

Python数据预处理从入门到入葬

文本数据预处理可以分为读取数据源、文本分词、文本清洗以及数据可视化展示
接下来会使用代码实现完整的预处理流程

1. 文本分词

Jieba库

Jieba是强大的Python中文分词库，主要功能是做中文分词，可以进行简单分词、并行分词、命令行分词

```
import jieba
```

补充-对比两种分词函数

cut()与lcut()方法

- i) `.cut` 生成的是一个生成器，generator，可以通过for循环来取里面的每一个词，通过print输出得到的是封装的数据
- ii) `.lcut` 直接生成的就是一个list，通过print输出得到的是一个列表

```
# 参数传入 需要分词的字符串 是否采用全模式
seg_list = jieba.cut("中国上海是一座美丽的国际性大都市", cut_all=True)
```

```
list(seg_list)
```

```
['中国', '上海', '是', '一座', '美丽', '的', '国际', '国际性', '大都', '大都市',
'都市']
```

```
# 对比是否采用全模式的区别
seg_list = jieba.cut("中国上海是一座美丽的国际性大都市", cut_all=False)
list(seg_list)
```

```
['中国', '上海', '是', '一座', '美丽', '的', '国际性', '大都市']
```

添加自定义词典

自定义的词汇的方法：`jieba.load_userdict(file_name)` 参数是文本文件，txt、csv都可以。
自定义词典文件的词汇格式是一个词占一行，每一行分三部分：词语、词频（可省略）、词性（可省略），用空格隔开，顺序不可颠倒

```
1 蒂法
2 最终幻想7
3 爱丽丝
4 米德加
5 萨菲罗斯
6 神罗战士
7 第五街区
8 赛特拉
9 约定之地
10 反神羅组织
11 魔晄
12 破坏剑
13 尼布尔海姆
14 杰诺瓦
15 重制版
16 扎克斯
```

```
# 载入词典
jieba.load_userdict("自定义分词.txt")
seg_list = jieba.cut("游戏最终幻想7，男主角为克劳德，游戏是双女主，分别是蒂法和爱丽丝。")
"/ ".join(seg_list)
```

```
'游戏/ 最终幻想7/ ， / 男主角/ 为/ 克劳德/ ， / 游戏/ 是/ 双/ 女主/ ， / 分别/ 是/
蒂法/ 和/ 爱丽丝/ 。 '
```

修改jieba的正则表达式

由于jieba在分词时会把它们空格左右的词语进行分词
这种方式有时并不符合我们的需求。我们可以通过改jieba包init.py中几个正则表达式来解决这个问题

```
import re

jieba.re_han_default = re.compile("([\u4E00-\u9FD5a-zA-Z0-9+&_.% -]+)", re.U)
```

2. 词频统计

任务1. 从本地word文档爬取正文内容存储为Python中的字符串

任务2. 进行文本分词

任务3. 将分词后的词段进行词频统计

docx库

使用Document(docx文件文件名)来读取word文档。调用.paragraphs()方法可以读取每一段内容，但该内容是封装的，还需要用.text()方法才能显示出文本内容

```

from docx import Document

docx = Document("FF7补完计划.docx")
# 将docx文档中每个段落的内容拼接到字符串中
contents = "".join(para.text for para in docx.paragraphs)
contents

