

## 【内容推荐】从文本到用户画像有多远

2018-03-14 刑无刀



### 【内容推荐】从文本到用户画像有多远

朗读人：黄洲君 19'02" | 6.55M

前面，我和你聊过了不要把用户画像当成银弹，也不要觉得一无是处。对于一个早期的推荐系统来说，基于内容推荐离不开为用户构建一个初级的画像，这种初级的画像一般叫做用户画像（User Profile），一些大厂内部还习惯叫做 UP，今天我就来讲一讲从大量文本数据中挖掘用户画像常常用到的一些算法。

### 从文本开始

用户这一端比如说有：

1. 注册资料中的姓名、个人签名；
2. 发表的评论、动态、日记等；
3. 聊天记录（不要慌，我举个例子而已，你在微信上说的话还是安全的）。

物品这一端也有大量文本信息，可以用于构建物品画像（Item Profile），并最终帮助丰富用户画像（User Profile），这些数据举例来说有：

1. 物品的标题、描述；
2. 物品本身的内容（一般指新闻资讯类）；

### 3. 物品的其他基本属性的文本。

文本数据是互联网产品中最常见的信息表达形式，数量多、处理快、存储小，因为文本数据的特殊地位，所以我今天专门介绍一些建立用户画像过程中用到的文本挖掘算法。

## 构建用户画像

要用物品和用户的文本信息构建出一个基础版本的用户画像，大致需要做这些事：

1. 把所有非结构化的文本结构化，去粗取精，保留关键信息；
2. 根据用户行为数据把物品的结构化结果传递给用户，与用户自己的结构化信息合并。

第一步最关键也最基础，其准确性、粒度、覆盖面都决定了用户画像的质量。仿佛如果真的要绘制一个用户的模样，要提前给他拍照，这个拍照技术决定了后面的描绘情况，无论是采用素描、油画、工笔还是写意。这一步要用到很多文本挖掘算法，稍后会详细介绍。

第二步会把物品的文本分析结果，按照用户历史行为把物品画像（Item Profile）传递给用户。你也许会问：传递是什么意思？没关系，这个稍后我会介绍。

### 一、结构化文本

我们拿到的文本，常常是自然语言描述的，用行话说，就是“非结构化”的，但是计算机在处理时，只能使用结构化的数据索引，检索，然后向量化后再计算；所以分析文本，就是为了将非结构化的数据结构化，好比是将模拟信号数字化一样，只有这样才能送入计算机，继续计算。这个很好理解，不多解释。

从物品端的文本信息，我们可以利用成熟的 NLP 算法分析得到的信息有下面几种。

1. 关键词提取：最基础的标签来源，也为其他文本分析提供基础数据，常用 TF-IDF 和 TextRank。
2. 实体识别：人物、位置和地点、著作、影视剧、历史事件和热点事件等，常用基于词典的方法结合 CRF 模型。
3. 内容分类：将文本按照分类体系分类，用分类来表达较粗粒度的结构化信息。
4. 文本：在无人制定分类体系的前提下，无监督地将文本划分成多个类簇也很常见，别看不是标签，类簇编号也是用户画像的常见构成。
5. 主题模型：从大量已有文本中学习主题向量，然后再预测新的文本在各个主题上的概率分布情况，也很实用，其实这也是一种聚类思想，主题向量也不是标签形式，也是用户画像的常用构成。
6. 嵌入：“嵌入”也叫作 Embedding，从词到篇章，无不学习这种嵌入表达。嵌入表达是为了挖掘出字面意思之下的语义信息，并且用有限的维度表达出来。

下面我来介绍几种常用的文本结构化算法。

## 1 TF-IDF

TF 全称就是 Term Frequency，是词频的意思，IDF 就是 Inverse Document Frequency 是逆文档频率的意思。TF-IDF 提取关键词的思想来自信息检索领域，其实思想很朴素，包括了两点：在一篇文字中反复出现的词会更重要，在所有文本中都出现的词更不重要。非常符合我们的直觉，这两点就分别量化成 TF 和 IDF 两个指标：

