Parsing users and posts

December 12, 2022

0.1 Clean data - posts

0.1.1 Description of the code:

This code gets the data from the XML-download to the computer. As the size of the files increases what my computer can work with at a time, I will read them in line-by-line instead. This is highly time-inefficient, but due to the constrains on the processing power of the computer, it is necessary.

0.1.2 Plan:

- draw 300,000 random users
- parse to make dataframe
- obtain list of posts
- get all posts these users have ever made
- save in csv

0.1.3 Make dataframe out of site - moderation project

From https://ia800107.us.archive.org/27/items/stackexchange/readme.txt:

- **posts**.xml
  - Id
  - PostTypeId
    - 1: Question
    - 2: Answer
  - ParentID (only present if PostTypeId is 2)
  - AcceptedAnswerId (only present if PostTypeId is 1)
  - CreationDate
  - Score
  - ViewCount
  - Body
  - OwnerUserId
  - LastEditorUserId
  - LastEditorDisplayName="Jeff Atwood"
  - LastEditDate="2009-03-05T22:28:34.823"
  - LastActivityDate="2009-03-11T12:51:01.480"
  - CommunityOwnedDate="2009-03-11T12:51:01.480"
  - ClosedDate="2009-03-11T12:51:01.480"
  - Title=

- Tags=
- AnswerCount
- CommentCount
- FavoriteCount

[1]:
```
# Import packages
import random
import pandas as pd
import matplotlib.pyplot as plt
import math
import time # to see how long it takes to run different parts of code
import linecache
import re

users_file = 'D:/Data/Users.xml'

file = 'D:/Data/Posts.xml'
```

0.2 Draw

300,000 random users

Count total number off users, and draw a random subset

[119]:
```
start = time.time()


with open(users_file, 'r', encoding='UTF-8') as f:
    num_users = sum(1 for line in f)
    print('Total lines:', num_users)

end = time.time()
print((end - start)/60)
```

```
Total lines: 17053425
0.7961349685986837
```

[3]:
```
# First, make a seed, to reproduceable list
SEED = ##Insert seed number
random.seed(SEED)


## Next, make a list with random numbers. Draw numbers from (line) 2 (first␣
 ↪lines contains information of parsing-structure),
## to the total number of lines. Draw 2000 - the number of random observations
list_of_random = random.sample(range(2, num_users), 300000)
```

[4]:
```
start = time.time()

# Make a list that can contain the text data, one line as each item
users_dat = []

# Make a counter, to see how the loop is progressing
perc = num_lines/100
list_of_numbers=[]

for i in range(1,101):

list_of_numbers.append(round(i*perc))
i = 0
b = 0

# loop over lines in a file
with open(users_file, 'r', encoding = 'UTF-8') as f:
    for pos, l_num in enumerate(f):
# check if the line number (pos) is specified in the lines (list_of_random)
        if pos in list_of_random:
# If the line is in the list-of-random, then append the lines
            users_dat.append(l_num)
     i = i+1
        if i in list_of_numbers:
            b = b+1
            print(b)



end = time.time()
print((end - start)/60)
```

2518.7492540240287

[6]:
```
#Make a list of variables in data
reputation = []
creation_date = []
display_name = []
location = []
LastAccessDate = []
AboutMe = []
Views = []
UpVotes = []
DownVotes = []
Id = []
EmailHash = []
WebsiteUrl = []
AccountId = []
```

```
# Parse the data and find the variables. If a variable is missing, then assign␣
 ↪the variable "missing"
for i in range(len(users_dat)):
    if "Reputation" in users_dat[i]:
        z = users_dat[i].split('Reputation="')[1].split('"')[0]
        reputation.append(z)
    else:
        reputation.append("missing")
    if "CreationDate" in users_dat[i]:
        z = users_dat[i].split('CreationDate="')[1].split('"')[0]
        creation_date.append(z)
    else:
        creation_date.append("missing")
    if "DisplayName" in users_dat[i]:
        z = users_dat[i].split('DisplayName="')[1].split('"')[0]
        display_name.append(z)
    else:
        display_name.append("missing")
    if '" Location="' in users_dat[i]:
        z = users_dat[i].split('Location="')[1].split('"')[0]
        location.append(z)
    else:
        location.append("missing")
    if "LastAccessDate" in users_dat[i]:
        z = users_dat[i].split('LastAccessDate="')[1].split('"')[0]
        LastAccessDate.append(z)
    else:
        LastAccessDate.append("missing")
    if "AboutMe" in users_dat[i]:
        z = users_dat[i].split('AboutMe="')[1].split('"')[0]
        AboutMe.append(z)
    else:
        AboutMe.append("missing")
    if "Views" in users_dat[i]:
        z = users_dat[i].split('Views="')[1].split('"')[0]
        Views.append(z)
    else:
        Views.append("missing")
    if "UpVotes" in users_dat[i]:
        z = users_dat[i].split('UpVotes="')[1].split('"')[0]
        UpVotes.append(z)
    else:
        UpVotes.append("missing")
    if "UpVotes" in users_dat[i]:
        z = users_dat[i].split('DownVotes="')[1].split('"')[0]
        DownVotes.append(z)
```

```
    else:
        DownVotes.append("missing")
    if "rowId" in users_dat[i]:
        z = users_dat[i].split('Id="')[1].split('"')[0]
        Id.append(z)
    else:
        Id.append("missing")
    if "EmailHash" in users_dat[i]:
        z = users_dat[i].split('EmailHash="')[1].split('"')[0]
        EmailHash.append(z)
    else:
        EmailHash.append("missing")
    if "WebsiteUrl" in users_dat[i]:
        z = users_dat[i].split('WebsiteUrl="')[1].split('"')[0]
        WebsiteUrl.append(z)
    else:
        WebsiteUrl.append("missing")
    if "AccountId" in users_dat[i]:
        z = users_dat[i].split('AccountId="')[1].split('"')[0]
        AccountId.append(z)
    else:
        AccountId.append("missing")



df_user = pd.DataFrame(reputation)

df_user['reputation'] = reputation
df_user['creation'] =  creation_date
df_user['display_name'] = display_name
df_user['location'] = location
df_user['lastaccessdate'] = LastAccessDate
df_user['aboutme'] = AboutMe
df_user['view'] = Views
df_user['upvotes'] = UpVotes
df_user['downvotes'] = DownVotes
df_user['Id'] = Id
df_user['EmailHash'] =  EmailHash
df_user['WebsiteUrl'] = WebsiteUrl
df_user['AccountId'] = AccountId
```

[7]:
```
df_user = df_user.loc[df_user['AccountId'] != "missing"]

list_of_users = df_user["AccountId"].values.tolist()
```

[8]:
```
# Save dataframe w users

df_user.to_csv("users.csv")
```

0.3 Set up to make loop to extract the posts and comments

First - we need to see how many lines there are in total

[2]:
```
## Only run first time (very time-consuming):

## Get number of lines to make a random sample:

#with open(file, 'r', encoding='UTF-8') as f:
#    num_lines = sum(1 for line in f)
#    print('Total lines:', num_lines)
```

```
Total lines: 55513871
```

[9]:
```
# Else - make an abject which holds the total number of lines

num_lines = 55513871
```

Total number of lines is : 55513871

[10]:
```
# Make a list with all usernames in, in the same format as in the big line


list_of_search_strings = []

for i in range(len(list_of_users)):
        list_of_search_strings.append(str('OwnerUserId="' + list_of_users[i] +␣
 ↪'"'))
```

0.4 Make a loop that extracts the posts and comments from the random subset

[11]:
```
# Make a list of numbers equalling fractions of the data. This is to enable one␣
 ↪to see how "far" we are getting

perc = num_lines/100
list_of_numbers=[]

for i in range(1,101):

    list_of_numbers.append(round(i*perc))
```

[12]:
```
start = time.time()

# Make a regex-string, which contains all of the user-numbers randomly drawn
```

```
temp = '(?:% s)' % '|'.join(list_of_search_strings)

# Make a list which can contain all of the posts
list_of_posts = []



# Make i and b == 0, which will be used to count during the loop
i = 0
b = 0

# Open the file and read each line
with open(file, 'r', encoding = 'UTF-8') as f:
    for line in range(num_lines):
        i = i+1
        z = f.readline()
        if  re.search(temp, z):
                list_of_posts.append(z)
        if i in list_of_numbers:
            b = b+1
            print(b)

end = time.time()
print((end - start)/60)
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
```

```
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
3757.6823801080386
```

[18]:
```
## Parse the dataframe

start = time.time()

#Make lists for all variables in the data
Id = []
PostTypeId = []
ParentID = []
AcceptedAnswerId = []
CreationDate = []
Score = []
ViewCount = []
Body = []
OwnerUserId = []
LastEditorUserId = []
LastEditorDisplayName = []
```

```
LastEditDate = []
CommunityOwnedDate = []
ClosedDate = []
Title = []
Tags = []
AnswerCount = []
CommentCount = []
FavoriteCount = []


#Make a loop assigning values to all variables in data. If data is not there,␣
 ↪assign "missing".
for i in range(len(list_of_posts)):
    if "w Id=" in list_of_posts[i]:
        z = list_of_posts[i].split('Id="')[1].split('"')[0]
        Id.append(z)
    else:
        Id.append("missing")
    if "PostTypeId=" in list_of_posts[i]:
        z = list_of_posts[i].split('PostTypeId="')[1].split('"')[0]
        PostTypeId.append(z)
    else:
        PostTypeId.append("missing")
    if '" ParentId="' in list_of_posts[i]:
        z = list_of_posts[i].split('" ParentId="')[1].split('"')[0]
        ParentID.append(z)
    else:
        ParentID.append("missing")
    if '" AcceptedAnswerId=' in list_of_posts[i]:
        z = list_of_posts[i].split('AcceptedAnswerId="')[1].split('"')[0]
        AcceptedAnswerId.append(z)
    else:
        AcceptedAnswerId.append("missing")
    if "CreationDate=" in list_of_posts[i]:
        z = list_of_posts[i].split('CreationDate="')[1].split('"')[0]
        CreationDate.append(z)
    else:
        CreationDate.append("missing=")
    if "Score=" in list_of_posts[i]:
        z = list_of_posts[i].split('Score="')[1].split('"')[0]
        Score.append(z)
    else:
        Score.append("missing")
    if "ViewCount=" in list_of_posts[i]:
        z = list_of_posts[i].split('ViewCount="')[1].split('"')[0]
        ViewCount.append(z)
    else:
```

```
        ViewCount.append("missing")
    if "Body=" in list_of_posts[i]:
        z = list_of_posts[i].split('Body="')[1].split('"')[0]
        Body.append(z)
    else:
        Body.append("missing")
    if "OwnerUserId=" in list_of_posts[i]:
        z = list_of_posts[i].split('OwnerUserId="')[1].split('"')[0]
        OwnerUserId.append(z)
    else:
        OwnerUserId.append("missing")
    if "LastEditorUserId=" in list_of_posts[i]:
        z = list_of_posts[i].split('LastEditorUserId="')[1].split('"')[0]
        LastEditorUserId.append(z)
    else:
        LastEditorUserId.append("missing")
    if "LastEditorDisplayName=" in list_of_posts[i]:
        z = list_of_posts[i].split('LastEditorDisplayName="')[1].split('"')[0]
        LastEditorDisplayName.append(z)
    else:
        LastEditorDisplayName.append("missing")
    if "LastEditDate=" in list_of_posts[i]:
        z = list_of_posts[i].split('LastEditDate="')[1].split('"')[0]
        LastEditDate.append(z)
    else:
        LastEditDate.append("missing")
    if "CommunityOwnedDate=" in list_of_posts[i]:
        z = list_of_posts[i].split('CommunityOwnedDate="')[1].split('"')[0]
        CommunityOwnedDate.append(z)
    else:
        CommunityOwnedDate.append("missing")
    if "ClosedDate=" in list_of_posts[i]:
        z = list_of_posts[i].split('ClosedDate="')[1].split('"')[0]
        ClosedDate.append(z)
    else:
        ClosedDate.append("missing")
    if '" Tags="' in list_of_posts[i]:
        z = list_of_posts[i].split('" Tags="')[1].split('"')[0]
        Tags.append(z)
    else:
        Tags.append("missing")
    if "AnswerCount=" in list_of_posts[i]:
        z = list_of_posts[i].split('AnswerCount="')[1].split('"')[0]
        AnswerCount.append(z)
    else:
        AnswerCount.append("missing")
    if "CommentCount=" in list_of_posts[i]:
```

```
        z = list_of_posts[i].split('CommentCount="')[1].split('"')[0]
        CommentCount.append(z)
    else:
        CommentCount.append("missing")
    if '" FavoriteCount=' in list_of_posts[i]:
        z = list_of_posts[i].split('FavoriteCount="')[1].split('"')[0]
        FavoriteCount.append(z)
    else:
        FavoriteCount.append("missing")
    if '" Title="' in list_of_posts[i]:
        z = list_of_posts[i].split('Title="')[1].split('"')[0]
        Title.append(z)
    else:
        Title.append("missing")



df = pd.DataFrame(Id)

df['PostTypeId'] = PostTypeId
df['ParentID'] = ParentID
df['AcceptedAnswerId'] = AcceptedAnswerId
df['CreationDate'] = CreationDate
df['Score'] = Score
df['ViewCount'] = ViewCount
df['Body'] = Body
df['OwnerUserId'] = OwnerUserId
df['LastEditorUserId'] = LastEditorUserId
df['LastEditorDisplayName'] = LastEditorDisplayName
df['LastEditDate'] = LastEditDate
df['CommunityOwnedDate'] = CommunityOwnedDate
df['ClosedDate'] = ClosedDate
df['Title'] = Title
df['Tags'] = Tags
df['AnswerCount'] = AnswerCount
df['CommentCount'] = CommentCount
df['FavoriteCount'] = FavoriteCount


end = time.time()
print((end - start)/60)
```

```
1.0194374879201253
```

[19]:
```
## Save csv w dataframe

df.to_csv("posts.csv")
```