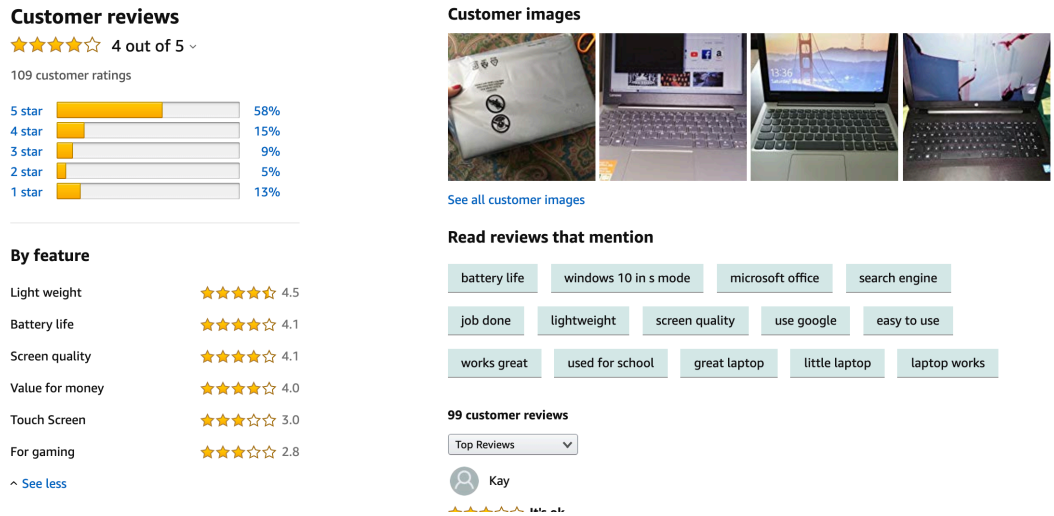
项目：基于类别的情感分析系统

1. 项目背景

情感分析系统是自然语言处理领域最为经典的应用之一，一直长盛不衰。特别是，移动互联网的发展极大提高了每个人的参与度。比如在淘宝上买东西，很多人会去填写简单几句话作为评论；去了餐馆之后，很多人也会在大众点评上留下自己对本次的评价；这些数据一方面可以让每一位用户清楚地看到每个商家所提供的服务质量，同时让一个商家也意识到自己的问题所在。 但是面对大量的评论数据，如何让一个用户或者商家能够更方便地看到全貌呢？ 这就是本项目中需要解决的问题。

2. 项目描述



上面的这幅图是针对于一个项目的评价。 “Customer reviews”栏里所展示的是对产品的**整体**评价， “By feature”栏里描述的是对于一个商品每一个**维度**的评价。这使得我们对产品的认识一目了然。这就是本项目中需要完成的任务，也叫作 Aspect-Based Sentiment Analysis。

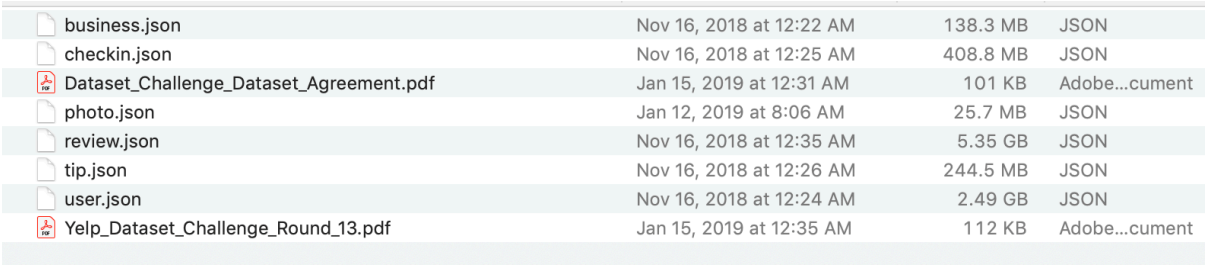
我们经常讨论的 Sentiment Analysis（情感分析）一般只是来判断一个句子或者文本的情感是什么（正面还是负面），但不会考虑到每个细节上。在本项目中不仅要考虑到整体的情感，也需要考虑到用户对产品的每个方面的评价也需要抽取出来。这里的每一个方面也叫作 aspect。而且一个用户评价中可能会存在多个 aspect 比如”我对这款产

