基于朴素贝叶斯的商品尺寸匹配程度分类器

DZ1933026 王国畅[1)]

[1)](南京大学计算机系)

摘　要　本次实验目标为构建一个分类器, 基于用户的商品评论及其他相关信息, 预测商品尺寸与用户的匹配程度。这份报告中描述一种基于朴素贝叶斯和文本词频特征的分类器构造方法, 包括问题分析、算法的动机和描述、技术细节以及对性能的分析。

关键词　朴素贝叶斯; 文本分类

A Naive Bayes Based Method for Predicting How Product Size Fits The User

Guochang Wang[1)]

[1)](Department of Computer Science and Technology, Nanjing University)

Abstract　The target of this experiment is to build a classifier, based on the user's product comments and other related information, to predict how this product size fits the user. This report describes my Naive Bayes and word frequency based approach for constructing a classifier, with problem analysis, motivation and description of algorithm, technical details and my analysis of performance.

Key words　Naive Bayes; Text Classification

1　问题分析

1.1　问题描述

本次实验目标为训练一个商品尺寸匹配程度分类器, 其输入为用户的评论记录, 输出为匹配程度。训练集有已标注数据 135000 条。

用户评论记录　包含年龄 (age), 身材 (body type), 胸围 (bust size), 商品类型 (category), 匹配程度 (fit), 身高 (height), 商品 ID(item_id), 打分 (rating), 租赁目的 (rented for), 评论日期 (review_date), 评论总结 (review_summary), 评论文本 (review_text), 尺寸 (size), 用户 ID(user_id), 体重 (weight)。

尺寸匹配程度　为枚举值, 选值范围为 fit(尺寸合适),small(尺寸偏小), 和 large(尺寸偏大)。

1.2　输入特征选择

在输入的特征中, 每个特征的选择对应一个假设。例如:

- 身高: 假设身高不同的用户能买到尺寸合适的商品概率是不同的
- ……
- 评论文本: 假设购买到尺寸合适/偏大/偏小商品的用户会发表不同类型的评论
- 打分: 假设购买到尺寸合适/偏大/偏小商品的用户在打分上会有明显区别

笔者认为其中较为合理的假设是评论文本, 评论总结和打分, 其中经过抽样验证评论总结包含于评论文本, 故以下验证评论文本和打分与尺寸匹配程度强相关的假设。

评论文本　可以通过训练集在不同标签下的词频分布对其类别差异进行简单展示, 如图1, 图2和图3。可以清晰的看到除了未经过滤的 dress 等单词外, 不同类型的词频图中的高频词汇频率分布有所不同, 例如 small 类中有高频的 small,tight 等单词, 而 large 类中有高频的 large,long 等单词。也有部分单词在多个分类下都有较高词频, 如 fit 在 fit 类和 small 类中都经常出现,great 在 fit 和 large 类中经常出现。这表明评论文本的词频信息的确具有较为鲜明的类别特征, 适用于尺寸匹配程度的分类。

打分　取值为 [0, 10] 的枚举值, 可以通过频率直方图对打分分布进行分类别统计, 如图4, 图5和图6。从图中不难发现, 无论尺寸如何, 用户总是倾向于给

图 1 fit 类词频图

图 2 small 类词频图

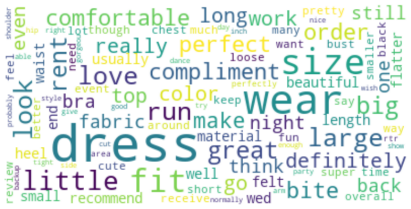

图 3 large 类词频图

[8-10] 的高分, 而极少给 5 分以下的低分, 这使得不同匹配程度的商品只在 8,9,10 的高分段分数频率有略微的区别, 使得打分这一特征通过频率区分类别, 即当一个用户给一个商品评分 9 分, 在训练集的数据分布下, 尺寸合适/偏大/偏小的概率极为接近。通过箱线图7可以更直观地看出三个类别打分分布十分接近, 因而打分不具有鲜明的类别特征, 不适用于尺寸匹配程度的分类。

1.3 问题总结

综上所述, 笔者将本实验任务定义为对商品评论记录的评论文本进行文本分类的多分类任务。任务的目标分类器输入为评论文本, 输出为尺寸匹配程度。

2 算法设计

2.1 算法选择

综合上述问题分析可以看出, 词频信息虽然含有较多噪音, 但具有区分类别的能力, 故本次实验采用基于词频信息的朴素贝叶斯分类器。具体的选择动机, 算法背景和伪代码算法描述于本章的后续小节。

