数据挖掘 Homework 3

赵晨阳 2020012363

数据预处理与可视化

运行方法

所依赖库都是基本常见的第三方库，可以手动安装。直接运行 `python3 main.py` 即可完成运行。

新闻数据读入与建立数据框对象

创建 `DataFrame`，利用 `elemmentTree` 等进行处理。生成的数据框对象中的属性包括日期、类别和时间，定义其展示顺序为 `["day", "month", "year", "classes", "content", "processed_content"]`，其中 `processed_content` 为经过预处理后的 `content`。完成了去除标点符号、停用词、数字、空白字符，将大写字母都转化为小写，以及词干化处理。

得到的 `DataFrame` 对象如下：

```
In [3]: df
Out[3]:
     day month  year                                            classes                                            content                                          processed_content
0     14     5  1997                                ['New York and Region']  WNYC, the public radio station that offers cl...  [wnyc, public, radio, station, offer, classic,...
1     27     4  1989  ['World', 'Sports', 'Sports', 'Sports', 'World...  LEAD: A shot-putter suspended for using banne...  [lead, shotputt, suspend, use, ban, drug, said...
2     11     1  2002  ['U.S.', 'U.S.', 'U.S.', 'World', 'World', 'He...  Many Americans are surprised at the speed wit...  [mani, american, surpris, speed, assur, immens...
3     10     1  1993                                              <NA>  To the Editor: We shouldn't be surprised at M...  [editor, shouldnt, surpris, milan, panic, defe...
4     25     4  1989  ['World', 'World', 'World', 'World', 'Washingt...  LEAD: The United States will withdraw a small...  [lead, unit, state, withdraw, small, number, a...
..    ..   ...   ...                                               ...                                                ...                                                ...
495   15    10  1989     ['New York and Region', 'New York and Region']  LEAD: THE eldest of my three daughters, a 19-...  [lead, eldest, three, daughter, yearold, cheer...
496    7    10  1995         ['U.S.', 'Obituaries', 'Travel', 'Travel']  Wladyslaw Otton Biernacki-Poray, an architect...  [wladyslaw, otton, biernackiporay, architect, ...
497    9     5  1990                                   ['Sports']  LEAD: CYCLING Bishop Is Winner of Seventh Sta...  [lead, cycl, bishop, winner, seventh, stage, b...
498    5     9  1992                                              <NA>  To the Editor: There is no difference in the ...  [editor, differ, way, hurrican, destroy, rich,...
499   29    11  1989                                 ['Business']  LEAD: Honeywell Inc. said it had agreed to se...  [lead, honeywel, inc, said, agre, sell, electr...

[500 rows x 6 columns]
```

预处理

预处理过程主要是各种词句的摘取、简化与筛选。综合使用了 `nltk` 库中的 `stopwords` ，`SnowballStemmer` 等词句处理工具，尽管 `nltk` 库相对 `HuggingFace` 等等 toolkits 相对简单，但是也取得了非常良好的效果。其他处理工具还有 `re` 库中的替换函数 `sub`，字符串自带的 `lower` 等通用函数。经过去除标点符号、停用词、数字、空白字符、大写转小写、词干化等预处理后，得到 `all_words` 长度为151811，而 `all_classes` 长度为 1683。

BOW 表示

将每一篇新闻的全文表示成 BagOfWords 向量，根据相应计数方式计算 BagOfWords 向量：

```
def create_bag_of_words(
    processed_content: List[str], all_unique_words: List[str]
) -> np.ndarray:
    bag_of_words = np.zeros(len(all_unique_words))
    for word in processed_content:
        if word in all_unique_words:
            bag_of_words[all_unique_words.index(word)] += 1
    return bag_of_words
```

得到的 bag_of_words_all 情况如下：

```
In [16]: bag_of_words_all.shape
Out[16]: (495, 16552)

In [17]: bag_of_words_all.sum()
Out[17]: 151811.0
```

词云图

利用 Counter 类可以便捷地统计词频信息：

```
In [18]:     top_words = Counter(all_words).most_common(100)
    ...:     print(top_words)
[('said', 1932), ('mr', 1462), ('new', 850), ('year', 828), ('would',
667), ('one', 646), ('state', 595), ('compani', 541), ('like', 473),
('also', 453), ('time', 452), ('two', 419), ('peopl', 415), ('last',
414), ('work', 402), ('say', 389), ('york', 361), ('american', 359),
('nation', 340), ('percent', 331), ('first', 325), ('unit', 306),
('make', 305), ('school', 303), ('go', 295), ('mani', 293), ('million',
293), ('presid', 289), ('even', 288), ('includ', 283), ('get', 279),
('use', 277), ('day', 275), ('citi', 274), ('could', 273), ('report',
260), ('today', 259), ('call', 253), ('street', 251), ('offici', 247),
('may', 246), ('offic', 234), ('three', 233), ('share', 230), ('way',
228), ('month', 225), ('govern', 221), ('hous', 218), ('ms', 218),
('open', 214), ('much', 213), ('pm', 209), ('week', 207), ('group', 206),
('sale', 206), ('univers', 206), ('center', 205), ('made', 205),
('world', 199), ('countri', 199), ('take', 196), ('still', 196),
('public', 195), ('come', 195), ('law', 195), ('lead', 194), ('chang',
190), ('program', 189), ('want', 187), ('sinc', 187), ('tax', 184),
('plan', 184), ('art', 184), ('play', 181), ('part', 180), ('famili',
180), ('back', 180), ('live', 179), ('run', 178), ('look', 178), ('seem',
177), ('dr', 173), ('book', 170), ('anoth', 170), ('well', 168), ('need',
168), ('life', 167), ('recent', 165), ('sever', 164), ('end', 164),
('home', 163), ('think', 163), ('place', 162), ('yesterday', 161),
('right', 161), ('director', 161), ('help', 160), ('market', 159),
('long', 159), ('case', 158)]
```

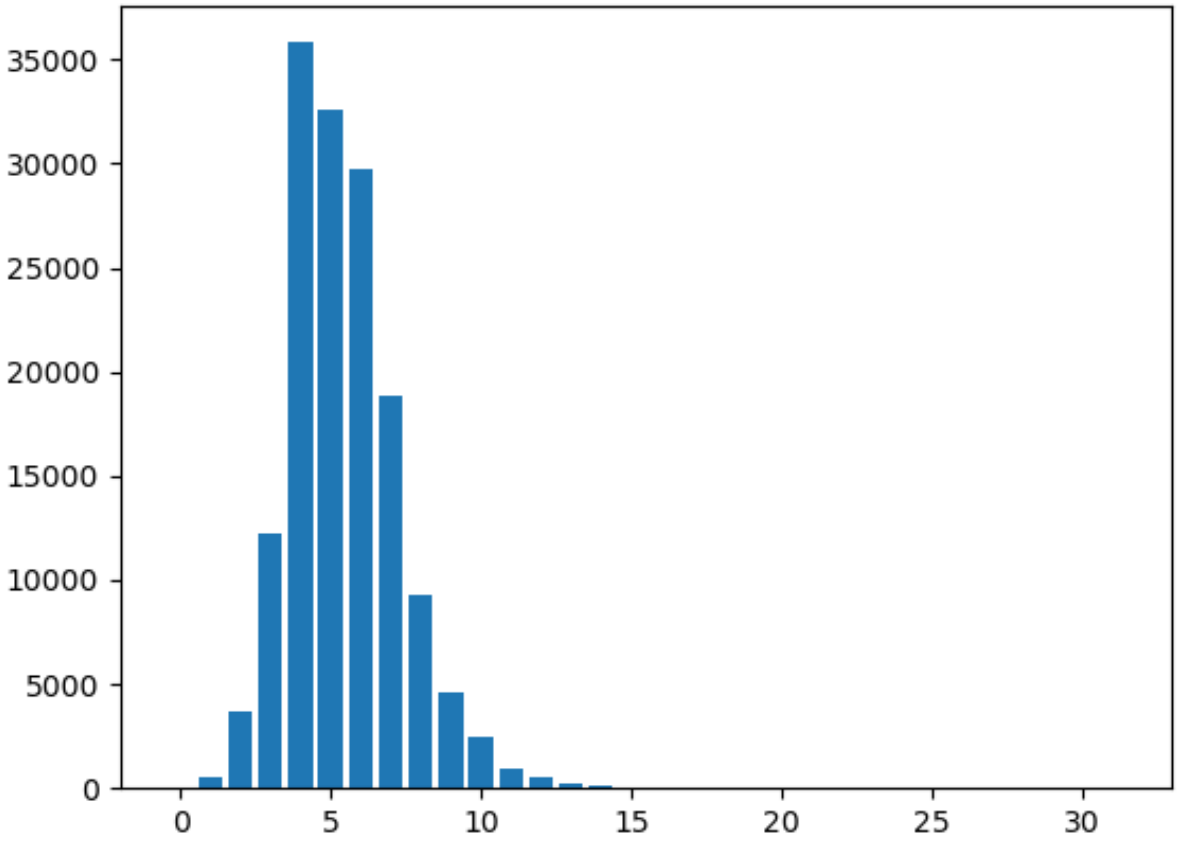
随后绘制出的词云图如下:

单词长度分布

统计得到最长的单词不超过 31，进一步得到的单词长度分布如下：

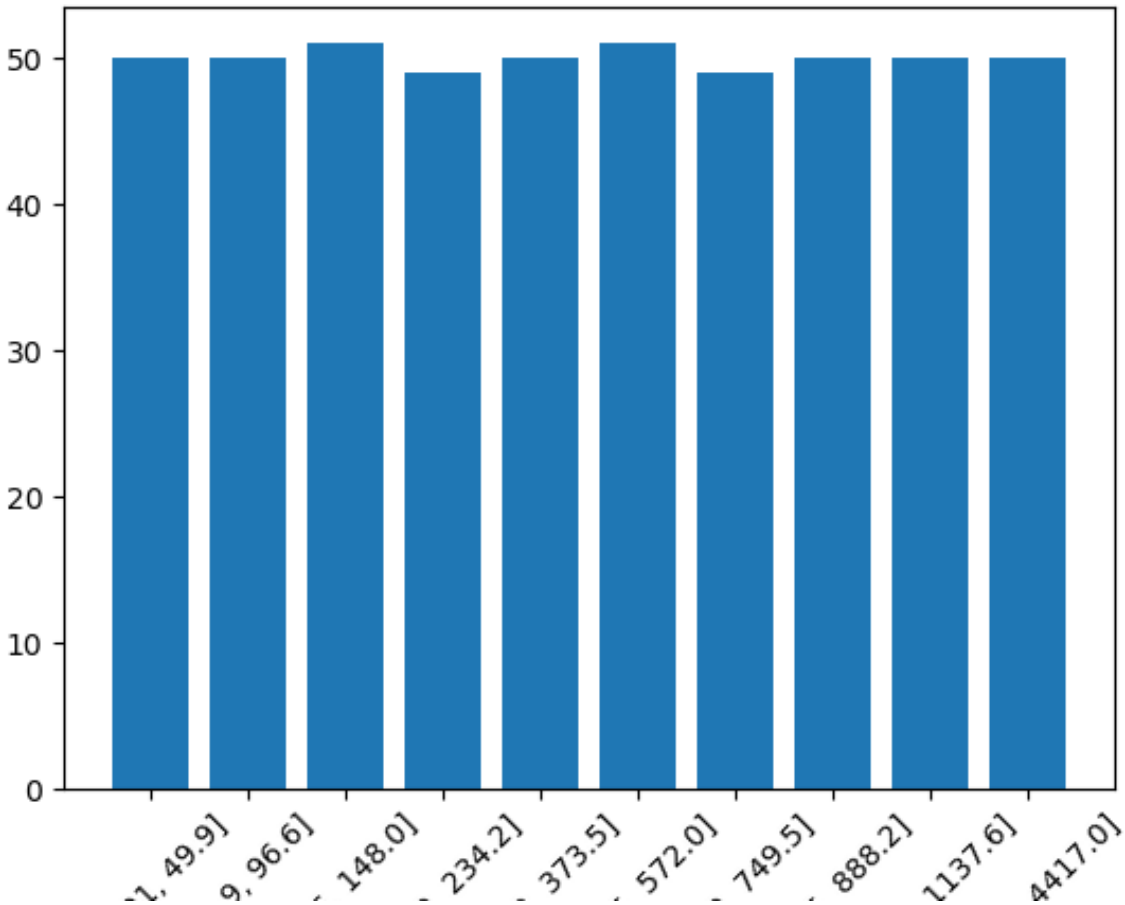
```
In [3]: word_lengths
Out[3]:
array([0.0000e+00, 5.3700e+02, 3.6280e+03, 1.2240e+04, 3.5821e+04,
       3.2608e+04, 2.9747e+04, 1.8821e+04, 9.2570e+03, 4.6430e+03,
       2.4900e+03, 9.3900e+02, 5.3200e+02, 2.5200e+02, 1.3400e+02,
       6.2000e+01, 4.1000e+01, 1.2000e+01, 2.7000e+01, 4.0000e+00,
       5.0000e+00, 2.0000e+00, 0.0000e+00, 1.0000e+00, 3.0000e+00,
       1.0000e+00, 2.0000e+00, 1.0000e+00, 0.0000e+00, 1.0000e+00,
       0.0000e+00, 0.0000e+00])
```

