自然語言處理實戰演練(一)
- 資料愈處理、建立詞向量空間

自然語言資料的預處理

```
import nltk
from nltk import word_tokenize, sent_tokenize
from nltk.corpus import stopwords
from nltk.stem.porter import *
nltk.download('gutenberg')
nltk.download('punkt')
nltk.download('stopwords')

import string

import gensim
from gensim.models.phrases import Phraser, Phrases
from gensim.models.word2vec import Word2Vec

from sklearn.manifold import TSNE

import pandas as pd
from bokeh.io import output_notebook, output_file
from bokeh.plotting import show, figure
%matplotlib inline
```

斷句與分詞（tokenization）

```
from nltk.corpus import gutenberg
```

```
gutenberg.fileids()
```


```
[ 'austen-emma.txt' ,
  'austen-persuasion.txt' ,
  'austen-sense.txt' ,
  'bible-kjv.txt' ,
(中略)
  'shakespeare-hamlet.txt' ,
  'shakespeare-macbeth.txt' ,
  'whitman-leaves.txt' ]
```

各文字檔

```
len(gutenberg.words())
```


```
2621613
```

```
gberg_sent_tokens = sent_tokenize(gutenberg.raw())
```

```
gberg_sent_tokens[0]
```

扉頁、該章標題

```
「[Emma by Jane Austen 1816]\n\nVOLUME I\n\nCHAPTER I\n\n\nEmma
Woodhouse, handsome, clever, and rich, with a comfortable home\nand
happy disposition, seemed to unite some of the best blessings\nof
existence; and had lived nearly twenty-one years in the world\nwith
very little to distress or vex her.」
```

第 0 句

```
gberg_sent_tokens[1]
```


```
「She was the youngest of the two daughters of a most affectionate, \
nindulgent father; and had, in consequence of her sister's marriage,\
nbeen mistress of his house from a very early period.」
```

第 1 句

```
word_tokenize(gberg_sent_tokens[1])
```

輸出

```
['She',
 'was',
 'the',
 'youngest',
 'of',
 'the',
 'two',
 'daughters',
 'of',
 'a',
 'most',
 'affectionate',
 ',',
 'indulgent',
 'father',
 ';',
 'and',
 'had',
 ',',
 'in',
 'consequence',
 'of',
 'her',
 'sister',
 "'",
 's',
 'marriage',
 ',',
 'been',
 'mistress',
 'of',
 'his',
 'house',
 'from',
 'a',
 'very',
 'early',
 'period',
 '.']
```

▲ 第 1 句的分詞結果

```
word_tokenize(gberg_sent_tokens[1])[14]
```


```
'father'
```

```
gberg_sents = gutenberg.sents()
```

```
gberg_sents[0:3]
```


```
[['[', 'Emma', 'by', 'Jane', 'Austen', '1816', ']'],
['VOLUME', 'I'],
['CHAPTER', 'I']]
```

將所有字母轉換為小寫

將艾瑪的第 1 句 (索引 4) 轉為小寫

```
[w.lower() for w in gberg_sents[4]]
```

這裡使用 list 生成式外加 for 迴圈來做處理, 不熟悉的話還請查看參考書目 Ref 1.「**用 Python 學運算思維**」


```
[ 'she' ,   ← 改成小寫
  'was' ,
  'the' ,
  'youngest' ,
  'of' ,
  'the' ,
  'two' ,
  'daughters' ,
(以下略)
```

移除停用詞與標點符號

```
stpwrds = stopwords.words('english') + list(string.punctuation)
```

```
[w.lower() for w in gberg_sents[4] if w.lower() not in stpwrds]
```

利用 list 生成式一口氣去除索引 4 這句的停用詞與標點符號


```
['youngest',
'two',
'daughters',
'affectionate',
'indulgent',
'father',
'consequence',
'sister',
'marriage',
'mistress',
'house',
'early',
'period']
```