2.2 选择动机

本次实验任务有较大量的数据, 经过统计约 70% 的评论文本类别标签为合适, 这与我们的生活经验是较为符合的, 由此笔者提出以下假设:

基本假设 1 训练集中的数据与预测任务数据类别分布相同。

同时, 不同单词间的 context 信息与最终预测结果的相关性是较为高维的特征, 需要复杂的方式来进行学习。笔者这里简单地假设不同单词条件独立:

基本假设 2 训练集中的单词词频特征满足条件独立性。

基于**基本假设 1**, 训练集中的各个类别出现的先验概率即其在训练集中出现的概率, 从而适合于贝叶斯分类器的应用。基于**基本假设 2**, 贝叶斯分类器判别式中的联合概率可以被重写为条件概率迭乘, 从而形成朴素贝叶斯分类器。

因此, 在上述基本假设下, 本任务适用朴素贝叶斯方法。

同时, 相较于 Bert、LSTM 和 RCNN 等基于神经网络的学习方法, 朴素贝叶斯具有良好的可解释性, 能够复现, 同时训练时间开销更小, 硬件需求更低。

综上, 笔者选择朴素贝叶斯作为分类算法。

2.3 算法背景

朴素贝叶斯分类器[1] 是带有待分类特征相互条件独立约束的贝叶斯分类器。

贝叶斯分类器的基本原理为贝叶斯决策。贝叶斯决策就是在不完全情报下, 对部分未知的状态用主观概率估计, 然后用贝叶斯公式对发生概率进行修正, 最后再利用期望值和修正概率做出最优决策。令 $D_1, D_2, ...D_n$ 为对空间 S 的一个划分, 则此空间中的向量 $\boldsymbol{x}$ 出现于空间 D_j 的概率可以表示为:

$$p(D_j|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|D_j)p(D_j)}{\sum_{i=1}^{n} p(\boldsymbol{x}|D_i)p(D_i)} \tag{1}$$

加入条件独立约束后, 朴素贝叶斯的上述概率表述可以改写为:

$$p(D_j|\boldsymbol{x}) = \frac{p(D_j)}{p((x))} \prod_{i=1}^{n} p(\boldsymbol{x}_i|D_j) \tag{2}$$

在文本分类任务下,D_j 即为某个文本类别,$\boldsymbol{x}$ 即为一条文本信息的特征。

2.4 算法描述

前置小节已说明朴素贝叶斯分类器原理, 不难看出朴素贝叶斯分类器在文本分类任务下, 核心任务为文本信息特征的表示。

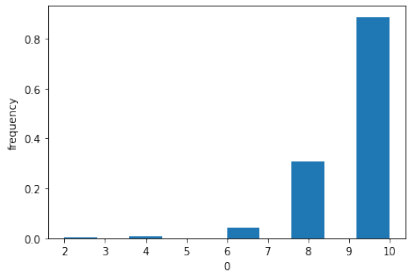

图 4 fit 类打分频率直方图

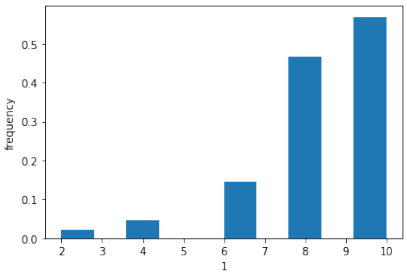

图 5 small 类打分频率直方图

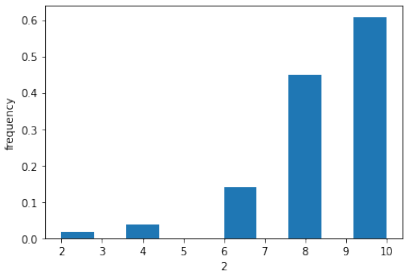

图 6 large 类打分频率直方图

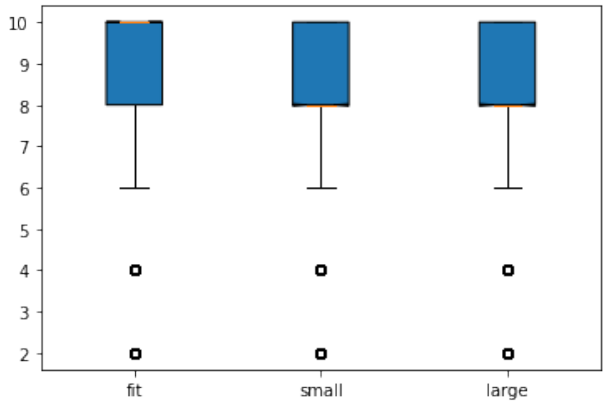

图 7 三类别打分箱线图

基于**基本假设 2**, 本文认为文本的词频特征是多项式分布的。因此本文选用词袋模型, 并用词频特征作为文本信息的特征表示。

即对从词袋 V 中选中的给定长度 n 的单词集合 $W = \{w_1, w_2, ..., w_n\}$, 任意文本的特征可以表示为 $\boldsymbol{x} = (x_1, x_2, ..., x_n)$, 其中 x_i 为 w_i 在文本中出现的次数。

