熟练掌握 TF-IDF 模型
- 3. 熟练掌握 word2vec 模型
- 4. 熟练掌握使用 BERT Embedding
- 5. 熟练掌握构造句向量的技术
- 6. 熟练掌握逻辑回归、朴素贝叶斯模型的搭建

2 项目数据描述

项目的数据来源：https://www.kaggle.com/utmhikari/doubanmovieshortcomments。项目的数据是基于用户在豆瓣电影平台上的评论和打分而建立的。

项目的原始数据示例如图 1所示。

	ID	Movie_Name_EN	Movie_Name_CN	Crawl_Date	Number	Username	Date	Star	Comment	Like
0	0	Avengers Age of Ultron	复仇者联盟2	2017-01-22	1	然潘	2015-05-13	3	连奥创都知道整容要去韩国。	2404
1	10	Avengers Age of Ultron	复仇者联盟2	2017-01-22	11	影志	2015-04-30	4	“一个没有黑暗面的人不值得信任。”第二部剥去冗长的铺垫，开场即高潮、一直到结束，会有人觉...	381
2	20	Avengers Age of Ultron	复仇者联盟2	2017-01-22	21	随时流感	2015-04-28	2	奥创弱爆了弱爆了弱爆了啊！！！！！！	120
3	30	Avengers Age of Ultron	复仇者联盟2	2017-01-22	31	乌鸦火堂	2015-05-08	4	与第一集不同，承上启下，阴郁严肃，但也不会不好看啊，除非本来就不喜欢漫威电影。场面更加宏大...	30
4	40	Avengers Age of Ultron	复仇者联盟2	2017-01-22	41	办公室甜心	2015-05-10	5	看毕，我激动地对友人说，等等奥创要来毁灭台北怎么办厚，她拍了拍我肩膀，没事，反正你买了两份...	16
...	...	...	...	...	...	...	...	...	...	...
212501	2125010	Zootopia	疯狂动物城	2017-01-04	141151	Lu	2016-03-05	5	里里外外我都打满分，太赞了。	0
212502	2125020	Zootopia	疯狂动物城	2017-01-04	141161	titaa	2016-03-06	5	超棒！！！ 拟人超级像的，每个配角都好有戏，每个动物都好萌，想摸摸毛，fox一直叫bunny...	0
212503	2125030	Zootopia	疯狂动物城	2017-01-04	141173	宛如智障鱼摆摆	2016-03-06	4	狐狸确实帅	0
212504	2125040	Zootopia	疯狂动物城	2017-01-04	141184	Ligger	2016-03-07	5	不负我望，超级好看！	0
212505	2125050	Zootopia	疯狂动物城	2017-01-04	141194	Usthiel	2016-03-06	4	毛——茸茸——的——胜利——	0

212506 rows × 10 columns

图 1: 原始数据示例

本项目使用的数据集中，一共包括 212506 个样本，我们会将其中的 70% 作为训练样本，30% 作为测试样本。数据集共有 10 列，但我们本次项目只会用到其中的 Comment 列和 Star 列。Comment 列保存了用户的一些评论，Star 列保存了用户的打分，这是我们需要预测的列。

3 项目事宜

本项目是一个文本分类任务，按照中文分词、文本预处理、文本特征提取、模型搭建与训练 4 部分组成。通过本项目的实操，你将会体会到每个环节的细节如何去落地。

3.1 项目的整个框架

整个项目框架如图 2所示。下面对于图中每个模块做简要的描述，具体的细节请参考本文章后续的内容。

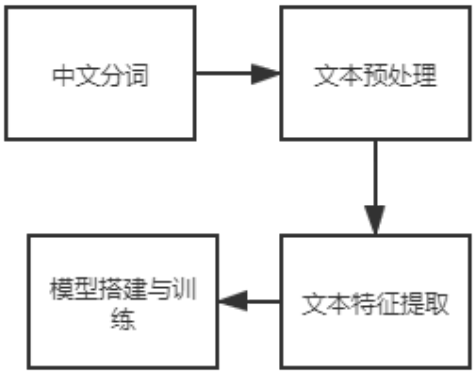


图 2: 豆瓣电影评分预测系统项目框架图

- **中文分词**：我们通常使用 python 中的 jieba 库来进行中文分词。
- **文本预处理**：文本预处理阶段通常包括对文本的清洗，去停用词以及去低频词等步骤。
- **文本特征提取**：文本特征提取有多种方式，其中 TF-IDF、Word2vec 和 BERT Embedding 都是常用的文本特征提取的方式。
- **模型搭建与训练**：本项目主要使用逻辑回归和朴素贝叶斯这两种机器学习分类模型。这也是大家之前学过的两个模型。