1. TF，就是词频，在要提取关键词的文本中出现的次数；
2. IDF，是提前统计好的，在已有的所有文本中，统计每一个词出现在了多少文本中，记为  $n$ ，也就是文档频率，一共有多少文本，记为  $N$ 。

IDF 就是这样计算：

$$IDF = \log \frac{N}{n + 1}$$

计算过程为：词出现的文档数加 1，再除总文档数，最后结果再取对数。

IDF 的计算公式有这么几个特点：

1. 所有词的  $N$  都是一样的，因此出现文本数越少 ( $n$ ) 的词，它的 IDF 值越大；
2. 如果一个词的文档频率为 0，为防止计算出无穷大的 IDF，所以分母中有一个 1；
3. 对于新词，本身应该  $n$  是 0，但也可以默认赋值为所有词的平均文档频率。

计算出 TF 和 IDF 后，将两个值相乘，就得到每一个词的权重。根据该权重筛选关键词的方式有：

1. 给定一个  $K$ ，取 Top  $K$  个词，这样做简单直接，但也有一点，如果总共得到的词个数少于  $K$ ，那么所有词都是关键词了，显然这样做不合理；
2. 计算所有词权重的平均值，取在权重在平均值之上的词作为关键词；

另外，在某些场景下，还会加入以下其他的过滤措施，如：只提取动词和名词作为关键词。

## 2 TextRank

TextRank 这个名字看上去是不是和著名的 PageRank 有点亲戚关系？是的，TextRank 是 PageRank 的私生子之一，著名的 PageRank 算法是 Google 用来衡量网页重要性的算法，TextRank 算法的思想也与之类似，可以概括为：

1. 文本中，设定一个窗口宽度，比如 K 个词，统计窗口内的词和词的共现关系，将其看成无向图。图就是网络，由存在连接关系的节点构成，所谓无向图，就是节点之间的连接关系不考虑从谁出发，有关系就对了；
2. 所有词初始化的重要性都是 1；
3. 每个节点把自己的权重平均分配给 “和自己有连接” 的其他节点；
4. 每个节点将所有其他节点分给自己的权重求和，作为自己的新权重；
5. 如此反复迭代第 3、4 两步，直到所有的节点权重收敛为止。

通过 TextRank 计算后的词语权重，呈现出这样的特点：那些有共现关系的会互相支持对方成为关键词。

### 3 内容分类

在门户网站时代，每个门户网站都有自己的频道体系，这个频道体系就是一个非常大的内容分类体系，这一做法也延伸到了移动互联网 UGC 时代，图文信息流 App 的资讯内容也需要被自动分类到不同的频道中，从而能够得到最粗粒度的结构化信息，也被很多推荐系统用来在用户冷启动时探索用户兴趣。

在门户时代的内容分类，相对来说更容易，因为那时候的内容都是长文本，长文本的内容分类可以提取很多信息，而如今 UGC 当道的时代，短文本的内容分类则更困难一些。短文本分类方面经典的算法是 SVM，在工具上现在最常用的是 Facebook 开源的 FastText。

### 4 实体识别

命名实体识别（也常常被简称为 NER，Named-Entity Recognition）在 NLP 技术中常常被认为是序列标注问题，和分词、词性标注属于同一类问题。

所谓序列标注问题，就是给你一个字符序列，从左往右遍历每个字符，一边遍历一边对每一个字符分类，分类的体系因序列标注问题不同而不同：

1. 分词问题：对每一个字符分类为“词开始”“词中间”“词结束”三类之一；
2. 词性标注：对每一个分好的词，分类为定义的词性集合的之一；
3. 实体识别：对每一个分好的词，识别为定义的命名实体集合之一。