詞幹提取（stemming）

▼ 將詞幹提取步驟併入 list 生成式

```
[stemmer.stem(w.lower()) for w in gberg_sents[4] 接下行
    if w.lower() not in stpwrds]
```


```
['youngest',
'two',
'daughter',
'affection',
'indulg',
'father',
'consequ',
'sister',
'marriag',
'mistress',
'hous',
'earli',
'period']
```

處理 n-gram 語法

- 找出 2-gram 語法

```
phrases = Phrases(gberg_sents)
bigram = Phraser(phrases)
```

2-gram 語法配對

```
bigram.phrasegrams
```

輸出

接下頁

```
{(b'two', b'daughters'): (19, 11.966813731181546),
 (b'her', b'sister'): (195, 17.7960829227865),
 (b"'", b's'): (9781, 31.066242737744524),
 (b'very', b'early'): (24, 11.01214147275924),
 (b'Her', b'mother'): (14, 13.529425062715127),
 (b'long', b'ago'): (38, 63.223436289847 88),
 (b'more', b'than'): (541, 29.023584433996874),
 (b'had', b'been'): (1256, 22.306024648925288),
 (b'an', b'excellent'): (54, 39.063874851750626),
 (b'Miss', b'Taylor'): (48, 453.75918026073305),
 (b'very', b'fond'): (28, 24.134280468850747),
 (b'passed', b'away'): (25, 12.35053642325912),
 (b'too', b'much'): (173, 31.376002029426687),
 (b'did', b'not'): (935, 11.728416217142811),
 (b'any', b'means'): (27, 14.096964108090186),
 (b'wedding', b'-'): (15, 17.4695197740113),
 (b'Her', b'father'): (18, 13.129571562488772),
 (b'after', b'dinner'): (21, 21.5285481168817),
```

▲ 從語料庫檢測出的二元語法詞典

將 2-gram 語法轉換成單一 token

```
tokenized_sentence = "Jon lives in New York City".split()
```

```
bigram[tokenized_sentence]
```


```
['Jon', 'lives', 'in', 'New_York', 'City']
```

為整個語料庫進行資料預處理

- 進行資料預處理

▼ 將古騰堡計劃語料庫中的大寫全轉為小寫, 並移除標點符號

```
lower_sents = []
for s in gberg_sents:
    lower_sents.append([w.lower() for w in s if w.lower()
                        not in list(string.punctuation)])
```

```
lower_bigram = Phraser(Phrases(lower_sents))
```

```
{(b'two', b'daughters'): (19, 11.080802900992637),
 (b'her', b'sister'): (201, 16.93971298099339),
 (b'very', b'early'): (25, 10.516998773665177),
 (b'her', b'mother'): (253, 10.708126186077 42),
 (b'long', b'ago'): (38, 59.226442015336005),
 (b'more', b'than'): (562, 28.529926612065935),
 (b'had', b'been'): (1260, 21.583193129694834),
 (b'an', b'excellent'): (58, 37.41859680854167),
 (b'sixteen', b'years'): (15, 131.42913000977515),
 (b'miss', b'taylor'): (48, 420.4340982546865),
 (b'mr', b'woodhouse'): (132, 104.19907841850323),
 (b'very', b'fond'): (30, 24.185726346489627),
 (b'passed', b'away'): (25, 11.751473221742694),
 (b'too', b'much'): (177, 30.36309017383541),
 (b'did', b'not'): (977, 10.846196223896685),
 (b'any', b'means'): (28, 14.294148100212627),
 (b'after', b'dinner'): (22, 18.60737125272944),
 (b'mr', b'weston'): (162, 91.63290824201266),
```

▲ 從純小寫且無標點符號的語料庫檢測出的 2-gram 語法清單（僅節錄部分）

進一步過濾 2-gram 語法

```
lower_bigram = Phraser(Phrases(lower_sents,
                               min_count=32, threshold=64))
```

用更高的閾值來配對 2-gram 語法

lower_bigram.phrasegrams


```
{(b'miss', b'taylor'): (48, 156.44059469941823),
 (b'mr', b'woodhouse'): (132, 82.04651843976633),
 (b'mr', b'weston'): (162, 75.87438262077481),
 (b'mrs', b'weston'): (249, 160.68485093258923),
 (b'great', b'deal'): (182, 93.36368125424357),
 (b'mr', b'knightley'): (277, 161.74131790625913),
 (b'miss', b'woodhouse'): (173, 229.03802722366902),
 (b'years', b'ago'): (56, 74.31594785893046),
 (b'mr', b'elton'): (214, 121.3990121932397),
 (b'dare', b'say'): (115, 89.94000515807346),
 (b'frank', b'churchill'): (151, 1316.4456593286038),
 (b'miss', b'bates'): (113, 276.39588291692513),
 (b'drawing', b'room'): (49, 84.91494947493561),
 (b'mrs', b'goddard'): (58, 143.57843432545658),
 (b'miss', b'smith'): (58, 73.03442128232508),
 (b'few', b'minutes'): (86, 204.16834974753786),
 (b'john', b'knightley'): (58, 83.03755747111268),
 (b'don', b't'): (830, 250.30957446808512),
```

▲ 產生新的 2-gram 語法詞典（僅節錄部分）

處理整個語料庫

```
clean_sents = [] ←— 建立一個含有 2-gram 語法的乾淨語料庫
for s in lower_sents:
    clean_sents.append(lower_bigram[s])
```

```
clean_sents[6]
```


```
['sixteen',
 'years',
 'had',
 'miss_taylor',
 'been',
 'in',
 'mr_woodhouse',
 's',
 'family',
 'less',
 'as',
 'a',
 'governess',
 'than',
 'a',
 'friend',
 'very',
 'fond',
 'of',
 'both',
 'daughters',
 'but',
 'particularly',
 'of',
 'emma']
```

▲ 古騰堡計劃語料庫預處理後某個句子

用 word2vec 建立單詞嵌入向量

- word2vec 基礎知識

▼ word2vec 兩架構比較

架構	預測方法	優點
Skip-gram (SG)	根據目標詞預測脈絡詞	適合較小的語料庫；對罕見單詞較有利
CBOW	根據脈絡詞預測目標詞	快很多；對常見單詞較有利

評估詞向量

- 外部 (extrinsic) 評估與內部 (intrinsic) 評估。

使用 word2vec

```
model = Word2Vec(sentences=clean_sents, size=64,
                 sg=1, window=10, iter=5,
                 min_count=10, workers=4)
```

執行 word2vec

Word2Vec() 各參數說明

- sentences
- size
- sg
- window
- iter
- min_count
- workers

建立模型

```
model = gensim.models.Word2Vec.load('/content/drive/MyDrive/ 接下行
    (您存放的雲端硬碟目錄)/Ch11/ch11-clean_gutenberg_model.w2v')
```

查看模型（詞向量空間）的內容

```
len(model.wv.vocab)
```

前面在 Word2Vec() 內設定 min_count=10, 這就表示 clean_cents 語料庫內出現 10 次以上的單詞（token）有 10,329 個

```
10329
```

```
model.wv[ 'dog' ]
```


```
array([ 0.38401067,  0.01232518, -0.37594706, -0.00112308,  0.38663676,
        0.01287549,  0.398965  ,  0.0096426 , -0.10419296, -0.02877572,
        0.3207022 ,  0.27838793,  0.62772304,  0.34408906,  0.23356602,
        0.24557391,  0.3398472 ,  0.07168821, -0.18941355, -0.10122284,
       -0.35172758,  0.4038952 , -0.12179806,  0.096336  ,  0.00641343,
        0.02332107,  0.7743452 ,  0.03591069, -0.20103034, -0.1688079 ,
       -0.01331445, -0.29832968,  0.08522387, -0.02750671,  0.32494134,
       -0.14266558, -0.4192913 , -0.09291836, -0.23813559,  0.38258648,
        0.11036541,  0.005807  , -0.16745028,  0.34308755, -0.20224966,
       -0.77683043,  0.05146591, -0.5883941 , -0.0718769 , -0.18120563,
        0.00358319, -0.29351747,  0.153776  ,  0.48048878,  0.22479494,
        0.5465321 ,  0.29695514,  0.00986911, -0.2450937 , -0.19344331,
        0.3541134 ,  0.3426432 , -0.10496043,  0.00543602], dtype=float32)
```