后文将词袋中选中的单词集合 W 称为语料库。

理想情况下, 语料库为人类讨论特定主题的用词集合, 后文将理想情况的语料库标记为 W_c。

通常情况下有:$W \in V$, $W_c \in V$ 且 $W - W_c \neq \varnothing$。

此时, 集合 $N = W - W_c$ 即为语料库中的噪声特征集合, 集合 $L = W_c - W$ 即为语料库缺失的特征集合。

优化目标 由此, 算法设计的优化目标就在于如何令 W 逼近 W_c。本文将之分解为三个方面的挑战, 一是构建一个能够覆盖 W_c 的词袋 V; 二是选择 W 时如何尽可能避免选中 N 中的噪声特征; 三是选择 W 时如何尽可能包含 W_c 中的元素, 使得 L 趋近空集。

核心思路一 对于挑战一, 理想情况下, 词袋模型需要由尽可能大规模的, 实际场景的主题文本作为数据源构建。本实验中基于**基本假设 1** 简单认为 train.txt 即是符合真实场景的文本全集, 并以此构建词袋 (过滤技术细节在**技术细节**描述)。

核心思路二 对于挑战二, 本实验中需要实现一个特征过滤器, 在从词袋选择语料库的过程中尽可能多地过滤滤高频的主题无关单词, 例如 it,I,the 等。

词袋构建算法由此描述如下:

算法 1 词袋构建算法

输入: $train \leftarrow$ 训练集数据

```
vocab ← empty k,v map
words ← empty list
reviews ← review_texts of train
FOR review in reviews DO
    tokens ← tokenize(review)
    filteredTokens ← featureFilter(tokens)
    words.extend(tokens)
END FOR
FOR word in words DO
    IF word appears first time THEN
        vocab[word]←1
    ELSE
        vocab[word] + +
    END IF
END FOR
```

核心思路三 对于挑战三, 本文采用词频 top n 的单词集合作为语料库 W, 并迭代增加 W 的长度 n 同时在迭代时查看在训练集上的表现 p_n(本实验中目标函数为 $f1\_macro$ 函数)。当迭代到第 n_t 后表现不再提升时, 则认为当前的 W 包含了足够多的 W_c 元素和较多的噪声特征 N。此时选择满足 $p_t - p_i < \sigma$

的最小 n_i 作为理想长度 (本实验取 $\sigma = 0.002$), 对应的 W_i 为理想语料库。

具体语料库构建和模型优化伪代码描述如下:

算法 2 语料库构建算法

输入: $vocab = \{word : count\}$ ← 尺寸主题的词袋, n ← 语料库长度

1: $candidates \leftarrow vocab.words$ sorted by $count$
2: $corpus \leftarrow$ first n elements of $candidates$

算法 3 模型优化算法

输入: $train$ ← 训练集数据, $test$ ← 测试集数据, n ← 特征/语料库长度, m ← 词袋长度

1: $models \leftarrow$ empty map
2: **FOR** n from 1 to m **DO**
3: $nbmodel \leftarrow$ fit Naive Bayes model on train
4: $y\_pred \leftarrow predict\ reviews$ in test
5: $y\_true \leftarrow labels$ in test
6: $models[nbmodel] \leftarrow f1\_macro(y\_pred, y\_true)$
7: **END FOR**
8: $bestmodel \leftarrow (models.model$ sorted by $model.f1\_macro)[0]$

2.5 技术细节

数据集划分 : 本次实验中首先将 train.txt 中的有标签记录清洗掉含有 NaN 的标签, 随后按 fit 标签分为 fit,large,small 三个集合, 并在每个集合中采样 50% 作为训练集, 剩余部分作为测试集。详见代码中的 create_dataset 方法。

