main

October 5, 2024

0.1 ideia - conceitos associados a empresas portuguesas

- decidir X empresas portuguesas
- analisar noticias sobre as mesmas e encontrar palavras/pessoas/temas associadas (fazer grafo) [ver se é positivo / negativo o termo/pessoa]
- tendo por base a tendência dessas palavras (+- faladas), analisar performance da empresa

---

Trabalho tem de ter 3 partes: 1. project structure + data acquisition 2. exploratory data analysis and visualization 3. results & discussion

Fonte de Dados: arquivo.pt (https://github.com/arquivo/pwa-technologies/wiki/Arquivo.pt-API)

onde fui buscar site de notícias: https://www.kadaza.pt

---

pode ser preciso filtrar sites, pq não queremos primeiras páginas de notícias

[1]:
```
import pandas as pd
import requests
from bs4 import BeautifulSoup
```

1 which companies to study

some PSI-20 companies

[2]:
```
def datav1(companies):
    """
    this is the function where we choose the companies which will be in study
    -
    companies should be a dictionary
        {"company1": [aliases or other names the company is or was known by],
        "company2": [...]}
    this data will be saved into a parquet file for future use
    """
    parquet = {"companies": [], "aliases": []}
    for company in companies.keys():
        parquet["companies"].append(company)
```

```
        parquet["aliases"].append(companies[company])

    df = pd.DataFrame(parquet).set_index("companies")
    df.to_parquet("data01.parquet")
    return df


companies = {"Banco Comercial Português": ["Banco Comercial Português", "BCP"],
             "Galp Energia": ["Galp Energia", "GALP"],
             "EDP": ["EDP", "Energias de Portugal", "Electricidade de␣
 ↪Portugal"],
             "Sonae": ["Sonae", "SON"],
             "Mota-Engil": ["Mota-Engil", "EGL"]}

datav1(companies)
```

[2]:

companies	aliases
Banco Comercial Português	[Banco Comercial Português, BCP]
Galp Energia	[Galp Energia, GALP]
EDP	[EDP, Energias de Portugal, Electricidade de P…
Sonae	[Sonae, SON]
Mota-Engil	[Mota-Engil, EGL]

2 where to grab the news from

[3]:
```
def news(txtFile = 'noticias.txt'):
    """
    grab the news websites from a text file
    """
    with open(txtFile, 'r') as file:
        links = file.read().splitlines()
    return ",".join(links)

news()
```

[3]:
```
'www.publico.pt,publico.pt,www.dn.pt,www.rtp.pt,rpt.pt,www.cmjornal.pt,www.iol.p
t,www.tvi24.iol.pt,tvi24.iol.pt,noticias.sapo.pt,observador.pt,expresso.pt,www.e
xpresso.pt,sol.sapo.pt,www.jornaldenegocios.pt,www.jn.pt,jn.pt,ionline.pt,sicnot
icias.pt,www.sicnoticias.pt,www.lux.iol.pt,www.ionline.pt,news.google.pt,www.din
heirovivo.pt,www.aeiou.pt,aeiou.pt,www.tsf.pt,tsf.pt,www.sabado.pt,dnoticias.pt,
www.dnoticias.pt,economico.sapo.pt,cnnportugal.iol.pt'
```

3 api requests in 3 years intervals

1 year to **3 years** is long enough to smooth out short-term fluctuations and identify underlying trends. Charts with weekly or monthly intervals over these periods show developments over full economic/market cycles.

[4]:
```
def api_request(search, websites, date):
    """
    search: expression/word (what to look for)
    websites: comma separated websites (where to look for)
    date: list such as [20030101, 20031231] (when to look for)
    -
    returns the responde_items from arquivo.pt api
    """

    search = f"q=%22{search.replace(' ', '%20')}%22"
    websites = f"&siteSearch={websites}"
    date = f"&from={date[0]}&to={date[1]}"

    url = (
        f"https://arquivo.pt/textsearch?{search}{websites}{date}"
        "&fields=title,linkToArchive,tstamp,linkToExtractedText,snippet"
        "&maxItems=500&dedupValue=25&dedupField=url&prettyPrint=false&type=html"
        )

    req = requests.get(url)
    json = req.json()
    data = json["response_items"]
    if len(data) == 500:
        print(f"You might have lost some data: {search, date}")

    return data
```

[5]:
```
df = pd.read_parquet("data01.parquet",)
websites = news()

for cluster in range(2000, 2021, 3):
    api_cluster = [] #reset api_cluster for each cluster (group of 3 year)
    print(f"Processing cluster: {cluster}")
    print("Processing company:", end=" ")
    for company_aliases in df["aliases"]:
        api_company = [] #reset api_company for each company
        print(f"{company_aliases[0]}", end = "; ")

        for alias in company_aliases:
            for year in range(cluster, cluster + 3):
```

```
                api_aliasS1 = api_request(alias, websites, [int(f"{year}0101"),␣
↪int(f"{year}0630")])
                api_aliasS2 = api_request(alias, websites, [int(f"{year}0701"),␣
↪int(f"{year}1231")])
                api_company += api_aliasS1 + api_aliasS2

        api_cluster.append(api_company)


    df[f"api.{cluster}"] = api_cluster
    print(f"{cluster} OK.")

df.to_parquet("data02.parquet")
print("Finished.")
```

```
Processing cluster: 2000
Processing company: Banco Comercial Português; Galp Energia; EDP; Sonae; Mota-
Engil; 2000 OK.
Processing cluster: 2003
Processing company: Banco Comercial Português; Galp Energia; EDP; Sonae; Mota-
Engil; 2003 OK.
Processing cluster: 2006
Processing company: Banco Comercial Português; Galp Energia; EDP; Sonae; Mota-
Engil; 2006 OK.
Processing cluster: 2009
Processing company: Banco Comercial Português; Galp Energia; EDP; Sonae; Mota-
Engil; 2009 OK.
Processing cluster: 2012
Processing company: Banco Comercial Português; Galp Energia; EDP; Sonae; Mota-
Engil; 2012 OK.
Processing cluster: 2015
Processing company: Banco Comercial Português; Galp Energia; EDP; Sonae; Mota-
Engil; 2015 OK.
Processing cluster: 2018
Processing company: Banco Comercial Português; Galp Energia; EDP; Sonae; Mota-
Engil; 2018 OK.
Finished.
```

[6]:
```
df.map(lambda x: len(x))
```

[6]:

companies	aliases	api.2000	api.2003	api.2006	api.2009	\
Banco Comercial Português	2	153	241	183	561	
Galp Energia	2	128	389	272	582	
EDP	3	133	339	173	653	
Sonae	2	192	435	279	502	
Mota-Engil	2	4	83	60	195	

companies	api.2012	api.2015	api.2018
Banco Comercial Português	1074	1430	954
Galp Energia	1156	1391	968
EDP	1232	1970	1096
Sonae	1215	1705	1196
Mota-Engil	538	828	560

4 if the url is the same, check the content to see if its repeated

because we used `&dedupValue=25&dedupField=url` and different aliases, we might have repeated data, so it's important to check for it

and also check for extractedText that doesn't have our aliases

[ ]: