Job search and application

In [1]:
```
# import libraries

import numpy as np
import pandas as pd
import regex as re # to clean text
import os #why?
import time # to save csv to today's date
import requests
from bs4 import BeautifulSoup #to get data out of html or xml
from collections import Counter # to count keyword appearance
```

In [2]:
```
# read the text file

with open('./urls.txt',"r") as urls:
    urls = urls.readlines()
```

In [3]:
```
#agent to interact with web content

hdr = {'User-Agent': 'Mozilla/5.0'}
```

In [4]:
```
# check the request is successful

res = []
for url in urls:
    res.append(requests.get(url,headers=hdr))

for i in range(len(res)):
    if(res[i].status_code) != 200:
        print(i,res[i].status_code)
```

In [5]:
```
# get the content of each url

soup = []
for i in range(len(res)):
    soup.append(BeautifulSoup(res[i].content, 'lxml'))
    # soup is a list of beautifulsoup objects
```

In [6]:
```
# append each job description to a list

jd = []
for i in range(len(soup)):

# VentureFizz
    if soup[i].find('div',{'class': 'views-field views-field-field-job-description'}):
        jd.append(soup[i].find('div',{'class': 'views-field views-field-field-job-description'}))

# Linkedin
    elif soup[i].find('div',{'class': 'description__text description__text--rich'}):
        jd.append(soup[i].find('div',{'class': 'description__text description__text--rich'}))

# indeed.com
    elif soup[i].find('div',{'class': 'jobsearch-JobComponent icl-u-xs-mt--sm'}):
        jd.append(soup[i].find('div',{'class': 'jobsearch-JobComponent icl-u-xs-mt--sm'}))

# Glassdoor?
    elif soup[i].find(name='div', attrs={'id':'JobDescriptionContainer','class': 'gdGrid tabSection p-st
        jd.append(soup[i].find(name='div', attrs={'id':'JobDescriptionContainer','class': 'gdGrid tabSec

# dice.com
    else:
        jd.append(soup[i].find(name='div', attrs={'id':'jobdescSec','class': 'highlight-black'}))
```

Improvement point:

- skip problematic jobs/lines

In [7]:
```
# split the job descriptions into a list of words
# if the job description can't be read(empty) let me know which post ...
# ... it has probably expired and needs to be ommitted

job_desc = [] # list of all words in the job desc
for i in range(len(jd)):
    try:
        job_desc.append(re.findall(r'[A-z]+', jd[i].text))
    except AttributeError:
        print('error on {}'.format (i))
        job_desc.append(re.findall(r'[A-z]+', jd[i-1].text)) # temp fix: if there is an error append
                                                             # the prior cleared post
                                                            # code will break if two consecutive posts a
```

```
error on 29
error on 44
```

In [8]:
```
# small cap all keywords to be matched with the dictionary of search words

for i in range(len(job_desc)):
    job_desc[i] = [x.lower() for x in job_desc[i]]
```

In [9]:
```
# have the dictionary of key search words ready

with open('./searchwords.txt',"r") as keywords:
    keywords = keywords.readlines() # a list of key words with line breaks
    keywords = [x.lower().replace('\n',"") for x in keywords]
    keywords = [x.lower().replace(' ',"") for x in keywords]
```

In [10]:
```
# match the job description to the dictionary of keywords

keywords_lol = [] # list of (lists of) matching keywords of all jobs

for i in range(len(job_desc)):
    job_desc_keywords = [] # list of matching keywords for each job
    for word in job_desc[i]:
        if word in keywords:
            job_desc_keywords.append(word)
    keywords_lol.append(job_desc_keywords)
```

In [11]:
```
#assign a weight for each keyword

weights = {'cfa':10,'arabic':10,'french':10,'returnship':10,'re-ignite':10,'relaunch':10,
           'mba':5,'python':5,
           'analytics':4,'analysis':4,'analyze':4,'analyzes':4,'analytical':4,'analyzing':4,
           'data':3,'dataset':3,'datasets':3,'finance':3,'financial':3,'financials':3,
           'tableau':2,'sql':2}
```

In [12]:
```
# replace each keyword by its weight equivalent

scores_lol = [] # list of (lists of) job scores

for i in range(len(keywords_lol)):
    job_score = [] # score of each job
    for word in keywords_lol[i]:
        if word in weights.keys():
            job_score.append(weights[word])
        else:
            job_score.append(1)
    scores_lol.append(sum(job_score))
```

In [13]:
```
# create the dataframe
jobs = pd.DataFrame(columns=['link', 'appeal'])
```

In [14]:
```
# clean up the urls
urls = [x.replace('\n',"") for x in urls]
```

In [15]:
```
# fill in the data frame
jobs = jobs.assign(appeal=scores_lol, link = urls).fillna(0)
```

In [16]:
```
# drop duplicate URLs
jobs = jobs.drop_duplicates(subset=['link','appeal'])
```

In [17]:
```
# create another column and mark all jobs as not applied to
jobs['application'] = 'not yet applied'
```

In [18]:
```
# do not abridge the urls
pd.set_option('display.max_colwidth', None)
```

In [19]:
```
# check the DF
jobs.head()
```

Out[19]:

0	(analysis+or+Analyst+or+Analyses+or+Analytical+or+Analyzing+or+Dataset+or+Datasets+or+Financial+or+Financials+or+Decisions+or+Insight+or+Insight
1	(analysis+or+Analyst+or+Analyses+or+Analytical+or+Analyzing+or+Dataset+or+Datasets+or+Financial+or+Financials+or+Decisions+or+Insight+or+Insights-
2	(analysis+or+Analyst+or+Analyses+or+Analytical+or+Analyzing+or+Dataset+or+Datasets+or+Financial+or+Financials+or+Decisions+or+Insight+or+Insigh
3	(analysis+or+analyze+or+analyses+or+analytical+or+analyzing+or+dataset+or+datasets+or+financ
6	

In [20]:
```
# save df to a csv file
jobs.to_csv('./jobs_{}.csv'.format(time.strftime("%Y%m%d")))
```

In [21]:
```
# show top 5 jobs
jobs.sort_values(['application', 'appeal'], ascending=[True, False] ).head()
```

Out[21]:

95	(analysis+or+Analyst+or+Analyses+or+Analytical+or+Analyzing+or+Dataset+or+Datasets+or+Financial+or+Financials+or+Decisions+or+Insight+or+Insigh
20	
64	https://www.linkedin.com/jobs/view/1806883960/?eBP=CwEAAAFxRS9HLsz6PcFWwr wWfLXhdf4zoPyHI0EShRs1HuB0T
73	
121	eBP=CwEAAAFxaj3yHqWAUthrwGOuj5ZVI4JmZphvXkACvOv4seSsPZPyLq6KivWmt2n8hwYGAUna0Iw4dzEKaOnMsSLiRouRnLcRAoEDilnzBCs

In [22]:
```
# count the times each keyword repeats in each job post

d = []
for i in range(len(keywords_lol)):
    cnt = Counter()
    for word in keywords_lol[i]:
        cnt[word] += 1
    d.append(cnt)
    #print(i,cnt)
```

Refresh point

In [23]:
```
# As you check the jobs, mark them as applied to or do not qualify for

jobs.loc[[112,104,105,40,94,93,71,90,0,3,18,20,57,60,64,65,73,95],'application'] = 'applied'
jobs.loc[[122,118,121,77,70,62,14,6,88,84,72,74,89,87,91,82,78,83,1,2,11,12,13,15,19,21,24,26,27,28,31,3
```

In [24]:
```
# sort job by application status, then by score

top_job = jobs.sort_values(['application', 'appeal'], ascending=[True, False]).head(1)
top_job
```

Out[24]:

---

95 (analysis+or+Analyst+or+Analyses+or+Analytical+or+Analyzing+or+Dataset+or+Datasets+or+Financial+or+Financials+or+Decisions+or+Insight+or+Insights

Improvement points:

- show a more user friendly listing (maybe as it shows in the website)
- extract the parts of the text where the words experience, require, etc. is mentioned
- extract the city and job title (all this in a DF, maybe)

In [25]:
```
# prior to applying, search the JD for required skills, experience, etc. # CMD + F
jd[top_job.index[0]].text
```

Out[25]:
```
"This job has expired on IndeedReasons could include: the employer is not accepting applications, is
not actively hiring, or is reviewing applicationsSr. Analyst, FP&ACox Media Group-Atlanta, GA\n
try {\n                    window.mosaic.onMosaicApiReady(function() {\n                        var
zoneId = 'aboveExtractedJobDescription';\n                        var providers = window.mosaic.zone
dProviders[zoneId];\n\n                        if (providers) {\n                            provide
rs.filter(function(p) { return window.mosaic.lazyFns[p]; }).forEach(function(p) {\n
return window.mosaic.api.loadProvider(p);\n                            });\n
}\n                    });\n                } catch (e) {};\n                \n                try
{\n                    window.mosaic.onMosaicApiReady(function() {\n                        var zone
Id = 'aboveFullJobDescription';\n                        var providers = window.mosaic.zonedProvider
s[zoneId];\n\n                        if (providers) {\n                            providers.filter
(function(p) { return window.mosaic.lazyFns[p]; }).forEach(function(p) {\n
return window.mosaic.api.loadProvider(p);\n                            });\n
}\n                    });\n                } catch (e) {};\n                POSITION SUMMARY\nCox
Media Group is seeking a Sr. Analyst, FP&A to join our storied, 122-year-old media organization, now
backed by one of the world's leading private equity firms in Apollo Global Management. We share a be
lief that local media is an essential contributor to a healthy and informed community, and together
we're poised to blaze a new trail, investing in growth to build the future of the industry around ou
r dozens of beloved consumer brands including WSB-TV ABC Channel 2 and B98.5 FM.\nThe Sr. Analyst wi
```

In [26]:
```
# check for keywords in top job ... to customize R/CL
pd.DataFrame.from_dict(d[top_job.index[0]], orient='index').sort_values(0, ascending=False).head(10)
```

Out[26]:

	0
data	13
financial	10
analytical	9
finance	7
analytics	6
planning	6
tableau	4
fp	3
analysis	3
forecast	2

back to refresh point

In [ ]:
```
```