**Step1维基百科中文语料的处理**

1使用WikiExtractor提取维基百科语料

WikiExtractor链接：[https://github.com/attardi/wikiextractor](https://github.com/attardi/wikiextractor" \t "_blank)

* 打开命令行，输入

git clone <https://github.com/attardi/wikiextractor> wikiextractor

cd wikiextractor

python setup.py install

* 将语料的压缩文件拷贝进wikiextractor文件夹里面，接下来就是核心命令：

打开命令行：

cd wikiextractor

python WikiExtractor.py -b 1024M -o extracted zhwiki-20200201-pages-articles-multistream.xml.bz2

* 会生成一个extracted的文件夹，里面有个AA文件，内有提取后的2个文件（wiki\_00.txt ，wiki\_01.txt）

2 将两个wiki的数据集合并

用合并代码 merge.py 进行合并, 下面红色的是输入文件的地址，蓝色的是输出文件的地址python merge.py F:\NLP-Project\extracted\wiki\_00 F:\NLP-Project\extracted\wiki\_01 F:\NLP-Project\extracted\wiki\_all

其中F:\NLP-Project\extracted\wiki\_all是最终的wiki的提取完语料的结果

3 wiki语料需进行繁体转简体

* 安装opencc

pip install opencc-python-reimplemented

* 简繁体可利用OpenCC进行繁体转简体的操作，这里已经写好了一份python版本的脚本来进行处理 chinese\_t2s

<https://github.com/bamtercelboo/corpus_process_script/tree/master/chinese_t2s>

* 使用： python chinese\_t2s.py –input input\_file –output output\_file
* 打开命令行，在命令行进到chinese\_t2s.py所在的文件夹下

在命令行里面输入

python chinese\_t2s.py --F:\NLP-Project\extracted\wiki\_all --output F:\NLP-Project\wiki-simplified\wiki\_all

4清洗非中文字符

* 这里已经写好了一份脚本

<https://github.com/bamtercelboo/corpus_process_script/tree/master/clean>

使用： python clean\_corpus.py –input input\_file –output output\_file

* 在命令行进到clean\_corpus.py所在的文件夹下，在命令框中输入

python clean\_corpus.py --input F:\NLP-Project\wiki-simplified\wiki\_all --output F:\NLP-Project\wiki-simplified\wiki\_all\_cleaned

5 对清理过的wiki文件去停用词和分词

* 可以从这里下载<https://github.com/goto456/stopwords> 停用词库

pip install jieba

* 之后用python运行我额外写的去停用词以及分词的代码fengci.py

python fengci.py F:\NLP-Project\wiki-simplified\wiki\_all\_cleaned F:\NLP-Project\wiki-simplified\wiki\_all\_fengci

6 Gensim词向量训练

* 用gensim训练对应的词向量

pip install gensim

word\_vector\_training.py是我编的，下面红色的是输入文件的地址，蓝色的是输出的词向量模型的地址

python word\_vector\_training.py F:\NLP-Project\wiki-simplified\wiki \_all\_fengci F:\NLP-Project\wiki-simplified\wiki\_wordvector.model

