Gensim_Fasttext_practice

2019 年 **12** 月 **10** 日

---

1 Gensim

Gensim 即 **"Generate similarity"**，支持 **LDA** 和 **LSI** 算法，比起其他的工具包 **(**例如 **scikit**、**R** 等**)** 有更多便利的功能。**Gensim** 用于文本处理，与诸如 **Word2Vec**、**FastText** 等词向量模型共同建立主题模型。**Gensim** 有个显著的好处：处理大文本文件时不需要将整个文件加载到内存。

1.1 应用

Harry Potter

使用 **gensim** 处理哈利波特 **1-7**

[98]:
```
# imports needed and set up logging
import gensim
import logging
import os

from gensim.parsing.preprocessing import STOPWORDS
from gensim.parsing.preprocessing import remove_stopwords

logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s',␣
 ↪level=logging.INFO)
```

[99]:
```
filePath = './harry/'
data_file = []
for i,j,k in os.walk(filePath):
    data_file = k
```

[ ]:
```
def read_input(input_file):
    """This method reads the input file which is in gzip format"""

    logging.info("reading file {0}...this may take a while".format(input_file))
    for filename in input_file:
        with open (filePath + filename) as f:
            for i, line in enumerate (f):
                if (i%1000==0):
                    logging.info ("read {0} line".format (i))
                yield gensim.utils.simple_preprocess (remove_stopwords(line))

documents = list (read_input (data_file))
logging.info ("Done reading data file")
```

使用 **Word2Vec** 模型训练

[ ]:
```
model = gensim.models.Word2Vec (documents, size=150, window=10, min_count=2,␣
↪workers=10)
model.train(documents,total_examples=len(documents),epochs=10)
```

[102]:
```
# 测试与”格兰芬多“相似的词语
w1 = "gryffindor"
model.wv.most_similar (positive=w1)
```

```
2019-12-10 10:06:44,062 : INFO : precomputing L2-norms of word weight vectors
```

[102]:
```
[('ravenclaw', 0.8368150591850281),
 ('hufflepuff', 0.8280871510505676),
 ('slytherin', 0.7947815656661987),
 ('points', 0.7544464468955994),
 ('penalty', 0.747697114944458),
 ('goal', 0.7353691458702087),
 ('team', 0.7298713326454163),
 ('cheering', 0.6956561207771301),
 ('tower', 0.6929386258125305),
 ('chaser', 0.692872941493988)]
```

[106]:
```
w1 = ["ron",'harry', 'hermione']
w2 = ["malfoy"]
model.wv.most_similar (positive=w1)
```

[106]:
```
[('furiously', 0.6423398852348328),
 ('both', 0.6290283203125),
 ('nervously', 0.6139020323753357),
 ('awkwardly', 0.6138213872909546),
 ('groggily', 0.6112465262413025),
 ('griphook', 0.5974970459938049),
 ('cho', 0.5930269360542297),
 ('ginny', 0.5894877910614014),
 ('lavender', 0.587001621723175),
 ('eagerly', 0.5856419205665588)]
```

[104]:
```
w1 = "dumbledore"
model.wv.most_similar (positive=w1)
```

[104]:
```
[('quirrell', 0.6694686412811279),
 ('slughorn', 0.6288319826126099),
 ('scrimgeour', 0.6079066395759583),
 ('fudge', 0.6000068187713623),
 ('dippet', 0.599680483341217),
 ('aberforth', 0.5885795950889587),
 ('headmaster', 0.5841313004493713),
 ('karkaroff', 0.5769506692886353),
 ('trelawney', 0.5737648010253906),
 ('kettleburn', 0.5608559250831604)]
```

[111]:
```
# w1 = "azkaban"
w1 = "black"
model.wv.most_similar (positive=w1)
```

[111]:
```
[('auburn', 0.5743957757949829),
 ('beard', 0.5336969494819641),
 ('lupin', 0.5283984541893005),
 ('sightless', 0.5270783305168152),
 ('shard', 0.5236501097679138),
```

```
 ('slighter', 0.5217884182929993),
 ('inherit', 0.5199994444847107),
 ('fawkes', 0.5195986032485962),
 ('silver', 0.5015254020690918),
 ('strand', 0.49161896109580994)]
```

1.2 Gensim 文件结构分析

1.3 参考文献

Gensim Word2Vec Tutorial – Full Working Example https://kavita-ganesan.com/gensim-word2vec-tutorial-starter-code/#.Xe5O-5MzY1J

Gensim Tutorial – A Complete Beginners Guide https://www.machinelearningplus.com/nlp/gensim-tutorial/