对于序列标注问题，通常的算法就是隐马尔科夫模型（HMM）或者条件随机场（CRF），我们在推荐系统中主要是挖掘出想要的结构化结果，对其中原理有兴趣再去深入了解。

实体识别还有比较实用化的非模型做法：词典法。提前准备好各种实体的词典，使用 trie-tree 数据结构存储，拿着分好的词去词典里找，找到了某个词就认为是提前定义好的实体了。

以实体识别为代表的序列标注问题上，工业级别的工具上 spaCy 比 NLTK 在效率上优秀一些。

## 5 聚类

传统聚类方法在文本中的应用，今天逐渐被主题模型取代，同样是无监督模型，以 LDA 为代表的主题模型能够更准确地抓住主题，并且能够得到软聚类的效果，也就是说可以让一条文本属于多个类簇。

作为初创公司或初创产品，我知道你的时间宝贵，也知道你的公司处处节俭，以至于没有业务专家为你的应用制定分类体系，这时候如果能在文本数据上跑一个 LDA 模型，那世界就显得非常美好了。

LDA 模型需要设定主题个数，如果你有时间，那么这个 K 可以通过一些实验来对比挑选，方法是：每次计算 K 个主题两两之间的平均相似度，选择一个较低的 K 值；如果你赶时间，在推荐系统领域，只要计算资源够用，主题数可以尽量多一些。

另外，需要注意的是，得到文本在各个主题上的分布，可以保留概率最大的前几个主题作为文本的主题。LDA 工程上较难的是并行化，如果文本数量没到海量程度，提高单机配置也是可以的，开源的 LDA 训练工具有 Gensim，PLDA 等可供选择。

## 6 词嵌入

关于嵌入，是一个数学概念。以词嵌入为例来说吧。

词嵌入，也叫作 Word Embedding。前面讲到的结构化方案，除了 LDA，其他都是得到一些标签，而这些标签无一例外都是稀疏的，而词嵌入则能够为每一个词学习得到一个稠密的向量。

这样说可能很抽象，换个说法，一个词可能隐藏很多语义信息，比如北京，可能包含“首都、中国、北方、直辖市、大城市”等等，这些语义在所有文本上是有限的，比如 128 个，于是每个词就用一个 128 维的向量表达，向量中各个维度上的值大小代表了词包含各个语义的多少。

拿着这些向量可以做以下的事情：

1. 计算词和词之间的相似度，扩充结构化标签；
2. 累加得到一个文本的稠密向量；
3. 用于聚类，会得到比使用词向量聚类更好的语义聚类效果。

这方面当然就属大名鼎鼎的 Word2Vec 了。Word2Vec 是用浅层神经网络学习得到每个词的向量表达，Word2Vec 最大的贡献在于一些工程技巧上的优化，使得百万词的规模在单机上可以几分钟轻松跑出来，得到这些词向量后可以聚类或者进一步合成句子向量再使用。

## 二、标签选择

前面说到，用户端的文本，物品端的文本如何结构化，得到了诸如标签（关键词、分类等）、主题、词嵌入向量。接下来就是第二步：如何把物品的结构化信息给用户呢？

我们想一想，用户在产品上看到了很多我们用各种逻辑和理由展示给他的物品，他只从中消费了一部分物品。现在问题就是，到底是那些特性吸引了他消费呢？

一种简单粗暴的办法是直接把用户产生过行为的物品标签累积在一起，但是这里要说的是另一种思路。

我们把用户对物品的行为，消费或者没有消费看成是一个分类问题。用户用实际行动帮我们标注了若干数据，那么挑选出他实际感兴趣的特性就变成了特征选择问题。

最常用的是两个方法：卡方检验（CHI）和信息增益（IG）。基本思想是：

1. 把物品的结构化内容看成文档；
2. 把用户对物品的行为看成是类别；
3. 每个用户看见过的物品就是一个文本集合；
4. 在这个文本集合上使用特征选择算法选出每个用户关心的东西。