得到的柱状图如下：

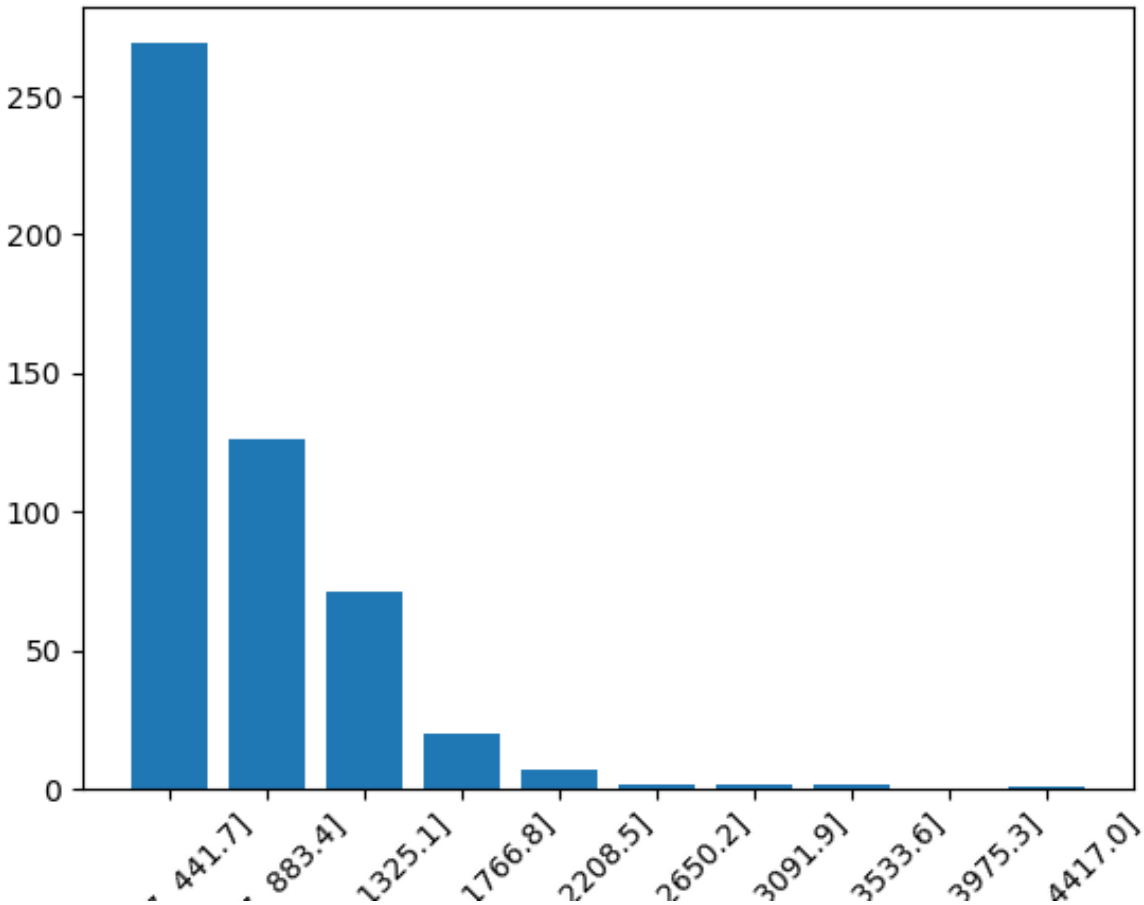


等深分箱与等宽分箱

等深分箱如下：

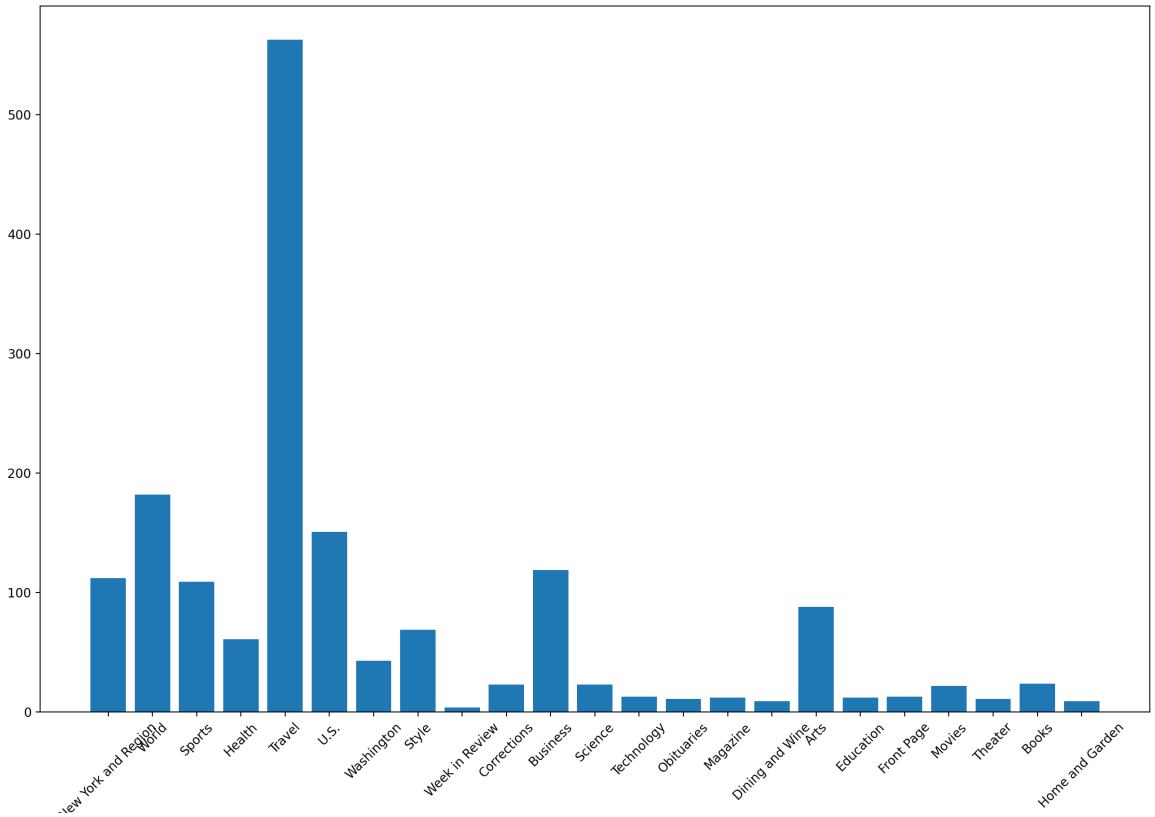

50
40
30
20
10
0
1, 49.9]
9, 96.6]
148.0]
234.2]
373.5]
572.0]
749.5]
888.2]
1137.6]
4417.0]