Gensim 官方指南 https://radimrehurek.com/gensim/auto_examples/

FastText 官网 https://fasttext.cc/

使用 **Gensim** 实现 **Word2Vec** 和 **FastText** 词嵌入 **https://zhuanlan.zhihu.com/p/59860985**

---

2 FastText

> **FastText** 的优点是开源、免费、轻量级，适用于文本分类和文本向量化表示场景。**fasttext** 只有 **1** 层神经网络，属于所谓的 **shallow learning**，但是 **fasttext** 的效果并不差，而且具备学习和预测速度快的优势，在工业界这点非常重要。它比一般的神经网络模型的精确度还要高。

2.0.1 监督学习

[142]:
```
import fasttext
import numpy as np


input_file = "fasttext/cooking.train"
test_file = "fasttext/cooking.valid"
```

[143]:
```
model = fasttext.train_supervised(input=input_file, lr=1.0, epoch=25,␣
↪wordNgrams=2)
```

[145]:
```
model.save_model("model_cooking.bin")
```

[146]:
```
model.predict("Which baking dish is best to bake a banana bread ?")
((u'__label__baking',), np.array([0.15613931]))
```

[146]:
```
(('__label__baking',), array([0.15613931]))
```

[147]:
```
model.predict("Why not put knives in the dishwasher?")
((u'__label__food-safety',), np.array([0.08686075]))
```

[147]:
```
(('__label__food-safety',), array([0.08686075]))
```

[148]:
```
# 输出:  样本数 查准率 召回率
model.test(test_file)
```

[148]:
```
(3000, 0.5603333333333333, 0.24232377108260056)
```

[149]:
```
model.test(test_file, k=5)
```

[149]:
```
(3000, 0.23466666666666666, 0.5074239584834943)
```

2.1 提升性能的方法

预处理

-epoch 设置迭代次数

-lr 设置学习速率

-wordNgrams 设置 **n-grams**

2.1.1 无监督学习

[151]:
```
# Skipgram model :
skip_model = fasttext.train_unsupervised(input_file, model='skipgram')

# or, cbow model :
cbow_model = fasttext.train_unsupervised(input_file, model='cbow')
```

[169]:
```
# print(skip_model.words)
print(skip_model['wrapped'])
# print(cbow_model.words)
print(cbow_model['wrapped'])
```

```
[-0.19465187 -0.04479896  0.10065624 -0.09421343 -0.02800996  0.21732694
  0.0077522  -0.2412107   0.09359407  0.17727724  0.05896186 -0.18488593
  0.01513194  0.15166113 -0.20329717 -0.0552001   0.42683694  0.00910661
  0.00961484 -0.44683605  0.04435758  0.21829973  0.17575434 -0.14606005
 -0.10665243  0.08387621 -0.03283828 -0.02797444 -0.12167862  0.17336623
 -0.17679827  0.11651226 -0.18751463  0.00174165 -0.41835696 -0.17673816
  0.08096708 -0.20303598 -0.31645945  0.06861698  0.21559025  0.1396372
  0.14146456  0.16863847  0.07490897  0.0116991   0.16614494  0.2089593
 -0.18916611 -0.01394802 -0.21171027  0.06075156  0.0399274   0.43547487
 -0.05727803  0.02331961 -0.11587591  0.0668207   0.2183619  -0.07176779
 -0.04831386  0.31906065  0.10926004  0.04284213  0.13000582  0.04012791
 -0.09185615  0.35881975 -0.27305296 -0.14017381 -0.153338   -0.03606869
  0.3065175  -0.33615887 -0.3371137  -0.1051529   0.13978297  0.226356
  0.06097033  0.10392857 -0.09528024  0.29116046  0.09591603  0.14119829
 -0.12853931 -0.01588185 -0.16494992  0.05728934  0.3184535   0.26291373
  0.35050708 -0.07680756  0.27723882 -0.1907171  -0.02547274 -0.13348278
 -0.13915177  0.09102795 -0.20560063 -0.06583065]
[-0.33493784 -0.10418532  0.19288555 -0.01614315  0.10011041  0.23471981
  0.03406423 -0.24404953  0.08595848  0.22848254  0.003607   -0.17094012
  0.08284683 -0.00463706 -0.14586852  0.00105621  0.43698174  0.05389548
 -0.07238291 -0.5083192   0.00412114  0.39209005  0.29678175 -0.04833782
 -0.24719977  0.20320424 -0.0696809   0.01187763 -0.02672263  0.00447042
 -0.14104913  0.16982497 -0.16750747 -0.06036973 -0.4540276  -0.14136617
  0.11084037 -0.38753772 -0.2219694   0.18250047  0.2424789   0.02152487
  0.12207846  0.11912917  0.16401175  0.09720892  0.1903937   0.29806456
 -0.02759703 -0.2997499  -0.28823012  0.0680689   0.20271893  0.45830035
  0.00222017  0.05028942 -0.22025801  0.2284014   0.44375446 -0.18972802
 -0.15594375  0.35257712  0.17183404 -0.01226057  0.17791972  0.0851055
 -0.12890814  0.3763002  -0.46463412 -0.25624838 -0.17769468 -0.19470379
  0.4587091  -0.36549366 -0.28361496 -0.1034703   0.16579568  0.2937373
  0.02732313  0.14524952 -0.07512996  0.41209462 -0.00329787  0.2651151
 -0.27630916  0.18143748 -0.17016496  0.04769963  0.27566594  0.32229716
```

```
 0.3668678   0.02817702  0.1404737  -0.17176524  0.05698291 -0.17603153
-0.19111109  0.00796013 -0.3222592   0.05299968]
```