品的电池比较满意，但它太贵了！”， 从这句话里我们可以得出：‘电池’：正面，‘价格’：负面” 。

3. 数据和任务

在本项目中，我们使用的开源数据是 Yelp DataSet， Yelp 对标国内的大众点评，在此平台上可以找到大量的用户评论，数据链接：

https://www.yelp.com/dataset/documentation/main



business.json	Nov 16, 2018 at 12:22 AM	138.3 MB	JSON
checkin.json	Nov 16, 2018 at 12:25 AM	408.8 MB	JSON
Dataset_Challenge_Dataset_Agreement.pdf	Jan 15, 2019 at 12:31 AM	101 KB	Adobe...cument
photo.json	Jan 12, 2019 at 8:06 AM	25.7 MB	JSON
review.json	Nov 16, 2018 at 12:35 AM	5.35 GB	JSON
tip.json	Nov 16, 2018 at 12:26 AM	244.5 MB	JSON
user.json	Nov 16, 2018 at 12:24 AM	2.49 GB	JSON
Yelp_Dataset_Challenge_Round_13.pdf	Jan 15, 2019 at 12:35 AM	112 KB	Adobe...cument

下载完数据之后可以看到上面的目录结构。本项目中，我们使用”business.json”, “review.json”，暂时可以不考虑其他的数据文件。 如果下载这个数据比较慢，直接可以从百度网盘上下载：

链接:https://pan.baidu.com/s/1RGl6TzQtpW6EHoo3t7YBKg 密码:x0db

business.json: 用来描述一个 business，包括地理位置，属性，邮编等信息

review.json: 一个用户对一个 business 的评价，这里包括具体的评价文本还有 stars。

```
{
    // string, 22 character unique review id
    "review_id": "zdSx_SD6obEhz9VrW9uAWA",

    // string, 22 character unique user id, maps to the user in user.
    "user_id": "Ha3iJu77CxlrFm-vQRs_8g",

    // string, 22 character business id, maps to business in business
    "business_id": "tnhfDv5Il8EaGSXZGiuQGg",

    // integer, star rating
    "stars": 4,

    // string, date formatted YYYY-MM-DD
    "date": "2016-03-09",

    // string, the review itself
    "text": "Great place to hang out after work: the prices are decer

    // integer, number of useful votes received
    "useful": 0,

    // integer, number of funny votes received
    "funny": 0,

    // integer, number of cool votes received
    "cool": 0
}
```

所以针对于一个 Business，可能有大量的评论，我们需要通过以上的数据来整理出针对于每一个 business 的 summary of reviews，具体输出如下（例子）：

Business Name: XXXXX
Overall Rating: X
Detailed Rating:
aspect1: { rating: XXX, pos: [XXX], neg: [XXX]}
aspect2: {rating: XXX, pos: [XXX], neg: [XXX]}
aspect3: {rating: XXX, pos: [XXX], neg: [XXX]}
……
aspect5: {rating: XXX, pos:[xxx], neg:[xxxx]}

标记为红色的部分是需要学出来的部分。这里 pos, neg 表示的是输出前 5 个表示某一个方面的正面或者负面的评价。 举个例子，比如对于贪心学院网络上可能有很多用户的评论，之后通过评论可以总结出：

Business Name: 贪心学院
Overall Rating: 4.6
Detail Rating:

教学质量：{rating: 4.9, pos: [“我上过贪心学院的 2 门课程，教学质量真的棒极了 “，” 其实没有比他们教学质量更好的 “，” 自然语言处理课程内容非常好，质量很高！ “，” 收获很大，强烈推荐！ “，” 都挺好的 “], neg: [“整体上都挺好，可能就是时长比较长”， “讲得很清楚，但有点太难了！”， “有难度，我之前就跟不上”，”难！ “，” 作业不容易啊！ “, “还行吧”]}