▲ 「dog」這個詞在 64 維詞向量空間中的座標 (編：好難想像啊, 有 64 維呢)

評估生成的詞向量空間

```
model.wv.most_similar('father', topn=3)
```


```
[('mother', 0.8257375359535217),
('brother', 0.7275018692016602),
('sister', 0.7177823781967163)]
```

▼ 找出與測試單詞最相似的單詞

測試單詞	最相似單詞	得分
father	mother	0.82
dog	puppy	0.78
eat	drink	0.83
day	morning	0.76
ma_am	madam	0.85

```
model.wv.doesnt_match("mother father sister brother dog".split())
```

'dog' ← 分析出「dog」是這些單詞中最與眾不同的

```
model.wv.similarity('father', 'dog')
```


```
0.44234338
```

```
model.wv.most_similar(positive=['father', 'woman'], negative=['man'])
```

加項　　減項

輸出

```
[( ‘mother’ , 0.7650133371353149),
 ( ‘husband’ , 0.7556628584861755),
 ( ‘sister’ , 0.7482180595397949),
 ( ‘daughter’ , 0.7390402555465698),
 ( ‘wife’ , 0.7284981608390808),
 ( ‘sarah’ , 0.6856439113616943),
 ( ‘daughters’ , 0.6652647256851196),
```

← 算出來得分最高的詞為「mother」, 答案正確

```
model.wv.most_similar(positive=['husband', 'woman'], 接下行
    negative=['man'])
```

輸出

```
[( 'wife' , 0.707526445388794), ←— 最符合的單詞為「wife」, 也是
 ( 'sister' , 0.697398543357849 1),  正確答案, 這代表我們生成的詞
 ( 'maid' , 0.691125988960266 1),    向量空間應該能正常運作
 ( 'daughter' , 0.679954648017883 3),
 ( 'mother' , 0.658308148384094 2),
 ( 'child' , 0.643347144126892 1),
 ( 'conceived' , 0.639138460159301 8),
 ( 'harlot' , 0.608969330787658 7),
```

將詞向量空間描繪出來

- 用 t-SNE 進行降維

```
tsne = TSNE(n_components=2, n_iter=1000)
X_2d = tsne.fit_transform(model.wv[model.wv.vocab])
coords_df = pd.DataFrame(X_2d, columns=['x','y'])
coords_df['token'] = model.wv.vocab.keys()
```

用 t-SNE 進行降維

```
coords_df.to_csv('clean_gutenberg_tsne.csv', index=False)
```

```
coords_df = pd.read_csv('/content/drive/MyDrive/Colab_Notebooks/ 接下行
    (您的雲端硬碟存放路徑)/Ch11/ch11-clean_gutenberg_tsne.csv')
```

