

如何用Python做情感分析?



作者 王树义 (/u/7618ab4a30e4) [+ 关注](#)

2017.06.14 15:18* 字数 2639 阅读 3383 评论 3 喜欢 118 赞赏 1

(/u/7618ab4a30e4)

商品评论挖掘、电影推荐、股市预测.....情感分析大有用武之地。本文帮助你一步步用Python做出自己的情感分析结果，难道你不想试试看?



需求

如果你关注数据科学研究或是商业实践，“情感分析”(sentiment analysis)这个词你应该不陌生吧?

维基百科上，情感分析的定义是：

文本情感分析（也称为意见挖掘）是指用自然语言处理、文本挖掘以及计算机语言学等方法来识别和提取原素材中的主观信息。


听着很高大上，是吧？如果说得具体一点呢？

给你一段文本，你就可以用情感分析的自动化方法获得这一段内容里包含的情感色彩是什么。

神奇吧？



情感分析不是炫技工具。它是闷声发大财的方法。早在2010年，就有学者指出，可以依靠Twitter公开信息的情感分析来预测股市的涨落，准确率高达87.6%！



Cornell University
Library

We gratefully acknowledge support from
the Simons Foundation
and member institutions

[arXiv.org](#) > [cs](#) > [arXiv:1010.3003](#)

在这些学者看来，一旦你能够获得大量实时社交媒体文本数据，且利用情感分析的黑魔法，你就获得了一颗预测近期投资市场趋势的水晶球。



这种用数据科学碾压竞争者的感受，是不是妙不可言啊？

大数据时代，我们可以获得的文本数据实在太多了。仅仅是大众点评、豆瓣和亚马逊海量的评论信息就足够我们挥锹抡镐，深挖一通了。

你是不是疑惑，这么高深的技术，自己这个非计算机专业的文科生，如何才能应用呢？

不必担心。从前情感分析还只是实验室或者大公司的独门秘籍。现在早已飞入寻常百姓家。门槛的降低使得我们普通人也可以用Python的几行代码，完成大量文本的情感分析处理。

是不是摩拳擦掌，打算动手尝试了？

那我们就开始吧。

安装

为了更好地使用Python和相关软件包，你需要先安装Anaconda套装。详细的流程步骤请参考《如何用Python做词云 (<http://www.jianshu.com/p/e4b24a734ccc>) 》一文。

到你的系统“终端”(macOS, Linux)或者“命令提示符”(Windows)下，进入我们的工作目录demo，执行以下命令。

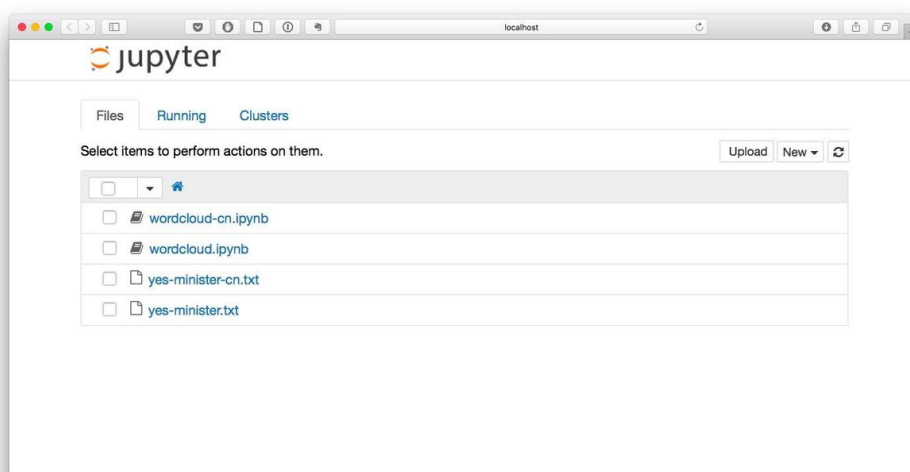
```
pip install snowlp
pip install -U textblob
python -m textblob.download_corpora
```