词袋创建 : 词袋和语料库创建算法已由上文描述, 本段落具体描述词袋创建时文本处理方法。在代码中文本的 tokenize 过程调用了 *nltk.tokenize* 的 *word_tokenize* 方法, 随后使用 *nltk.stem.WordLemmatizer* 进行语义标注。仅留下被标注为名词、动词、形容词、副词的 token。由于 *nltk* 的文本标注并非完全精准, 接着仍需过滤掉终结词, 这里终结词词库选用 *nltk.corpus.stopwords*。其中, 由于 *nltk.corpus.stopwords* 中包含否定词和部分多义状语如 down(e.g. I think I should have ordered a size down) 等, 这部分终结词被过滤会导致句子相关语义残缺[2], 因此需要从终结词集合中删除这一部分。

特征提取 : 前文已述, 对于文本 text, 其特征 $(x_1, x_2, ..., x_n)$ 中的 x_1 指词袋的词频 top 1 单词在 text 中出现的次数,n 为特征长度/语料库长度。

模型训练与调优 : 模型架构上选择 *sklearn.naivebayes.ComplementNB* 前文已述模型调优的具体算法, 本次试验中尝试的特征长度/语料库长度范围为 (0, 1020], 步长为 20, 同时补充了长度为 2020 和 4000 的数据 (由于 400 至 1020 的 f1_macro) 提升十分平缓, 笔者认为大致在 1020 前后收敛, 故补充 2020 和 4000 作为验证)。具体分析见性能分析章节。

3 性能分析

本实验中选取特征长度/语料库长度 (0, 1020] 分别训练模型并在验证集上计算 f1_macro 函数, 验证特征长度对模型表现的影响, 数据如下表:

表 1 文本特征长度对模型表现的影响

feature length	accuracy	f1_macro
20	0.5821156525089659	0.4066587771931469
60	0.7037552980230594	0.5508248050617507
100	0.7043777231097543	0.5557230329516174
200	0.7092089273541006	0.5698526222237081
400	0.7126619046207653	0.5797990906283427
620	0.7116689884110377	0.5806248121838539
1020	0.7105575150419396	0.5812550759032398
2020	0.709371943448235	0.5805696852219103
4000	0.7077269628619698	0.5783782599391908

实验数据分析 : 从表中可知在 1020 附近 f1_macro 最优, 而从 400 至 1020 的指标提升缓慢, 在 1020 后总体呈现下降趋势。这是因为对于当前词袋, 词频 top 400 的单词已经涵盖了大部分的有效特征, 后续仍然存在少量有效特征, 但同时噪声特征也在逐渐增加, 因而导致指标提升缓慢, 同时有 overfit 的风险。同时有 $p_1020 - p_400 < \sigma$, 故长度为 400 的语料库构建的模型即为本文的最优模型, **提交预测结果 output_DZ1933026.txt 也由此模型预测**。

训练开销 : 单次训练过程包括从训练集加载数据, 提取特征和标签,fit 模型, 从测试集加载数据, 提取特征和标签, 计算 f1_macro, 计算 test.txt 标签六个步骤。其中从训练集/测试集加载数据平均耗时约 4 分

36 秒, 从 test.txt 计算标签同样涉及加载数据和特征提取, 平均耗时约 4 分 54 秒。实验设备为 Linux n33 4.15.0-111-generic #112-Ubuntu SMP x86_64 x86_64 x86_64 GNU/Linux, CPU 规格为: Intel(R) Xeon(R) Gold 5118 CPU @ 2.30GHz, 未使用 GPU 或分布式并行加速。

4 总结

本文描述了本次笔者本次试验中使用朴素贝叶斯和词频特征, 通过文本分类构建尺寸匹配情况分类器的过程及结果分析。

进一步优化的方向 ：在本实验中基本确定当前词袋的 top 400-1000 即可较好地覆盖尺寸匹配情况的相关词, 但此时特征中仍然有不少噪声, 进一步地过滤掉词袋中的噪声词想必可以进一步提升指标。本次实验中仅采用 train.txt 中的数据来构建词袋, 而实际上词袋中的数据越多越好, 通过加入 test.txt 或是更进一步扩大词库, 使其更接近于实际任务分布, 则可以进一步提升模型效果。

参考文献

[1] SEBASTIANI F. Machine learning in automated text categorization[J]. ACM computing surveys (CSUR), 2002, 34(1): 1-47.

[2] NARAYANAN V, ARORA I, BHATIA A. Fast and accurate sentiment classification using an enhanced naive bayes model[C]//International Conference on Intelligent Data Engineering and Automated Learning. [S.l.]: Springer, 2013: 194-201.