[170]:
```
skip_model.save_model("skip_model_cooking.bin")
cbow_model.save_model("cbow_model_cooking.bin")
```

[171]:
```
s_model = fasttext.load_model("skip_model_cooking.bin")
c_model = fasttext.load_model("cbow_model_cooking.bin")
```

[172]:
```
print(s_model['wrapped'])
print(c_model['wrapped'])
```

```
[-0.19465187 -0.04479896  0.10065624 -0.09421343 -0.02800996  0.21732694
  0.0077522  -0.2412107   0.09359407  0.17727724  0.05896186 -0.18488593
  0.01513194  0.15166113 -0.20329717 -0.0552001   0.42683694  0.00910661
  0.00961484 -0.44683605  0.04435758  0.21829973  0.17575434 -0.14606005
 -0.10665243  0.08387621 -0.03283828 -0.02797444 -0.12167862  0.17336623
 -0.17679827  0.11651226 -0.18751463  0.00174165 -0.41835696 -0.17673816
  0.08096708 -0.20303598 -0.31645945  0.06861698  0.21559025  0.1396372
  0.14146456  0.16863847  0.07490897  0.0116991   0.16614494  0.2089593
 -0.18916611 -0.01394802 -0.21171027  0.06075156  0.0399274   0.43547487
 -0.05727803  0.02331961 -0.11587591  0.0668207   0.2183619  -0.07176779
 -0.04831386  0.31906065  0.10926004  0.04284213  0.13000582  0.04012791
 -0.09185615  0.35881975 -0.27305296 -0.14017381 -0.153338   -0.03606869
  0.3065175  -0.33615887 -0.3371137  -0.1051529   0.13978297  0.226356
  0.06097033  0.10392857 -0.09528024  0.29116046  0.09591603  0.14119829
 -0.12853931 -0.01588185 -0.16494992  0.05728934  0.3184535   0.26291373
  0.35050708 -0.07680756  0.27723882 -0.1907171  -0.02547274 -0.13348278
 -0.13915177  0.09102795 -0.20560063 -0.06583065]
[-0.33493784 -0.10418532  0.19288555 -0.01614315  0.10011041  0.23471981
  0.03406423 -0.24404953  0.08595848  0.22848254  0.003607   -0.17094012
  0.08284683 -0.00463706 -0.14586852  0.00105621  0.43698174  0.05389548
 -0.07238291 -0.5083192   0.00412114  0.39209005  0.29678175 -0.04833782
 -0.24719977  0.20320424 -0.0696809   0.01187763 -0.02672263  0.00447042
 -0.14104913  0.16982497 -0.16750747 -0.06036973 -0.4540276  -0.14136617
```

```
 0.11084037 -0.38753772 -0.2219694   0.18250047  0.2424789   0.02152487
 0.12207846  0.11912917  0.16401175  0.09720892  0.1903937   0.29806456
-0.02759703 -0.2997499  -0.28823012  0.0680689   0.20271893  0.45830035
 0.00222017  0.05028942 -0.22025801  0.2284014   0.44375446 -0.18972802
-0.15594375  0.35257712  0.17183404 -0.01226057  0.17791972  0.0851055
-0.12890814  0.3763002  -0.46463412 -0.25624838 -0.17769468 -0.19470379
 0.4587091  -0.36549366 -0.28361496 -0.1034703   0.16579568  0.2937373
 0.02732313  0.14524952 -0.07512996  0.41209462 -0.00329787  0.2651151
-0.27630916  0.18143748 -0.17016496  0.04769963  0.27566594  0.32229716
 0.3668678   0.02817702  0.1404737  -0.17176524  0.05698291 -0.17603153
-0.19111109  0.00796013 -0.3222592   0.05299968]
```

2.2 参考文献

Fasttext supervised tutorial https://fasttext.cc/docs/en/supervised-tutorial.html

Fasttext unsupervised tutorial https://pypi.org/project/fasttext/