教学服务：{rating: 4.8, pos: [“班主任很热情，服务很到位！ “，” 助教服务也非常好，回答很及时 “，” xxxx “, “xxx”, “xxx”], neg: [“xxx”, “xxx”, “xxx”, “xxx”, “xxx”]}

教师能力：{XXXXX}

上面的就是本项目需要输出的部分。对于每一个 business unit，需要整理出一份总结性的报告，让用户看到之后一目了然

4. 具体方法

在这里简单说一下完成此项目中需要涉及到的方法论，这个方法也可以理解成是 Baseline。 如果你想挑战自己，也可以想想更好的方法，会有很多种做法。 我们按照上述给定的模板来一个一个看一下。

4.1 Business Name: XXXXX 和 Overall Rating: XXX

对于这两个值很容易拿到，Business Name 就是直接提取出来的，商家名字； Overall Rating 就简单直接取一下平均就可以了。

4.2 Aspect 的提取

对于每一种商品用户所关注到的方面是不一样的。比如对于笔记本电脑，用户关注到的可能是轻便性，价格，性能； 对于餐馆用户关注到的是味道，服务，停车场等方面；所以需要针对于每一个 Business 提取 aspects，本项目中提取 top 5 aspects。

那如何自动提取 aspect 呢？本项目采用自动提取的方式。原理很简单，看哪一个方面被提及的最多，从中选择被提到最多的 5 个 aspects。具体做法如下：

```
# 针对于每一个 business，循环所有的评论，然后从评论中提取 aspects, 然后加入到
# aspects_dic
for review in business_id.get_reviews():
        extract_and_add_aspects(aspects_dic, review)

# 对于 aspects_dic 做排序，提取最高的前 5 个
```

在这里，我们针对于每一个 business 自动提取 aspects，这就要求评论数不能太少。因为评论数少的话，提取出来的 aspects 就不具备代表性了！所以本项目中，我们先过滤掉一些 business, 比如评论数达不到 100 的可以过滤掉，只考虑那些评论较多的商家。

上述过程里，唯一可能有疑问的是如何抽取 aspects。对于这个问题方法很多，在这里我们采用最简单的方法：通过 POS tagger 提取 noun phrase（名词短语）。 POS tagger 就是词性标注器，对于英文可以使用 NLTK 工具。具体用法可以搜一下 NLTK 的官方文档。所以通过 NLTK 把每一条评论里的名词短语（noun phrase）标记出来，并且记录下出现的次数，最终就可以统计出出现次数最多的 TOP 5。 注意：aspect 可以是一个单词（比如 price），也可以由多个单词构成（比如 parking cost）。

好了，通过上述过程我们可以自动提取出 aspects 了！

4.3 aspect1: { rating: XXX, pos: [XXX], neg: [XXX]}

接下来就是要解决对于某一个 aspect，如何获取它的 rating, 还有具有代表性的正面和负面评论。 需要通过如下步骤：

```
pos_sent = {}   # 存放正面的评论以及情感
neg_sent = {}   # 存放负面的评论以及情感
for review in business_id.get_reviews():
        if review.contains(aspect1):
                review_segment = get_segment(review, aspect)
                score = sentiment_model.predict_prob(review_segment)
                if score > threshold:
```

```
            pos_sent[review] = score
    else:
            neg_sent[review] = score
```

在这里，pos_sent 和 neg_sent 分别存放针对于某一个 business，某一个方面（aspect）的正面和负面的评价。之后可以通过排序的方式来获取评价值最高的正面评语和评价值最低的负面评语。 这里设定了 threshold 变量，可以自行设定。

一个关键步骤是如何判断一个评语有没有包含指定的 aspect。这个问题也有很多种方法，在本项目中使用的仍然是最简单的方法：判断一个字符串有没有存在于一个句子中。当然这种方法有自己的局限性。