檢視 2 維的詞向量內容

```
coords_df.head()
```

輸出

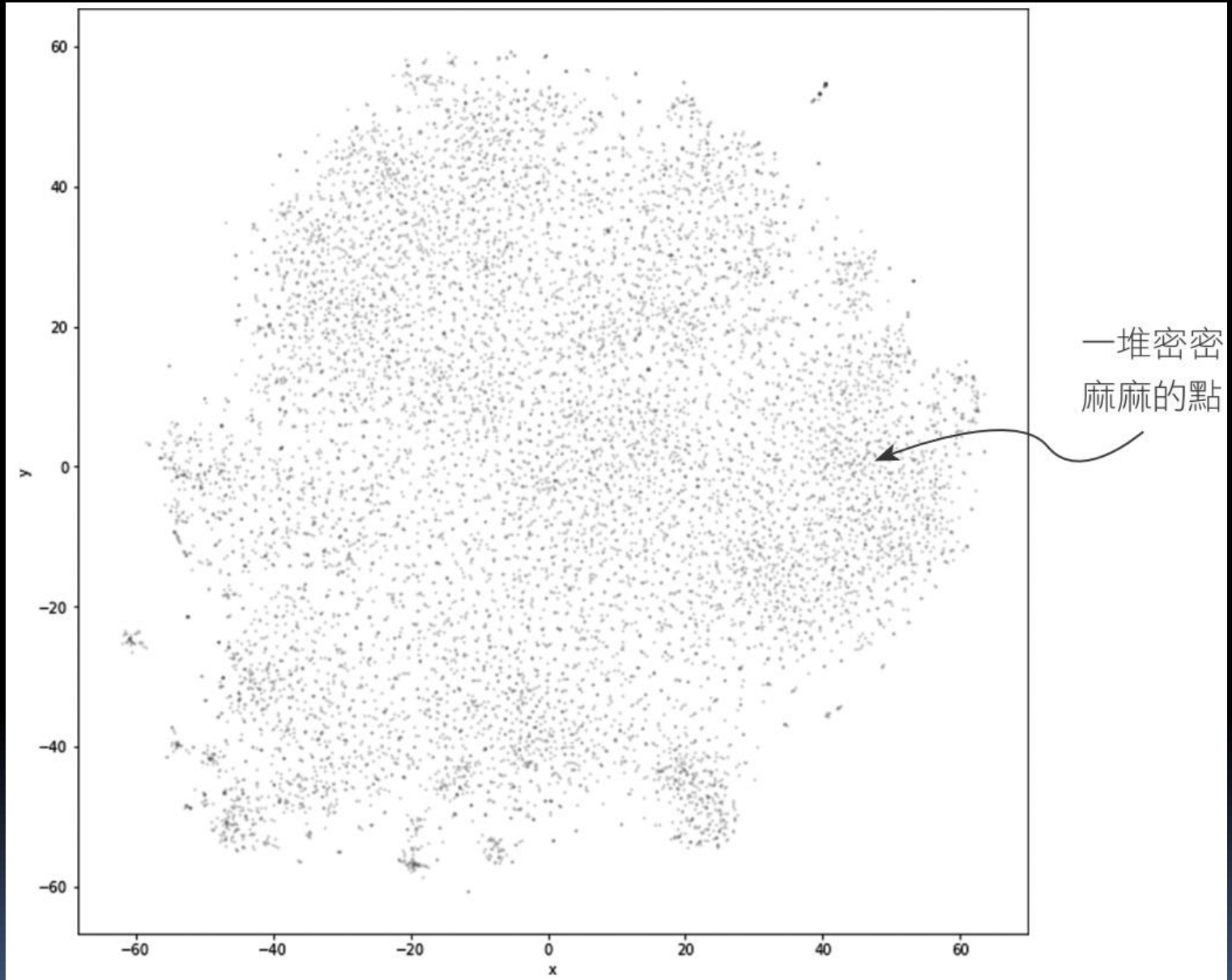
	x	y	token
0	62.494060	8.023034	emma
1	8.142986	33.342200	by
2	62.507140	10.078477	jane
3	12.477635	17.998343	volume
4	25.736960	30.876250	i

▲ 將語料庫創造的 64 維詞向量空間降為 2 維的結果, 每個 token (詞) 都以兩個數值表示 (編：想想好玄啊～兩個數值就能表示一個英文字)