好了，至此你的情感分析运行环境已经配置完毕。

在终端或者命令提示符下键入：

```
jupyter notebook
```

你会看到目录里之前的那些文件，忽略他们就好。



好了，下面我们就可以愉快地利用Python来编写程序，做文本情感分析了。

英文

我们先来看英文文本的情感分析。

这里我们需要用到的是 TextBlob包 (<https://github.com/sloria/TextBlob>) 。

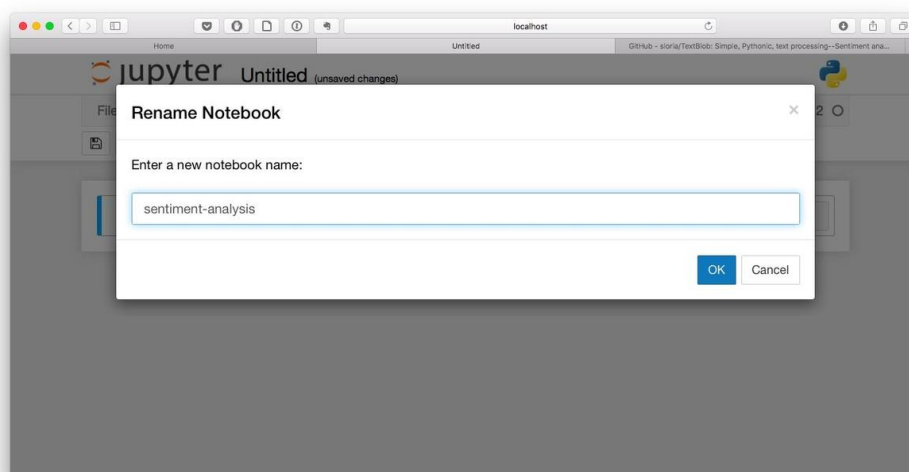


Features

- Noun phrase extraction
- Part-of-speech tagging
- Sentiment analysis
- Classification (Naive Bayes, Decision Tree)
- Language translation and detection powered by Google Translate
- Tokenization (splitting text into words and sentences)
- Word and phrase frequencies
- Parsing
- n-grams
- Word inflection (pluralization and singularization) and lemmatization
- Spelling correction
- Add new models or languages through extensions
- WordNet integration

其实，从上图可以看出，这个包可以做许许多多跟文本处理相关的事情。本文我们只专注于情感分析这一项。其他功能以后有时间我们再介绍。

我们新建一个Python 2笔记本，并且将其命名为“sentiment-analysis”。



先准备一下英文文本数据。

```
text = "I am happy today. I feel sad today."
```

这里我们输入了两句话，把它存入了text这个变量里面。学了十几年英语的你，应该立即分辨出这两句话的情感属性。第一句是“我今天很高兴”，正面；第二句是“我今天很沮丧”，负面。

下面我们看看情感分析工具TextBlob能否正确识别这两句话的情感属性。

首先我们呼唤TextBlob出来。

```
from textblob import TextBlob
blob = TextBlob(text)
blob
```



按Shift+Enter执行，结果好像只是把这两句话原封不动打印了出来而已嘛。

```
In [2]: from textblob import TextBlob
        blob = TextBlob(text)
        blob

Out[2]: TextBlob("I am happy today. I feel sad today.")
```

别着急，TextBlob已经帮我们一段文本分成了不同的句子。我们不妨看看它的划分对不对。

```
blob.sentences
```

执行后输出结果如下：

```
In [3]: blob.sentences

Out[3]: [Sentence("I am happy today."), Sentence("I feel sad today.")]
```

划分无误。可是你能断句有啥了不起?! 我要情感分析结果!