等宽分箱如下：



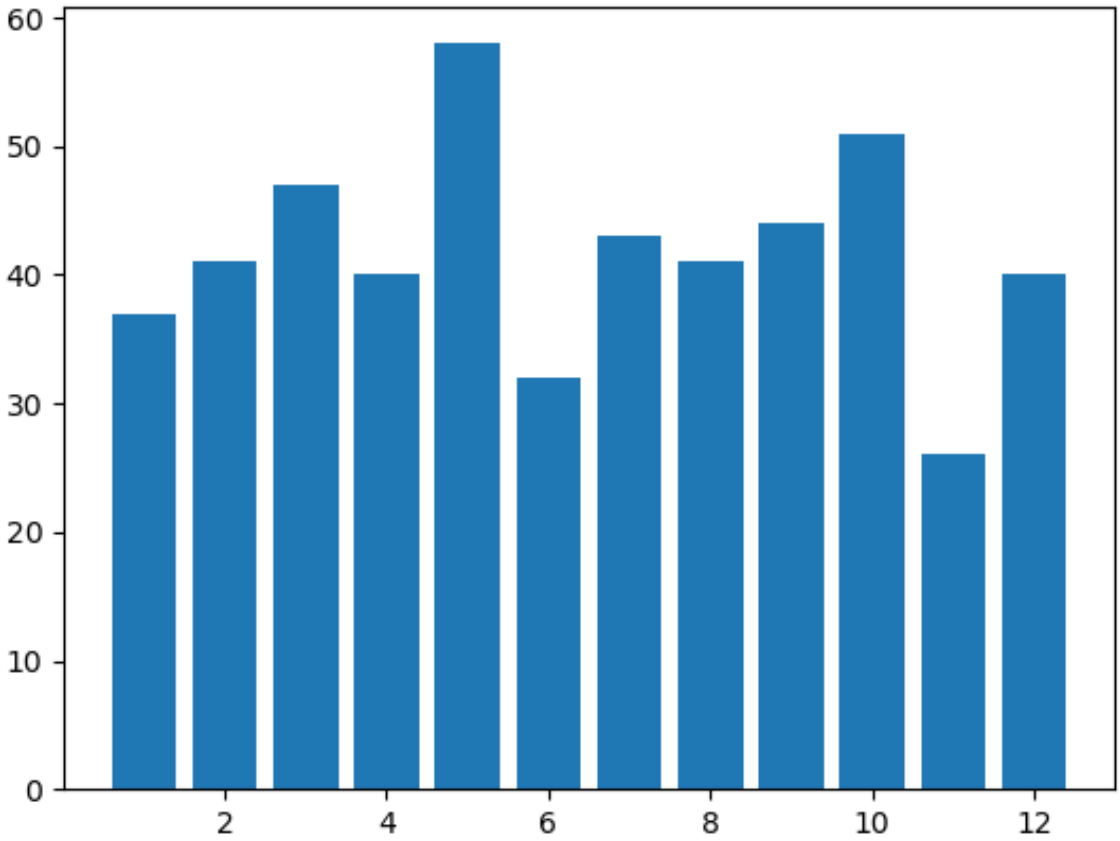
按类别新闻数量分布柱状图

根据结果绘制出柱状图，分布情况如柱状图所示：



按月新闻数量分布柱状图

根据结果绘制出柱状图，分布情况如柱状图所示：



高维向量可视化

实现思路

利用 Python，基于 PCA 和 t-SNE 降维技术来分析文本数据。首先从本地读取文本数据，然后使用 sklearn 的 PCA 算法进行降维并绘制出 PCA 降维图，使用 openTSNE 库中的 t-SNE 算法进行降维并绘制出 t-SNE 降维图。在绘制降维图时，使用了 matplotlib 和 seaborn 这两个 Python 可视化库。