```

[7]: '最终幻想7补完计划（设定集+编年史）历史篇正传发生的2000年前，星球上存在古代人类[塞特拉]，是最早一批可以使用来自星球神圣力量的原住民。有一天从天而降名为[杰诺瓦]的灾祸掉落在星球的最北边，巨大的冲击在北方的雪山上形成一个巨大的空洞。其中，杰诺瓦是类似寄生虫一样的寄生生物，有以下几种特点：可以操控宿主意识，从而改变宿主记忆。拥有可怕的恢复能力，可以很大程度保证宿主的生命不会受到攻击，而轻易遇到生命的威胁。可以利用“拟态能力”散播病毒，即可以通过同类的生命体进行传染。它会依照本能毁灭自己以外的事物，攻击力高，从不考虑破坏带来的结果。杰诺瓦最终会侵蚀整颗星球使星球生病，最终迎来毁灭。此后它便可以再次踏上星际旅行，以同样的方式去祸害下一颗有生命的星球。因此古人类因为杰诺瓦差点遭受灭顶之灾，然而幸运的是最后幸存的人类从这场惨烈的战斗中获得了胜利，他们将被杰诺瓦寄生的女性宿主封印在了深深的地底下，强制令杰诺瓦进入了永恒的长眠。³³ 33年前，时任[神羅]公司技术研发部门的部长[加斯特]博士通过挖掘的方式，成功找到了已变成木乃伊的杰诺瓦死骸，并误认为这就是已灭绝的古代种族人类塞特拉，随机便草率的基于这一观点启动了名为[GENOVA PROJECT(杰诺瓦计划)]的人体实验企划案。随后，依照这个计划，加斯特将死骸杰诺瓦的细胞成功提取，并植入进怀有胎儿的人体内进行实验，希望得到古代种族人类的能力，所以这个恐怖的人体实验便拉开了一切悲剧事件开端的序幕。当时还是普通研究员的[宝条]成功与同事兼恋人的[卢克蕾姿娅]发生“关系”，随机冷酷的宝条将卢克蕾姿娅当成了实验标的，将杰诺瓦细胞植入了她已怀有身孕的体内，从而使得这个携带有杰诺瓦细胞的生命体诞生了，他就是萨菲罗斯（拉丁语意为“神性的流出”）。30年前，加斯特终于意识到自己对杰诺瓦的认识是错误的，开始懊悔自己开展了非法且不人道主义的人体实验。愧疚之下，他辞掉了自己的职务，在星球的北边，也就是当时挖掘出杰诺瓦死骸的冰雪附近安顿下来，开始独自对杰诺瓦以及古代种的课题继续开展研究。之后他与当地真正的古代种族后裔[依法露娜]相识，并从她口中了解到真正的古代种历史，包括杰诺瓦的拟态能力以及攻击好战等特性，而这些都是神羅不知道的情报。随着时间的推移，加斯特与依法露娜相爱，两人组建了家庭并生育了一个女儿，这个孩子就是FF7故事中唯一的古代种族后裔爱丽丝，但加斯特最终被成功上位科研部长的宝条博士派人杀害，而加斯特关于杰诺瓦的所有研究成果，最终全部落入到宝条手中。¹⁰年前 -> ff7：危机之前 BC8年前 -> ff7：核心危机 C3年前 -> ff72年后 -> ff7：圣子降临 AC3年后 -> ff7：地狱大饥荒 DC.....最终幻想7的所有故事，是[真红XIII]的回忆录，他被宝条所改造，导致活了整整500年都没有老死（本身作为星之守护族的寿命很长，宝条对其进行的身体改造，编号为13）。根据他讲给后代的故事，在很久很久以前，事情是这样子开始的——神羅篇补充：故事的主舞台发生在[米德加]，取自《北欧神话》九界当中的人间。在这个无比巨大的金属圆盘的正中心，正是位于市中心区域的这家富可敌国的商业组织[神羅]公司，所以米德加也被民众称为神羅城。神羅的成立之初，是给各国建造兵器，一家兵工厂+重工业企业，因而顺理成章的拥有战斗和战争的企业文化，所以单凭贩卖武器这一业务就为神羅创造了源源不断的庞大现金流，直接使得这家公司在极短时间内实现了盈利。拥有钞能力的神羅，开始大量投资基础设施建设领域的诸多项目。比如厂房建设、修建铁路、或者报纸纂组纸钱类的轻工副业等等。这些业务相比较贩售武器（“现金生业务”）而言，现金流入速度与盈利能力偏弱，故它们均属于

进行分词并储存为列表形式

```

seg_list = jieba.lcut(contents)

```

通过strip方法将字符串中的转义字符和空格符剔除

```

words = []
for word in seg_list:
    if word.strip() != "":
        words.append(word.strip())
words

```

[9]: ['最终幻想7', '补完', '计划', '（', '设定', '集', '‘’, '编年史', '）', '历史', '篇', '正传', '发生', '的', '2000', '年前', '‘’, '星球']

collections库

使用collections库中的Counter()模块，调用most_common()实现词频统计

```

from collections import Counter

top_10 = Counter(words).most_common(10) # 筛选分词后词频前10名的数据
print(*top_10)

```

('的', 1078) ('，', 1035) ('。', 500) ('了', 415) ('在', 169) ('扎克斯', 165)
(‘是’, 139) ('他', 137) ('神羅', 112) ('[', 103)