4 项目主要技术介绍

4.1 中文分词

中文分词指的是将一个汉字序列切分成一个一个单独的词。分词就是将连续的字序列按照一定的规范重新组合成词序列的过程。

现有的分词算法可分为三大类：基于字符串匹配的分词方法、基于理解的分词方法和基于统计的分词方法。

基于字符串匹配的分词方法又叫做机械分词方法，它是按照一定的策略将待分析的汉字串与一个“充分大的”机器词典中的词条进行配，若在词典中

找到某个字符串，则匹配成功。

基于理解的分词方法是通过让计算机模拟人对句子的理解，达到识别词的效果。其基本思想就是在分词的同时进行句法、语义分析，利用句法信息和语义信息来处理歧义现象。它通常包括三个部分：分词子系统、句法语义子系统、总控部分。在总控部分的协调下，分词子系统可以获得有关词、句子等的句法和语义信息来对分词歧义进行判断，即它模拟了人对句子的理解过程。这种分词方法需要使用大量的语言知识和信息。由于汉语语言知识的笼统、复杂性，难以将各种语言信息组织成机器可直接读取的形式，因此目前基于理解的分词系统还处在试验阶段。

基于统计的分词方法给出大量已经分词的文本，利用统计机器学习模型学习词语切分的规律（称为训练），从而实现对未知文本的切分。例如最大概率分词方法和最大熵分词方法等。随着大规模语料库的建立，统计机器学习方法的研究和发展，基于统计的中文分词方法渐渐成为了主流方法。

使用 python 的 jieba 库可以很方便得进行中文分词。

4.2 TF-IDF

TF-IDF 是一种常用的文本特征提取的方法，TF-IDF 有两层意思，一层是”词频”（Term Frequency，缩写为 TF），另一层是”逆文档频率”（Inverse Document Frequency，缩写为 IDF）。TF-IDF 的步骤为：

第一步，计算词频：词频 (TF) = 某个词在文章中的出现次数。考虑到文章有长短之分，为了便于不同文章的比较，进行”词频”标准化。词频 (TF) = $\frac{\text{某个词在文章中的出现次数}}{\text{文章的总词数}}$。

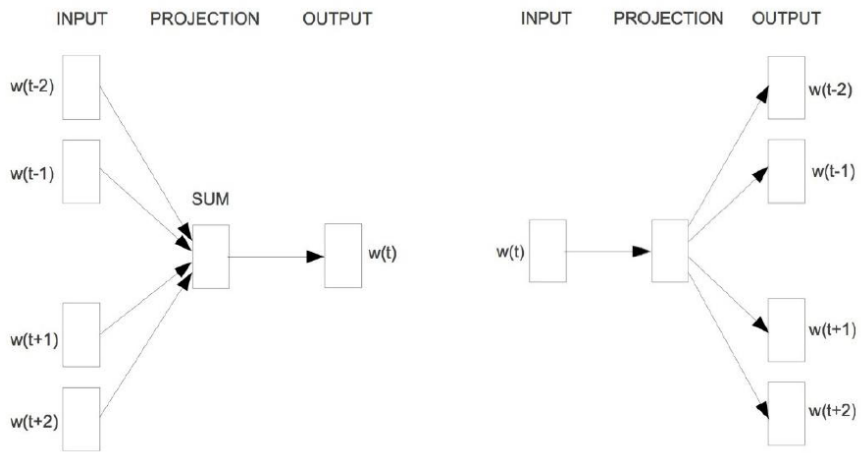
第二步，计算逆文档频率： 逆文档频率(IDF) = $\log\left(\frac{\text{语料库的文档总数}}{\text{包含该词的文档数+1}}\right)$。如果一个词越常见，那么分母就越大，逆文档频率就越小越接近 0。分母之所以要加 1，是为了避免分母为 0（即所有文档都不包含该词）。log 表示对得

到的值取对数。

第三步，计算 TF-IDF：TF − IDF = 词频 (TF) × 逆文档频率 (IDF) 。可以看到，TF-IDF 与一个词在文档中的出现次数成正比，与该词在整个语言中的出现次数成反比。