## 1 卡方检验

CHI 就是卡方检验，本身是一种特征选择方法。

前面的 TF-IDF 和 TextRank 都是无监督关键词提取算法，而卡方检验（CHI）则是有监督的，需要提供分类标注信息。

为什么需要呢？在文本分类任务中，挑选关键词就得为了分类任务服务，而不仅仅是挑选出一种直观上看着重要的词。

卡方检验本质上在检验“词和某个类别 C 相互独立”这个假设是否成立，和这个假设偏离越大，就越说明这个词和类别 C 暗中有一腿，那当然这个词就是关键词了。

计算一个词  $W_i$  和一个类别  $C_j$  的卡方值，需要统计四个值：

1. 类别为  $C_j$  的文本中出现词  $W_i$  的文本数 A；
2. 词  $W_i$  在非  $C_j$  的文本中出现的文本数 B；
3. 类别为  $C_j$  的文本中没有出现  $W_i$  的文本数 C；
4. 词  $W_i$  在非  $C_j$  的文本中没有出现的文本数 D。

听起来有点绕，我把它画成一个表格更加一目了然。

卡方检验	属于类别 $C_j$	不属于类别 $C_j$	总计
包含词 $W_i$	A	B	A+B
不包含 $W_i$	C	D	C+D
总计	A+C	B+D	N=A+B+C+D

然后按照如下公式计算每一个词和每一个类别的卡方值：

$$\chi^2(W_i, C_j) = \frac{N(AD - BC)^2}{(A + C)(A + B)(B + D)(C + D)}$$

关于这个卡方值计算，我在这里说明几点：

1. 每个词和每个类别都要计算，只要对其中一个类别有帮助的词都应该留下；
2. 由于是比较卡方值的大小，所以公式中的 N 可以不参与计算，因为它对每个词都一样，就是总的文本数；
3. 卡方值越大，意味着偏离“词和类别相互独立”的假设越远，靠“词和类别互相不独立”这个备择假设越近。

## 2 信息增益

IG 即 Information Gain，信息增益，也是一种有监督的关键词选择方法，也需要有标注信息。要理解信息增益，理解了信息熵就差不多了。

信息熵，一批文本被标注了类别，那么任意挑出一条文本问你，“猜猜这是什么类别？”如果原来每个类别的文本数量都一样，那你肯定最不好猜，如果其中一个类别的文本 C 数远远多于其他类别，那么你猜这条文本属于类别 C 就很可能猜对。这两个情况区别就在于信息熵不同：

1. 各个类别的文本数量差不多时，信息熵就比较大。
2. 其中少数类别的文本数量明显较多时，信息熵就较小。

进一步再想一件事，如果从这一堆文本中再挑出包含词 W 的文本数，再来猜任意一条文本的类别时，仍然会有上面两种情况。这时候考虑一个情况：如果在整个文本上的情况是 1，但挑出包含词 W 后的情况变成 2 了，那么你想，这个词 W 是不是非常有用？因为有了它，我们就能以较高的成功率猜对任意一条文本的类别了。

对，上面这个思考过程就是信息增益的思想，信息增益计算也是分成三步：

1. 统计全局文本的信息熵；
2. 统计每个词的条件熵，就是知道了一个词后再统计文本的信息熵，只不过这里要分别计算包含词和不包含词两部分的信息熵，再按照各自文本比例加权平均；
3. 两者相减就是每个词的信息增益。