不难发现，这些分词大部分都是我们不希望得到的词语。所以还需要做一步停用词的过滤

3. 停用词过滤

下载停用词库

停用词一般储存在txt文件中，每一行代表一个停用词

首先在网上下载了中文停用词库、四川大学机器智能实验室停用词库、哈工大停用词表、百度停用词表，并把它们整理为一个新的txt文件

```
14 “  
15 ”  
16 、  
17 。  
18 《  
19 》  
20 一  
21 一些  
22 一何  
23 一切  
24 一则  
25 一方面  
26 一旦  
27 一来  
28 一样  
29 一般  
30 一转眼  
31 万一  
32 上  
33 上下  
34 下
```

考虑到这四个停用词表可能会有重复的停用词

因此还需要定义一个集合用来去重

```
# 加载停用词库  
with open("停用词库.txt", encoding="utf-8") as f:  
    contents = f.readlines()  
    stop_words = set()  
    for word in contents:  
        word = word.replace("\n", "") # 去掉读取每一行数据的换行符  
        stop_words.add(word)
```

```
# 读取完停用词库后，去除停用词
```

```
data = []  
jieba.load_userdict("自定义分词.txt")  
for word in words:  
    for word in jieba.lcut(word):  
        if word not in stop_words:  
            data.append(word)  
data
```

```
[12]: ['最终幻想7',  
       '补完',  
       '计划',  
       '设定',  
       '集',  
       '编年史',  
       '历史',  
       '篇',  
       '正传',  
       '发生',  
       '2000',  
       '年前',  
       '星球',  
       '古代种',  
       '人类',  
       '赛特拉',  
       '最早',  
       '一批']
```

再次查看排名前60的词条

```
top_60 = Counter(data).most_common(60)  
top_60
```

```
('身体', 24),  
('士兵', 24),  
('神羅社长', 24),  
('中', 23),  
('宝条', 23),  
('埃尔夫', 23),  
('二人', 22),  
('弗西托', 22),  
('找到', 21),  
('魔晄炉', 21),  
('人类', 20),  
('G', 20),  
('告诉', 20),  
('加斯特', 19),  
('1st', 19),  
('拥有', 18),  
('执行', 18),  
('梦想', 18),
```

添加自定义停用词

依旧有一些我们不想要的词条，需要进行二次清洗
即定义一个自定义停用词库，把那些无效的词条放进去

```
16 最终  
17 便  
18 回到  
19 发生  
20 S  
21 身体  
22 部门  
23 中  
24 里  
25 年  
26 ...  
27 ...  
28 均  
29 没  
30 没想到  
31 年前  
32 新  
33 方式
```

```
with open("自定义停用词.txt", encoding="utf-8") as f:  
    contents = f.readlines()  
    stop_words = set()  
    for content in contents:  
        content = content.replace("\n", "")  
        stop_words.add(content)
```

```

filtered_data = []
for word in data:
    if word not in stop_words:
        filtered_data.append(word)
# 查看排名前60的词条
top_60 = Counter(filtered_data).most_common(60)
top_60

```

```
[14]: [('扎克斯', 165),
 ('神羅', 112),
 ('萨菲罗斯', 84),
 ('杰内西斯', 82),
 ('塔克斯', 74),
 ('安吉尔', 73),
 ('杰诺瓦', 66),
 ('克劳德', 63),
 ('特种兵', 54),
 ('雪崩', 53),
 ('细胞', 46),
 ('维尔多', 41),
 ('魔晄', 39),
 ('爱丽丝', 35),
 ('杰诺瓦细胞', 33),
 ('魔石', 33),
 ('公司', 32),
 ('赫兰迪', 28)]
```

完成数据清理后，就可以实现文本数据的可视化了。这里使用的第三方库是Pyecharts
具体的模块和方法参考官网：

<https://gallery.pyecharts.org/#/README>
<https://pyecharts.org/#/zh-cn/intro>

4. 绘制词频柱状图

制作一个柱状图查看词频排名前30的词条

Pyecharts库

需要用到Pyecharts库中的bar模块

options模块用于修改全局配置

要想在Jupyter_lab中显示出图表，还需要在顶部声明 Notebook 类型，必须在引入
pyecharts.charts 等模块前声明

```

from pyecharts.globals import CurrentConfig, NotebookType
CurrentConfig.NOTEBOOK_TYPE = NotebookType.JUPYTER_LAB