你怎么这么着急啊? 一步步来嘛。好，我们输出第一句的情感分析结果：

```
blob.sentences[0].sentiment
```

执行后，你会看到有意思的结果出现了：

```
In [4]: blob.sentences[0].sentiment

Out[4]: Sentiment(polarity=0.8, subjectivity=1.0)
```

情感极性0.8，主观性1.0。说明一下，情感极性的变化范围是[-1, 1]，-1代表完全负面，1代表完全正面。

既然我说自己“高兴”，那情感分析结果是正面的就对了啊。

趁热打铁，我们看第二句。

```
blob.sentences[1].sentiment
```

执行后结果如下：

```
In [5]: blob.sentences[1].sentiment

Out[5]: Sentiment(polarity=-0.5, subjectivity=1.0)
```

“沮丧”对应的情感极性是负的0.5，没毛病!

更有趣的是，我们还可以让TextBlob综合分析出整段文本的情感。

```
blob.sentiment
```



执行结果是什么?

给你10秒钟, 猜猜看。

不卖关子了, 是这样的:

```
In [6]: blob.sentiment
Out[6]: Sentiment(polarity=0.15000000000000002, subjectivity=1.0)
```

你可能会觉得没有道理。怎么一句“高兴”, 一句“沮丧”, 合并起来最后会得到正向结果呢?

首先不同极性的词, 在数值上是有区别的。我们应该可以找到比“沮丧”更为负面的词汇。而且这也符合逻辑, 谁会这么“天上一脚, 地下一脚”矛盾地描述自己此时的心情呢?

中文

试验了英文文本情感分析, 我们该回归母语了。毕竟, 互联网上我们平时接触最多的文本, 还是中文的。

中文文本分析, 我们使用的是 SnowNLP包 (<https://github.com/isnowfy/snownlp>)。这个包跟TextBlob一样, 也是多才多艺的。

Features

- 中文分词 ([Character-Based Generative Model](#))
- 词性标注 ([TnT 3-gram](#) 隐马)
- 情感分析 (现在训练数据主要是买卖东西时的评价, 所以对其他的一些可能效果不是很好, 待解决)
- 文本分类 ([Naive Bayes](#))
- 转换成拼音 ([Trie树实现的最大匹配](#))
- 繁体转简体 ([Trie树实现的最大匹配](#))
- 提取文本关键词 ([TextRank](#)算法)
- 提取文本摘要 ([TextRank](#)算法)
- tf, idf
- Tokenization (分割成句子)
- 文本相似 ([BM25](#))
- 支持python3 (感谢[erning](#))

我们还是先准备一下文本。这次我们换2个形容词试试看。

```
text = u"我今天很快乐。我今天很愤怒。"
```

注意在引号前面我们加了一个字母u, 它很重要。因为它提示Python, “这一段我们输入的文本编码格式是Unicode, 别搞错了哦”。至于文本编码格式的细节, 有机会我们再详细聊。

好了, 文本有了, 下面我们让SnowNLP来工作吧。

```
from snownlp import SnowNLP
s = SnowNLP(text)
```



我们来看看SnowNLP能不能像TextBlob一样正确划分我们输入的句子，所以我们执行以下输出：

```
for sentence in s.sentences:  
    print(sentence)
```

执行的结果是这样的：

```
In [55]: for sentence in s.sentences:  
        print(sentence)
```

```
我今天很快乐  
我今天很愤怒
```

好的，看来SnowNLP对句子的划分是正确的。

我们来看第一句的情感分析结果吧。

```
s1 = SnowNLP(s.sentences[0])  
s1.sentiments
```

执行后的结果是：

```
In [56]: s1 = SnowNLP(s.sentences[0])  
        s1.sentiments
```

```
Out[56]: 0.9717424426821268
```

看来“快乐”这个关键词真是很能说明问题。基本上得到满分了。

我们来看第二句：

```
s2 = SnowNLP(s.sentences[1])  
s2.sentiments
```

执行结果如下：

```
In [57]: s2 = SnowNLP(s.sentences[1])  
        s2.sentiments
```

```
Out[57]: 0.07762415224442543
```