信息增益应用最广就是数据挖掘中的决策树分类算法，经典的决策树分类算法挑选分裂节点时就是计算各个属性的信息增益，始终挑选信息增益最大的节点作为分裂节点。

卡方检验和信息增益不同之处在于：前者是针对每一个行为单独筛选一套标签出来，后者是全局统一筛选。

这些方法都是在离线阶段批量完成的，把用户的画像生成配置成离线任务，每天更新一遍，次日就可以使用新的用户画像。

那么对于一个新用户，能不能在他刚刚进入产品时就能够快速生成一个简单的画像呢？答案是：当然可以。这在后面的专栏中会讲到的 MAB 问题。

## 总结

用户画像对于推荐系统还是非常必要的，而产品中属文本数据最多，那如何用文本数据构建出用户的画像内容呢？

本文按照如下步骤梳理了这一过程：

1. 分析用户的文本和物品的文本，使其结构化；
2. 为用户挑选有信息量的结构化数据，作为其画像内容。

其中，我们提出了把为用户挑选画像标签看成是特征选择问题，除了卡方检验和信息增益，你还知道有哪些特征选择方法呢？欢迎留言一起讨论。

感谢你的收听，我们下次再见。





版权归极客邦科技所有，未经许可不得转载

#### 精选留言



张哲

这一期信息量好大.....

2018-03-14

作者回复

慢慢享用，嚼碎了再吞。

2018-03-14

6



Drxan

无刀老师，能否建立个微信群啊，大家可以对您每期的课程内容一起学习讨论

2018-03-15

作者回复

知识星球(原来叫小密圈)搜ResysChina。

2018-03-15

5



jt120

针对embedding，我不太理解，之前理解就是一种映射关系，但文里为什么说结果是稠密的，这是怎么保证的

2018-03-14

作者回复

用Word2vec跑出一个结果你就明白了，设定k维的话，你会得到一个k维向量，每个维度上都有值的。

2018-03-14

4



行行行

2



老师，关于word2vec，有几个疑问

- 1 工业上如果通过word2vec得到文档的向量呢，是用累加一个文档中各个词的词向量得到的稠密向量表示吗
- 2 用于聚类，是用上面得到的文档向量来做吗
- 3 到底是如何通过计算词和词之间的相似度，扩充标签的呢

2018-03-23

#### 作者回复

1.是

2.是

3.用你文本中出现的词，去查询通过word2vec计算得到的相似词，从而实现了“扩展”，原来只有两个词，现在增加了几个相似词，变成了4个词。

2018-03-23



林彦

2

谢谢刑无刀老师的分享。建议各种步骤和场景的工业化工具及其性能和便利，对我们读者提供了不小的价值。

1. 词嵌入里面提到学习到的包含更多语义信息的(新)词向量可以“用于聚类，会得到比使用词向量聚类更好的语义聚类效果”。这里的聚类是指文中之前提到的LDA等聚类模型来找出文章的主题吗？

2. “主题模型：从大量已有文本中学习主题向量，然后再预测新的文本在各个主题上的概率分布情况，也很实用，其实这也是一种聚类思想，主题向量也不是标签形式，也是用户画像的常用构成。”请问这里已有文本的主题向量中的主题词如果是通过LDA提取是不是需要有个停用词表排除那些所有文档中词频都很高的词？它不像TF-IDF会自动排除所有文档中词频高的词。这种场景的聚类就是判别新的文本和哪些主题的文本比较相似（“距离”接近或主题”概率“较大），然后判别新的文本的主题？

3. “向量中各个维度上的值大小代表了词包含各个语义的多少”有这句说明挺好的。我第一次阅读不太理解，后来查了一些文章，有一个解释我觉得比较直观，Word2Vec生成的向量值可以看成是N维语义空间(N个语义词)中的坐标值(每个坐标轴对应一个语义)。当2个词在同一个N维语义空间中的距离接近时，说明2个词的含义接近。

2018-03-14

#### 作者回复

你一定是个学霸，向你致敬。

1. 这里的聚类是指的传统数据挖掘中的聚类。基于距离或者密度等。如kmeans。
2. 停用词要去掉。Lda在预测阶段也要迭代的，而不是计算相似度。
3. 你若理解了，就是晴天。