另外，一个句子中可能包含多个 aspects，比如 “the price is good，but the location is bad”，所以如果想对”price”部分做情感分析，我们需要提取出 “the price is good”部分，这个过程是由 get_segment 函数来完成。那这部分如何提取呢？ 其实也可以使用很简单的方法：标注一下每一个 aspect 出现的部分，然后截取就可以了。 比如上述句子中， the price[aspect] is good, but the location[aspect] is bad， 那这时候就可以截取到逗号部分。那如果没有逗号怎么办？ 你也可以提出一些更精细化的规则（想一想）。但假如一个句子中只包含一个 aspect, 那就很简单，不需要做任何的句子分割。

还有一个问题是如何判断一个句子或者一个 segment 的情感？ 这个问题其实就是二分类问题，需要提前训练好才行。 那如何去训练呢？

```
// integer, star rating
"stars": 4,

// string, date formatted YYYY-MM-DD
"date": "2016-03-09",

// string, the review itself
"text": "Great place to hang out after work: the prices are decer
```

一种训练方式是把所有库里的 review 拿出来，然后根据这里给定的 stars 来训练。

```
pos_reviews = []
neg_reviews = []
for business_id in business.get_all_business():
        for review in business_id.get_reviews():
                if review.stars >= 4:
                        pos_reviews.add(review)
                if review.stars <=2:
                        neg_reviews.add(review)
                else:
                        // nothing
```

把库里的所有的 review 做一个循环，然后把正面和负面的情感全部拿出来，然后训练一个二分类的模型。训练模型上我们可以使用 SVM，Boosting， 朴素贝叶斯等模型。可以自行选择一个合适的模型来训练。另外，在这里，我们使用了一个简单的规则来判断哪些是正面的哪些是负面的情感。

到此为止，一个经典的 baseline 方法说完了。在本次项目作业中，至少要完成此 baseline。

5. 其他需要考虑的点

在此项目中，还有很多点可以考虑，可以按照自己的兴趣点来适当拓展一下。

5.1 Aspect 的抽取和匹配

在此问题中，aspect 的提取是一个核心问题。在上述 baseline 里面，我们从每一个 business 的 review text 里提取出了出现次数最多的名词短语作为 aspect，但这种做法是有很多缺点的，所以也会有其他的方法。

提前设计好针对于每一种 business 的 aspects。比如对于饭店、健身房、商场有各自的提前设计好的 aspects。这种做法的最大优点是准确，但缺点是比较费时间，需要一个一个整理，因为可能有成千上万个不同种类的 business。 但在工业界里，也不是不可以，可以按照每一个 business unit 的特点来整理出大家所关注的几个方面。 设计完 aspect 之后，就可以来判断哪些 review 里涉及到了某个 aspect 了，这个问题本质上其实就是分类问题。如果提前有标注好的数据的话，就可以训练一个多分类模型。 这样的

好处是可以处理一些特殊情况比如 “这个好贵啊 “， 在这句话里其实没有出现任何 noun phrase, 所以通过之前的方法是找不出具体 aspect 的，但实际上这句话里所说的其实就是价格。但如果我们使用的是基于分类算法的 aspect 分类，那这个模型本身可以根据” 好贵 “这个关键词有可能判断出这句话说的就是价格方面的 aspect。

在本项目中，我们是针对于每个 business 自动提取出了 aspects, 但这样的缺点是有些 business 评论比较少的话，统计出来的不够准确，所以我们去掉了很多评论数少的 business。实际上，我们也可以把 business 按照种类来划分，然后把相同种类的 aspects 一起提取出来（如果两个 business 的种类是一样的，那提取出来的 aspects 也是一样的）。这样就可以避免数据少的问题。

5.2 Aspect 相关的 Phrase 提取

在 baseline 中我们使用的 aspect 提取采用了匹配的方法，但这种方法必须要精准匹配才可以。一个 aspect 可能有不同的说法比如“价格 “， “价钱 “，其实说的都是一类。一种解决方法是提前整理好所有可能的说法，还有一种方式是在匹配的时候做模糊的匹配，比如使用 word2vec 等技术。