```

```

from pyecharts import options as opts
from pyecharts.charts import Bar
from pyecharts.globals import ThemeType

# 定义两个列表分别代表x轴和y轴数据
x_axis, y_axis = [], []
# 只展示排名前30的分词
count = 1
for part_tuple in top_60:
    if count == 30:
        break
    x_axis.append(part_tuple[0])

```

```

y_axis.append(part_tuple[1])
count += 1

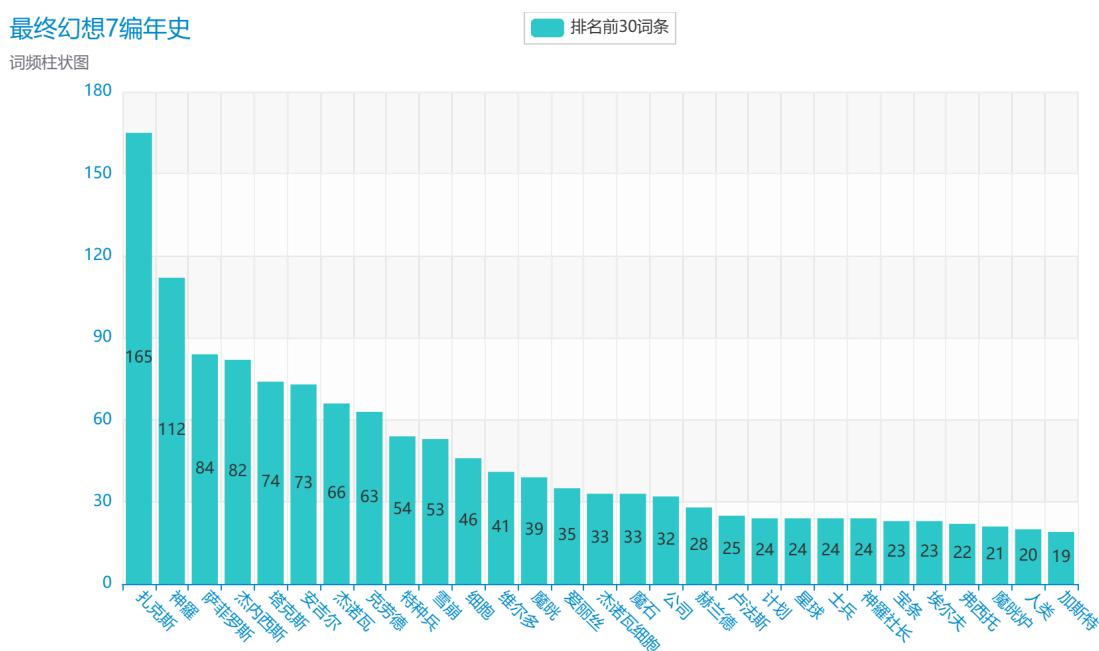
bar = (
    Bar(init_opts=opts.InitOpts(theme=ThemeType.MACARONS))
    .add_xaxis(x_axis)
    .add_yaxis("排名前30词条", y_axis)
    .set_global_opts(
        # 防止x轴数据名太长，使其向下旋转45度
        xaxis_opts=opts.AxisOpts(axislabel_opts=opts.LabelOpts(rotate=-45)),
        title_opts=opts.TitleOpts(title="最终幻想7编年史", subtitle="词频柱状图"),
    )
)
# 在第一次渲染时加载javascript文件
bar.load_javascript()

```

```
<pyecharts.render.display.Javascript at 0x1ee11d2f6d0>
```

注意：在.load_javascript()和.render_notebook两种方法要在不同的cell中调用，否则图片还是不会显示出来。

```
bar.render_notebook()
```



```
# 保存到本地
```

```
bar.render("词频图.html")
```

5. 绘制词云图

需要用到pyecharts中的wordcloud模块

```
cloud.load_javascript()
```

```
<pyecharts.render.display.Javascript at 0x1ee17bc9120>
```

```
cloud.render_notebook()
```

最终幻想7编年史-词云图



```
# 保存到本地  
cloud.render("词云图.html")
```

完成!