將 2 維的詞向量繪成散布圖

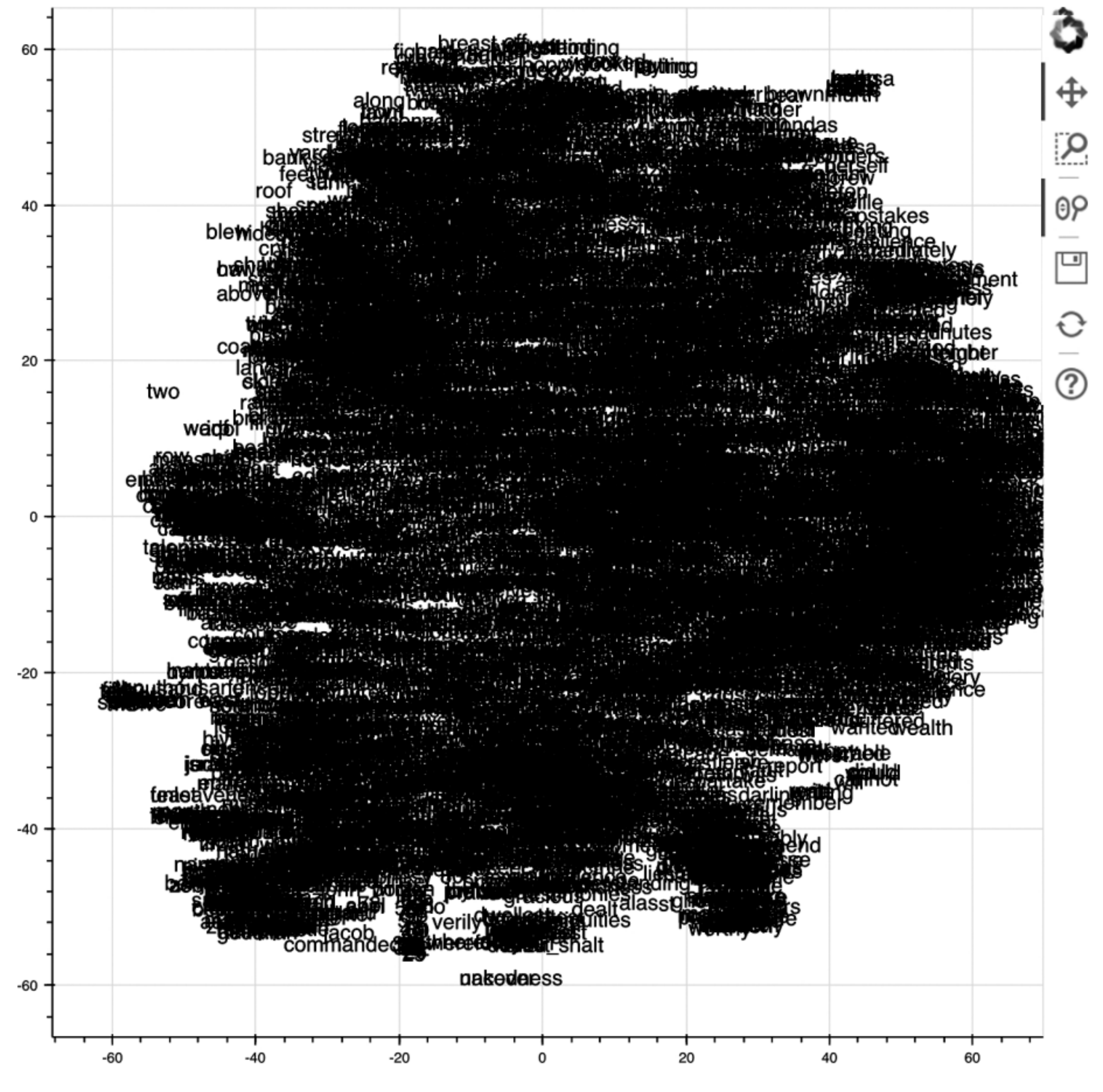
```
_ = coords_df.plot.scatter('x', 'y', figsize=(12,12),
                           marker='.', s=10, alpha=0.2)
```