5.3 Top 5 正面或者负面评论提取

在 baseline 里，当我们提取有代表性的 TOP 5 正面或者负面情感时，我们的做法是对于评分值做了个排序，然后返回评分制最高的 TOP5 和最低的 TOP5。但这种做法有什么问题呢？

我们需要考虑一个重要的问题: 多样性！ 比如针对于某一个 aspect 的 TOP 5 评论分别是： “课程质量非常好！ “，“课程质量很好！”， “课程质量很棒！”，其实对于用户来说没有太多价值的，因为都是一样的评论，虽然说得都是非常正面。 那对于这个问题，理想的情况下是需要返回具有多样性的评论，也就是互相之间没有太大的类似性。 那这个多样性又如何保证呢？

对于这个问题，其实可以使用相似度度量的方法。 比如根据预测出来的正面评分对评论排好了序，r1, r2, r3, r4, r5….. 假如我们选择了 r1，接下来我们需要判断是否 r2 跟 r1 之间有很强的相关性，如果没有就选择 r2，这时候集合里有[r1, r2]了， 接下来再判断一下是否 r3 跟 r1, r2 有很强的相关性，如果跟任何一个有很强的相关性，即可以 pass 掉，接着 move on 到 r4， 以此类推。

另外，对于多样性也有一些系统化的研究比如类似于加入一些正则项来解决多样性，经典的技术就是：Determinantal point process (https://arxiv.org/pdf/1207.6083.pdf
)

5.4 情感分类器的训练

我们在搭建情感分类器的时候，使用了 review 里自带的 star 的字段。但直接使用这些字段是有一些问题的。比如有些人就很苛刻，一般不会给很高的分数，但有些人可能比较 nice 一些，基本给出很高的分数。这就意味着每一个人的 scaling 是不一样的。比如我觉得一个商品非常好才会给 3 分，但有些人觉得还不错就直接给到 4 分。

为了解决这个问题，一种简单的方法就是考虑这个用户的 average rating，然后通过 average rating 来纠正一下它的评价值。

6. 作业的提交

本次作业跟**以往的不一样，具有一定的开放性**，所以不设定条条框框，当作一个小的开放式项目来对待。最低标准是完成本报告里所指出的 baseline。

贪心学院自然语言处理

作业需要提交两部分内容：

- **代码部分**：所有的代码需要放在 submit 文件夹下面，另外 submit 文件夹下面有 main.py 文件。main.py 里编写几个样例即可以，比如 business_ids = [xxx, xxx, xxx]，然后依次输出这几个 business 的评论的 summary。格式尽量弄得好看一些。提交前务必要测试命令：python main.py，因为我们检查作业的时候，直接通过 python main.py 来测试。如果需要加入一些参数，请注明每个参数的定义，或者编写一个 help 文档，可以在命令行里看到参数的定义以及使用方法。另外，如果环境依赖于 anaconda, nltk 之外的其他的 library，请在 requirements.txt 里面注明。代码里给了几个文件（business.py, sentence.py），可以参考，但具体代码结构自行安排即可。

- **报告部分**：写一份 report 放到知乎上，把链接放到 zhihu_link.txt 里。具体报告里需要包含：
    - 问题的背景以及描述
    - 使用的方法
    - 除了 baseline 方法，还尝试了哪些方法？
    - 结果展示
    - 目前方法论的不足以及之后改进思路

- **数据部分**：不要上传任何的数据。但程序里数据的默认地址是 data/，解压后的所有的数据全部放到 data 文件夹的根目录下即可。但上传作业的时候不要上传。

评分标准：

- 报告：50%
    - 问题是否描述清楚（10%）
    - 使用的方法（20%）

  - 尝试除了 baseline 之外的方法（10%）
  - 结果分析以及改进思路（10%）
- 代码：50%
  - 准确性（30%）
  - 结构以及清晰读 （20%）

参考：

https://github.com/jiangqn/Aspect-Based-Sentiment-Analysis