为了运行这段代码，需要安装以下 Python 库：

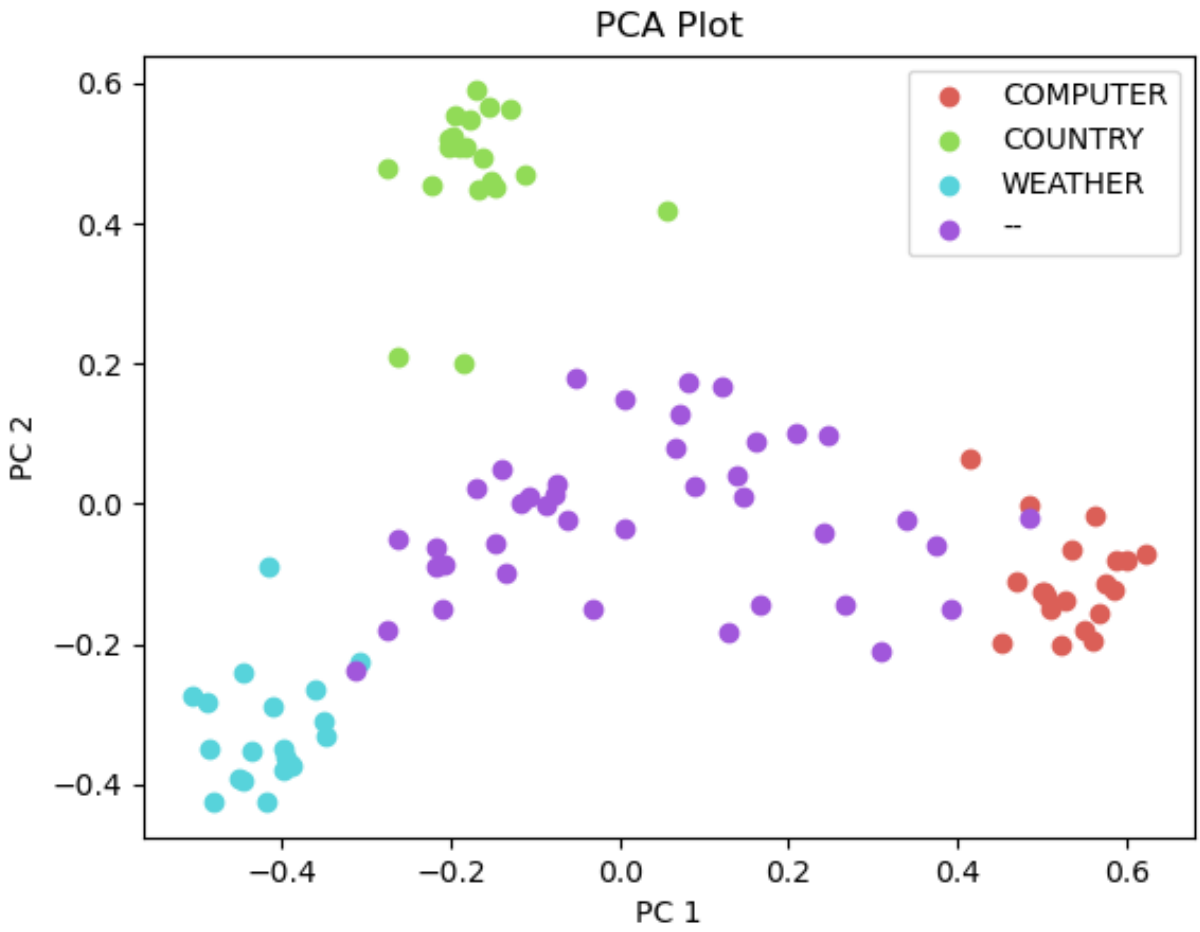
- pandas
- numpy
- matplotlib
- seaborn
- scikit-learn
- openTSNE

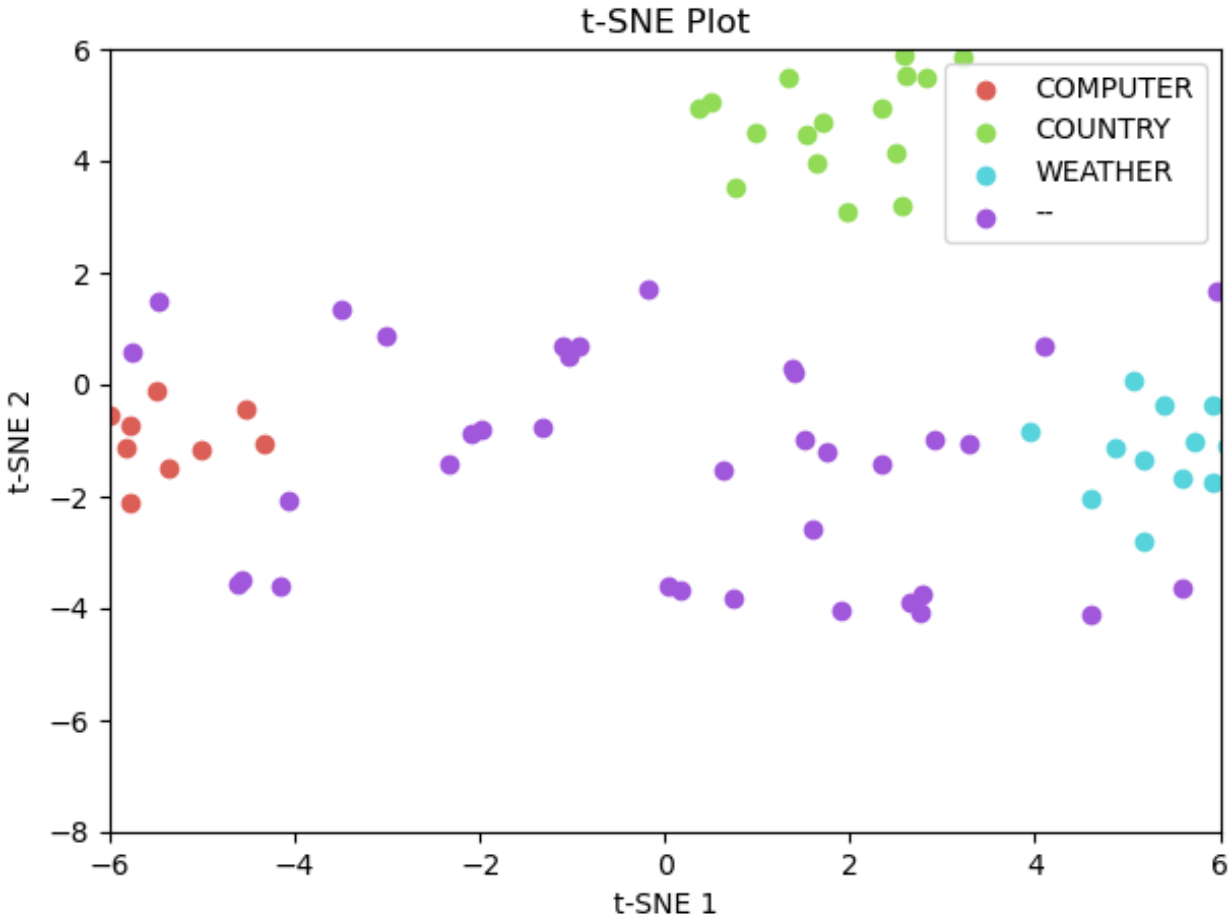
代码运行

代码运行

直接将 `100_word_vector.txt` 放置于当前路径下即可运行。

结果





分析

降维效果

在本次作业场景中，t-SNE 和 PCA 两种降维算法均能够有效发挥作用。

综合来看，在我所选取的 t-SNE 参数下，PCA 算法的降维效果要优于 t-SNE 算法，特别是在分析高维度数据时。可以观察到，PCA 算法降维后的词聚类更加紧密、分块明显，且没有出现远距离的离散点，因此更能够准确地反映出词语之间的关联性和差异性。PCA 算法的优越性在第 21 - 40 个单词中表现得比较明显，在第 1 - 20 个和第 41 - 60 个单词上也有一定体现。

虽然一般而言，t-SNE 的效果会更好，然而我对 perplexity 和 random seed 进行过多次搜索，均未找到 t-SNE 效果明显优于 PCA 的情况，而且一般而言，t-SNE 的效果不如 PCA。因此，虽然 t-SNE 可能对于大多数的高维向量情况较为优秀，然而并不适用于此例，还需要研究人员探索更为优秀的降维可视化方法。

词义分析

根据词义，可大致分为 4 组：

根据词义，可大致分为 4 组：

- 1-20：计算机（computer）相关单词
- 21-40：国家（country）相关单词
- 41-60：天气（weather）相关单词
- 61-100：不易分类单词（--）

我们发现，第 1 - 3 组可以根据词义很好的分离，在降维后的图中有分块聚集；第 4 组词义本身较分散，因此在降维后的图中比较分散，符合数据本身的意义。