7 查询最近似的词

运行word\_vector\_use.py之后他会提示输入要查询的词，然后会自动返回最接近的10个词，以及对应的相似度，下面红色的是输入词向量模型的地址

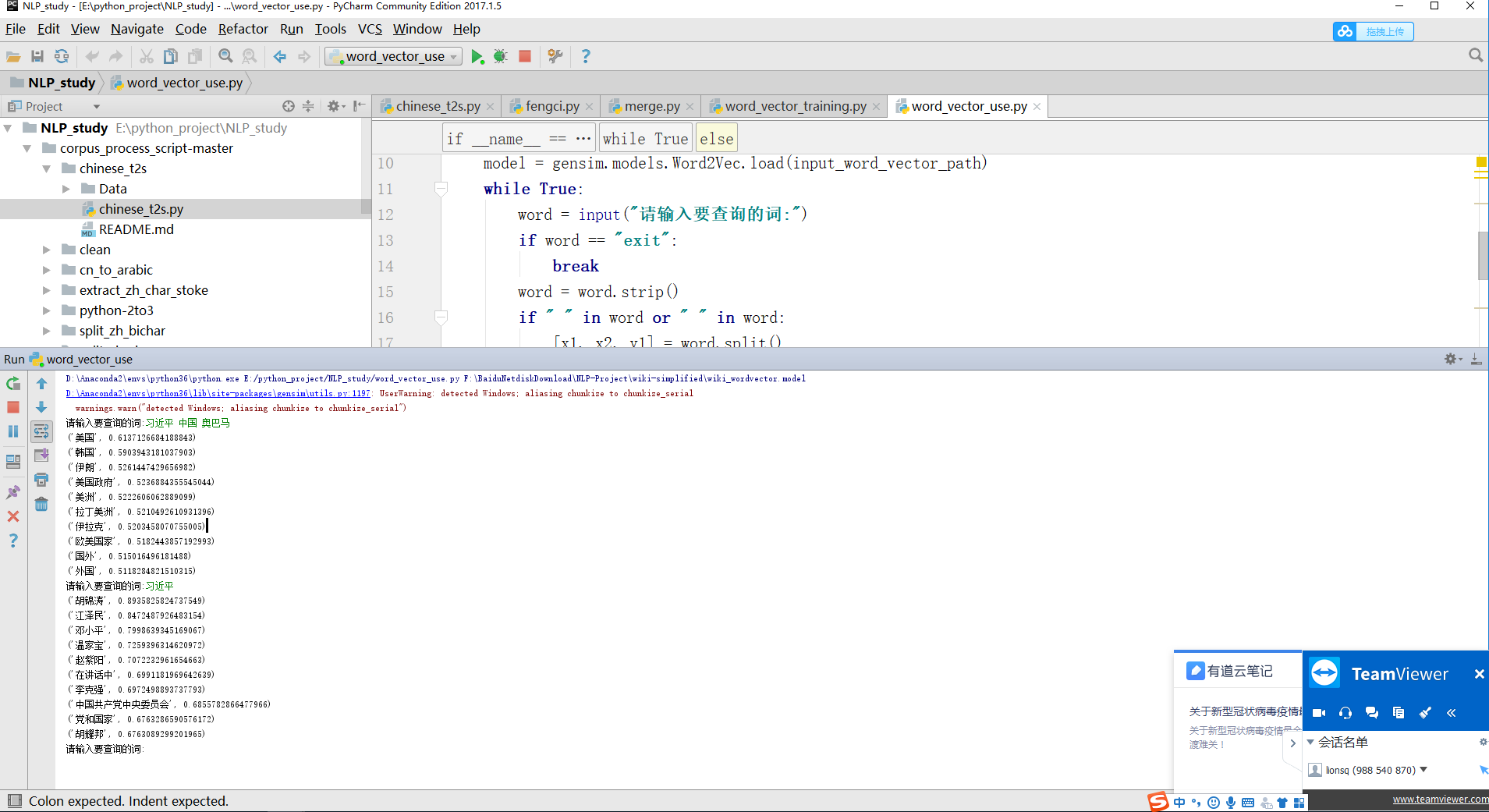
另外一种模式是输入三个单词，以英文的空格间隔，就会输出analogy

输入exit，就退出了

运行

python word\_vector\_use.py F:\NLP-Project\wiki-simplified\wiki\_wordvector.model

运行结果如下图



8 画tsne的图

tsne\_picture.py是我的代码，红色部分是训练好的词向量模型，蓝色部分是一个图要显示多少个单词。

我这里每运行一次都是随机选200个单词的。

运行

python tsne\_picture.py F:\NLP-Project\wiki-simplified\wiki\_wordvector.model 200

效果如下图所示

A close up of text on a white background

Description automatically generated

**Step2处理中文新闻数据集**

新闻数据集比较小，更快一些

从新闻数据集中提取数据

又实验了一遍编码，发现gb18030是可以的

但是还是要先用我的代码清洗新闻数据集

python process\_xinwen.py F:\NLP-Project\xinwen\sqlResult\_1558435.csv F:\NLP-Project\xinwen\content2.txt

之后才能调用clean\_corpus.py再清洗下数据集

python clean\_corpus.py --input F:\NLP-Project\xinwen\content2.txt --output F:\NLP-Project\xinwen\content\_clean.txt

调用jieba进行分词

python fengci.py F:\NLP-Project\xinwen\content\_clean.txt F:\NLP-Project\xinwen\xinwen\_fengci.txt

**Step3**词向量训练及可视化

可以将xinwen\_fengci.txt和之前的wiki\_fengci\_all合并生成一个新文件，之后按照之前类似的训练词向量的方法训练出词模型，以及画图。

python merge.py F:\NLP-Project\wiki-simplified\wiki\_fengci\_all F:\NLP-Project\xinwen\xinwen\_fengci.txt F:\NLP-Project\xinwen\wiki\_xinwen\_fengci.txt

**合并成一个文件之后可以类似的方法来训练了，需要再跑下面几个步骤**

python word\_vector\_training.py F:\NLP-Project\xinwen\wiki\_xinwen\_fengci.txt F:\NLP-Project\wiki-simplified\wiki\_xinwen\_wordvector.model

**看词的相似性和近义词之类的**

python word\_vector\_use.py F:\NLP-Project\wiki-simplified\wiki\_xinwen\_wordvector.model

**重新画tsne的图**

python tsne\_picture.py F:\NLP-Project\wiki-simplified\wiki\_xinwen\_wordvector.model 200