这里你肯定发现了问题——“愤怒”这个词表达了如此强烈的负面情感，为何得分依然是正的？

这是因为SnowNLP和textblob的计分方法不同。SnowNLP的情感分析取值，表达的是“这句话代表正面情感的概率”。也就是说，对“我今天很愤怒”一句，SnowNLP认为，它表达正面情感的概率很低很低。

这么解释就合理多了。

小结



学会了基本招式，很开心吧？下面你可以自己找一些中英文文本来实践情感分析了。

但是你可能很快就会遇到问题。例如你输入一些明确的负面情绪语句，得到的结果却很正面。

不要以为自己又被忽悠了。我来解释一下问题出在哪儿。

首先，许多语句的情感判定需要上下文和背景知识，因此如果这类信息缺乏，判别正确率就会受到影响。这就是人比机器（至少在目前）更强大的地方。

其次，任何一个情感分析工具，实际上都是被训练出来的。训练时用的是什么文本材料，直接影响到模型的适应性。

例如SnowNLP，它的训练文本就是评论数据。因此，你如果用它来分析中文评论信息，效果应该不错。但是，如果你用它分析其他类型的文本——例如小说、诗歌等，效果就会大打折扣。因为这样的文本数据组合方式，它之前没有见过。

解决办法当然有，就是用其他类型的文本去训练它。见多识广，自然就“见惯不怪”了。至于该如何训练，请和相关软件包的作者联系咨询。

讨论

除了本文提到的文本分析应用领域，你还知道哪些其他的工作可以用情感分析来自动化辅助完成？除TextBlob和SnowNLP外，你还知道哪些开放免费软件包可以帮助我们完成情感分析工作？欢迎留言分享给大家，我们一起交流讨论。

如果你对我的文章感兴趣，欢迎点赞，并且微信关注和置顶我的公众号“玉树芝兰”(nkwangshuyi)。

如果本文可能对你身边的亲友有帮助，也欢迎你把本文通过微博或朋友圈分享给他们。让他们一起参与到我们的讨论中来。

延伸阅读

- 如何用Python做词云? - 简书 (<http://www.jianshu.com/p/e4b24a734ccc>)
- 如何用Python做中文分词? - 简书 (<http://www.jianshu.com/p/721190534061>)
- 如何用Python做舆情时间序列可视化? - 简书 (<http://www.jianshu.com/p/4ea083874df4>)
- 贷还是不贷：如何用Python和机器学习帮你决策? - 简书 (<http://www.jianshu.com/p/67a71e366516>)
- 如何用Python从海量文本抽取主题? - 简书 (<http://www.jianshu.com/p/fdde9fc03f94>)

作者信息

王树义，大学教师，终身学习者。稍微懂一点儿写作、演讲、Python和机器学习。欢迎微信关注并置顶我的公众号“玉树芝兰”(nkwangshuyi)。



玉树芝兰 (/nb/130182)

举报文章 © 著作权归作者所有



王树义 (/u/7618ab4a30e4) ♂

写了 228730 字, 被 9759 人关注, 获得了 2543 个喜欢
(/u/7618ab4a30e4)


+ 关注

终身学习者、大学教师。稍微懂一点儿写作、演讲、Python和机器学习。欢迎微信关注并置顶我的公众号“...


喜欢 (/sign_in?utm_source=desktop&utm_medium=not-signed-in-like-button)

更多分享


被以下专题收入, 发现更多相似内容




@IT·互联网 (/c/V2CqjW?utm_source=desktop&utm_medium=notes-included-collection)




首页投稿 (/c/bDHhpK?utm_source=desktop&utm_medium=notes-included-collection)




爬虫专题 (/c/3e3636c40c41?utm_source=desktop&utm_medium=notes-included-collection)




python (/c/bd023c60a1d7?utm_source=desktop&utm_medium=notes-included-collection)



阅读记录 (/c/a1993f897cc2?utm_source=desktop&utm_medium=notes-included-collection)



工具癖 (/c/2mvgxp?utm_source=desktop&utm_medium=notes-included-collection)



AI (/c/26533158b58c?utm_source=desktop&utm_medium=notes-included-collection)

展开更多