4.3 Word2vec

word2vec 实际上是两种不同思想实现的：CBOW（Continuous Bag of Words）和 Skip-gram。图 3中展示了 word2vec 的两种模型框架。CBOW 模型框架如图 3(a) 所示，它的目标是根据上下文来预测当前词语的概率，且上下文所有的词对当前词出现概率的影响的权重是一样的，因此叫 continuous bag-of-words 模型。Skip-gram 刚好相反，Skip-gram 是根据当前词语来预测上下文的概率，模型框架如图 3(b) 所示。这两种方法都利用人工神经网络作为它们的分类算法。起初每个单词都是一个随机 N 维向量。经过训练之后，该算法利用 CBOW 或者 Skip-gram 的方法获得了每个单词的最优向量。



(a) CBOW 模型框架　　(b) Skip-gram 模型框架

图 3: word2vec 模型框架

对于 `Word2Vec`, 我们可以通过调用 `gensim.models.Word2Vec()` 来训练并

获取对应的词向量。

```
self.w2v = models.Word2Vec(min_count=2,
                           window=3,
                           size=300,
                           sample=6e-5,
                           alpha=0.03,
                           min_alpha=0.0007,
                           negative=15,
                           workers=4,
                           iter=10,
                           max_vocab_size=50000)
```

但由于训练出一个高效的 word2vec 词向量往往需要非常大的语料库与计算资源，所以我们通常不自己训练 Wordvec 词向量，而直接使用网上开源的已训练好的词向量。https://github.com/Embedding/Chinese-Word-Vectors 中有已训练好的中文词向量文件。

4.4 BERT Embedding

我们可以通过 bert_embedding.BertEmbedding() 来很方便得使用 bert 进行文本编码。

在安装 bert_embedding 库时，只需要使用"pip install bert_embedding" 便可将此 python 包安装成功。但若要使用其 gpu 版本，我们还需要安装其 mxnet 包。安装 mxnet 包时要注意我们应安装与我们本地的 cuda 版本相同的 mxnet 包。首先使用命令"nvcc -V" 查看本地的 cuda 版本，之后按照相应 cuda 版本的 mxnet 包，如 cuda 版本为 10.0，则应使用命令"pip install mxnet-cu100" 来按照 mxnet 包。

使用 bert_embedding.BertEmbedding() 来获取 bert embedding 的实例如下：

首先导入 gpu 版本的 bert embedding 预训练模型：

```
from bert_embedding import BertEmbedding
```

```
import mxnet
ctx = mxnet.gpu()embedding = BertEmbedding(ctx=ctx)
```

之后使用 bert embedding 来得到文本的编码，embedding() 的输入为一个列表：[第一句话，第二句话，..., 第 n 句话]。embedding() 的返回为一个列表：[(第一句话的 tokens, 第一句话的 tokens embedding)，(第二句话的 tokens, 第二句话的 tokens embedding)，...，(第 n 句话的 tokens, 第 n 句话的 tokens embedding)]。如：

```
result = embedding(['今天月亮真圆啊','你觉得这部电影怎么样'])
```

4.5 句子向量

为了对评论的打分进行预测，我们需要将一个句子表示为一个向量，这种方式叫做 Sentence Embedding 或者 Doc Embedding。其中最常用的一种模型是 Average word vectors 或者叫 Word Averaging Model(WAM)。其方法是得到将一个句子中每一个词向量在同一个 embedding 维度的值取平均值得到该句子在该嵌入维度的值。如图 4所示。

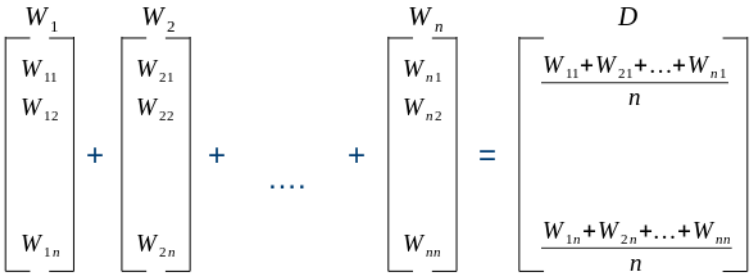


图 4: Average word vectors

按照这种方式，我们可以将 Word2vec 后的词向量取平均来表示句向量。也可以将 bert embedding 后的 tokens embedding 取平均来表示句向量。