ImportingLanguageData

January 15, 2021

1 Extracting Language Data from Various Sources

Introduction to NLP for Data Science. Week 1

1.1 PDF Import

[1]:
```
import PyPDF2
import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
pdfFileObject = open('test.pdf','rb')
```

[3]:
```
pdfReaderObject = PyPDF2.PdfFileReader(pdfFileObject)
print(pdfReaderObject.numPages)
```

```
1
```

[4]:
```
firstPageObject = pdfReaderObject.getPage(0)
#print(firstPageObject.extractText())
```

Extract all text for processing

[5]:
```
pdf_data = ""
number_of_pages = pdfReaderObject.numPages
for page_number in range(number_of_pages):   # use xrange in Py2
    page = pdfReaderObject.getPage(page_number)
    page_content = page.extractText()
    pdf_data+=page_content
```

[6]:
```
pdf_data = []
number_of_pages = pdfReaderObject.numPages
for page_number in range(number_of_pages):   # use xrange in Py2
    page = pdfReaderObject.getPage(page_number)
    page_content = page.extractText()
    pdf_data.append(page_content)
' '.join(pdf_data)
```

[6]:
```
" \nLorem Ipsum\n \nis simply dummy text of the printing and typesetting
industry. Lorem Ipsum has been \nthe industry's standard dummy text ever since
the 1500s, when an unknown printer took a galley of \ntype and scrambled it to
make a type specimen book. It has survived not o\nnly five centuries, but also
\nthe leap into electronic typesetting, remaining essentially unchanged. It was
popularised in the 1960s \nwith the release of Letraset sheets containing Lorem
Ipsum passages, and more recently with desktop \npublishing software like\n
\nAldus PageMaker including versions of Lorem Ipsum.\n \nWhy do we use it?\n
\nIt is a long established fact that a reader will be distracted by the readable
content of a page when \nlooking at its layout. The point of using Lorem Ipsum
is that it has a more\n-\nor\n-\nless \nnormal distribution \nof letters, as
opposed to using 'Content here, content here', making it look like readable
English. Many \ndesktop publishing packages and web page editors now use Lorem
Ipsum as their default model text, \nand a search for 'lorem ipsum' wi\nll
uncover many web sites still in their infancy. Various versions have \nevolved
over the years, sometimes by accident, sometimes on purpose (injected humour and
the like).\n \n \n"
```

[7]:
```
pdfFileObject.close()
```

[8]:
```
#pdf_data
```

1.2 JSON Documents

[9]:
```
import requests
import json
```

[10]:
```
r = requests.get("https://quotes.rest/qod.json") # Note your quore result will␣
↪be different
res = r.json()
print(json.dumps(res, indent = 4))
```

```
{
    "success": {
        "total": 1
    },
    "contents": {
        "quotes": [
            {
                "quote": "If we are not stretching, we are not growing. If we
are not growing, we are probably not fulfilling our potential. The only person
in this room that knows your potential is you.",
                "length": "177",
                "author": "Kevin Turner ",
                "tags": [
                    "comfort-zone",
```

```
                    "growth",
                    "inspire"
                ],
                "category": "inspire",
                "language": "en",
                "date": "2021-01-15",
                "permalink": "https://theysaidso.com/quote/kevin-turner-if-we-
are-not-stretching-we-are-not-growing-if-we-are-not-growing-w",
                "id": "uZl4zvREAf71Q6XfbAs8aweF",
                "background": "https://theysaidso.com/img/qod/qod-inspire.jpg",
                "title": "Inspiring Quote of the day"
            }
        ]
    },
    "baseurl": "https://theysaidso.com",
    "copyright": {
        "year": 2023,
        "url": "https://theysaidso.com"
    }
}
```


[11]:
```
#extract contents
q = res['contents']['quotes'][0]
```

[12]:
```
q
```

[12]:
```
{'quote': 'If we are not stretching, we are not growing. If we are not growing,
we are probably not fulfilling our potential. The only person in this room that
knows your potential is you.',
 'length': '177',
 'author': 'Kevin Turner ',
 'tags': ['comfort-zone', 'growth', 'inspire'],
 'category': 'inspire',
 'language': 'en',
 'date': '2021-01-15',
 'permalink': 'https://theysaidso.com/quote/kevin-turner-if-we-are-not-
stretching-we-are-not-growing-if-we-are-not-growing-w',
 'id': 'uZl4zvREAf71Q6XfbAs8aweF',
 'background': 'https://theysaidso.com/img/qod/qod-inspire.jpg',
 'title': 'Inspiring Quote of the day'}
```