2018-03-15



三竹先生

1

真的好难啊，邢老师能多结合一下例子吗？方便理解🤔🤔🤔

2018-03-29



jt120

👍 1

上面提到的都是主流的推荐系统方法，例如电影，图书，新闻这些经典场景。但对于特殊商品，例如房子，明显和上面不同，低频，高价，并且房子的特征基本都是分类特征，针对这种场景，如何选择特征，如何推荐了？

2018-03-14

作者回复

这种场景下，用户也更有耐心一些，可以多向用户询问他的诉求，更像一个辅助决策系统，而不是一个纯粹的推荐系统。

2018-03-14



那年岁月

👍 0

问一下老师，w2v算法把文章中没一个词都做向量化，是把文章所有词向量累加还是等长关键词累加，如果把所有词累加，那么对篇幅小的文章不公平。

2018-05-04

作者回复

可以求平均。

2018-05-10



mgxs

👍 0

老师您好，我有一个疑问。最后特征选择后得到的特征直接作为用户画像，每个特征在用户画像里头的有具体的取值吗？还是都认为是1？后面推荐的时候如果对物品同样的结构化完，得到所有特征的结构化表示，物品的维度数和特征选择后的用户画像维度数不一样，那么如何基于相似度匹配来推荐呢？

2018-05-02



top savor

👍 0

老师您好，我是初学者，感觉学到很多，但还是有些疑问：请问用户画像如果是用户本身的特征（比如个性签名的文本向量化）+物品端的特征，那在做匹配计算相似度的时候，两者的向量维度岂不是不一样？

2018-04-18



Lisa Li

👍 0

“每次计算 K 个主题两两之间的平均相似度，选择一个较低的 K 值”  
可以举个例子吗？

2018-04-05



尹士

👍 0

Fasttext准确率跟cnn比，有差距，我的实验结果，不知邢老师参数如何设置的，可以工程中使用fasttext

2018-04-01

## 作者回复

我似乎觉得工程中fasttext用得更多些，尤其中小企业。

2018-04-01



吴文敏

👍 0

根据用户点击过的商品来做用户画像，假如采用信息增益的方法。可否理解成对每个用户计算商品集的每个标签对用户点击因为的贡献率，然后选取贡献率超过某个阈值的标签作为用户标签。这种做法是不是只对点击过足够多商品的用户比较有效？

2018-03-29



小刀子轩

👍 0

这期涉及好几个算法，要消化消化

2018-03-27



行行行

👍 0

老师我的理解是新增益越大特征越有区分度，如果这个特征在正例中，用户和这个特征越相关，但如果这个特征正例和负例中都有呢，该怎么算

2018-03-23

## 作者回复

信息增益就是用来计算特征出现在正例和负例中的。先计算整体上正例负例的熵，再计算在已知特征的条件下正负例的熵，两者差值就是信息增益。

2018-03-23



清风六十五

👍 0

简洁扼要，很不错

2018-03-15

## 作者回复

欢迎传阅。

2018-03-16



👍 0

一到实际内容的时候，就会在听说过的和没听说过的，懂了的还有不知道是什么的知识点里无法理顺思路了

2018-03-14

## 作者回复

不用管知识点，先一把梭干起来。

2018-03-15



Drxan

👍 0

无刀老师，不好意思早晨地铁上把字打错了，请见谅！文中提到利用每个用户所见过的物品的文档集合来生成用户画像特征，一般是如何确定哪些物品是用户见过的呢？如果我只有用户的购买记录，而没有浏览、收藏等记录，那是否可以从用户没有购买过的商品中随机抽取部份商品作为负样本呢

2018-03-14

作者回复

我貌似已经回复过你了。

2018-03-15



莘哥 ( cheung ! )

都有哪些成熟的用户画像开发框架呢？

2018-03-14

作者回复

这个貌似没有，但相关工具都有开源的。

2018-03-14

0



叶晓锋

重要的应该是把文本数据转换成机器能处理的量化数据

2018-03-14

作者回复

对。特征化。

2018-03-14

0