用散布圖繪製詞向量空間



▲ 以散布圖呈現詞向量空間

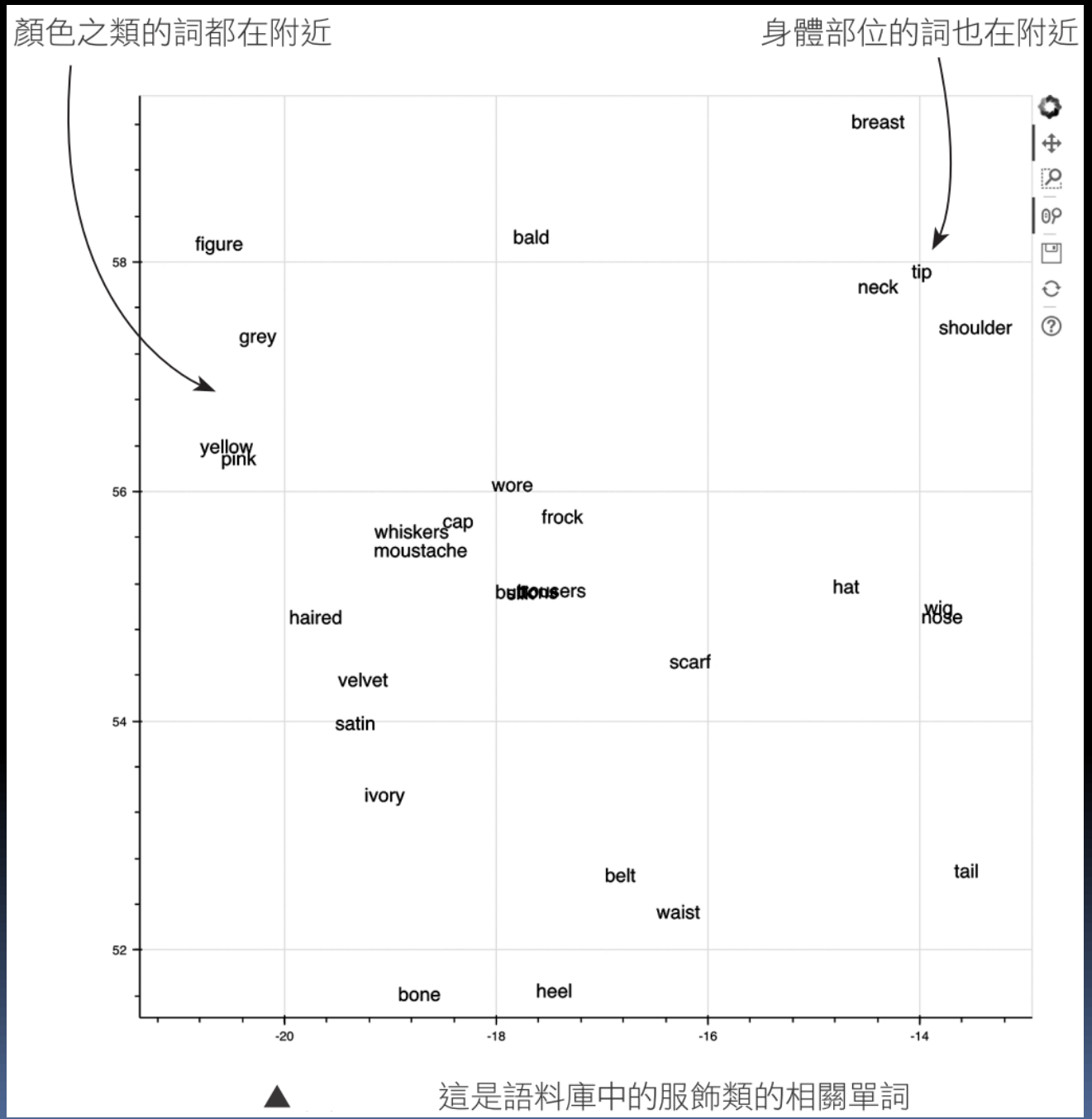
```
output_notebook()
subset_df = coords_df.sample(n=5000)
p = figure(plot_width=800, plot_height=800)
_ = p.text(x=subset_df.x, y=subset_df.y, text=subset_df.token)
show(p)
```



▲ 繪製好的散布圖

顏色之類的詞都在附近

身體部位的詞也在附近



這是語料庫中的服飾類的相關單詞