[13]:
```
print(q['quote'], '\n--', q['author'])
```

```
If we are not stretching, we are not growing. If we are not growing, we are
probably not fulfilling our potential. The only person in this room that knows
your potential is you.
-- Kevin Turner
```

1.3 HTML Import

[14]:
```
import urllib.request as urllib3
from bs4 import BeautifulSoup
```

[15]:
```
response = urllib3.urlopen('https://en.wikipedia.org/wiki/
 ↪Natural_language_processing')
html_doc = response.read()
```

[16]:
```
soup = BeautifulSoup(html_doc, 'html.parser')
strhtm = soup.prettify()
print(strhtm[:1000])
```

```
<!DOCTYPE html>
<html class="client-nojs" dir="ltr" lang="en">
 <head>
  <meta charset="utf-8"/>
  <title>
   Natural language processing - Wikipedia
  </title>
  <script>
   document.documentElement.className="client-js";RLCONF={"wgBreakFrames":!1,"wg
SeparatorTransformTable":["",""],"wgDigitTransformTable":["",""],"wgDefaultDateF
ormat":"dmy","wgMonthNames":["","January","February","March","April","May","June
","July","August","September","October","November","December"],"wgRequestId":"X-
zBEQpAICOAADw4l78AAADF","wgCSPNonce":!1,"wgCanonicalNamespace":"","wgCanonicalSp
ecialPageName":!1,"wgNamespaceNumber":0,"wgPageName":"Natural_language_processin
g","wgTitle":"Natural language processing","wgCurRevisionId":999766273,"wgRevisi
onId":999766273,"wgArticleId":21652,"wgIsArticle":!0,"wgIsRedirect":!1,"wgAction
":"view","wgUserName":null,"wgUserGroups":["*"],"wgCategories":["CS1 maint:
location","Articles with short description","Short description matches
Wikidata","Commons link from Wikidata","W
```

[17]:
```
print(soup.title)
print(soup.title.string)
#for x in soup.find_all('p'): print(x.text)
```

```
<title>Natural language processing - Wikipedia</title>
Natural language processing - Wikipedia
```

1.4 Tweeter API

[18]:
```
import numpy as np
import tweepy
import json
import pandas as pd
```

```
from tweepy import OAuthHandler
```

[19]:
```
# credentials [type your credentials between quotes]
consumer_key = ""
consumer_secret = ""
access_token = ""
access_token_secret = ""
```

[20]:
```
# calling API
#auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
#auth.set_access_token(access_token, access_token_secret)
#api = tweepy.API(auth)
```

[21]:
```
# Provide the query you want to search.
query ="ABC"
```

[22]:
```
#home timeline, print each tweets text. Your results will be different
#public_tweets = api.home_timeline()
#for tweet in public_tweets:
 #   print(tweet.text)
```

[23]:
```
#public_tweets[0]
#status = public_tweets[0]
#json_str = json.dumps(status._json)
#parsed = json.loads(json_str)
#print(json.dumps(parsed, indent=4, sort_keys=True))
```

[24]:
```
#status.text
```

[25]:
```
# Fetching tweets with the query "ABC"
#for tweet in tweepy.Cursor(api.search,
#                           q="ABC",
#                           rpp=100,
#                           result_type="recent",
#                           include_entities=True,
 #                          lang="en").items(5):
 #   print(tweet.text)
```

1.5 Google Trends

[26]:
```
# !pip install pytrends #uncomment if you need to install it
```

[27]:
```
import pandas as pd
from pytrends.request import TrendReq
pytrend = TrendReq()
```

[28]:
```
keywords = ['Python', 'R'] # select your own keywords
```

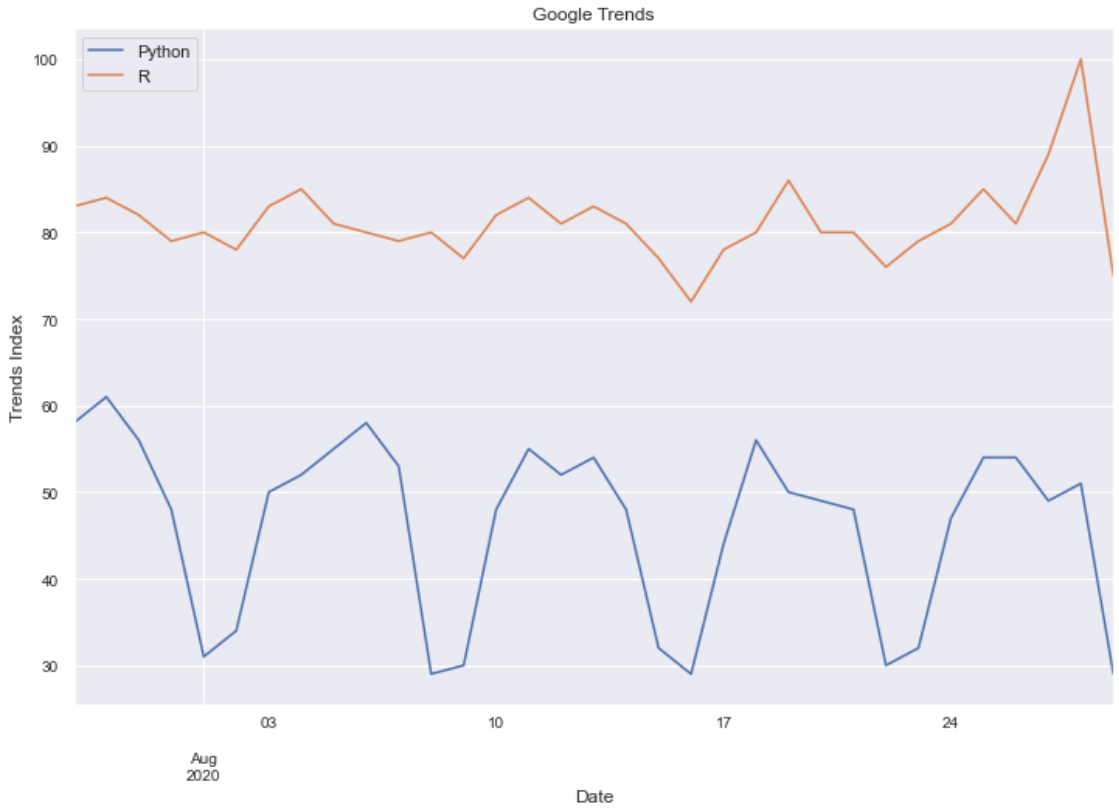
[29]:
```
pytrend.build_payload(kw_list = keywords, timeframe='2020-07-28 2020-08-29',␣
↪geo='US') # select your timeframe
```

[30]:
```
df = pytrend.interest_over_time()
#df.columns = df.columns.droplevel(0) #drop outside header
df = df.drop('isPartial', axis = 1) # drop column isPartial
df.reset_index(level=0,inplace=True) #reset_index
df.columns=['date','Python','R']
df.head()
```

[30]:

	date	Python	R
0	2020-07-28	58	83
1	2020-07-29	61	84
2	2020-07-30	56	82
3	2020-07-31	48	79
4	2020-08-01	31	80

[31]:
```
import seaborn as sns
sns.set(color_codes=True)
fig = df.plot(figsize = (12,8),x="date", y=['Python','R'], kind="line", title =␣
↪"Google Trends")
fig.set_xlabel('Date')
fig.set_ylabel('Trends Index')
fig.tick_params(axis='both', which='both', labelsize=10)
```



[32]:
```
pytrend.interest_by_region(resolution='COUNTRY', inc_low_vol=True,␣
↪inc_geo_code=False)
```

[32]:

geoName	Python	R
Alabama	28	72
Alaska	32	68
Arizona	32	68
Arkansas	23	77
California	51	49
Colorado	37	63
Connecticut	35	65
Delaware	25	75
District of Columbia	35	65
Florida	27	73
Georgia	29	71
Hawaii	29	71
Idaho	32	68
Illinois	33	67
Indiana	26	74
Iowa	34	66

```
Kansas                27  73
Kentucky              27  73
Louisiana             17  83
Maine                 29  71
Maryland              40  60
Massachusetts         47  53
Michigan              28  72
Minnesota             34  66
Mississippi           19  81
Missouri              28  72
Montana               27  73
Nebraska              24  76
Nevada                22  78
New Hampshire         38  62
New Jersey            41  59
New Mexico            24  76
New York              39  61
North Carolina        34  66
North Dakota          24  76
Ohio                  28  72
Oklahoma              21  79
Oregon                46  54
Pennsylvania          29  71
Rhode Island          25  75
South Carolina        22  78
South Dakota          16  84
Tennessee             27  73
Texas                 34  66
Utah                  37  63
Vermont               33  67
Virginia              48  52
Washington            47  53
West Virginia         19  81
Wisconsin             25  75
Wyoming               26  74
```

[ ]: