Lecture 17: Big Data Wrangling, OnDemand on CRC

LING 1340/2340: Data Science for Linguists

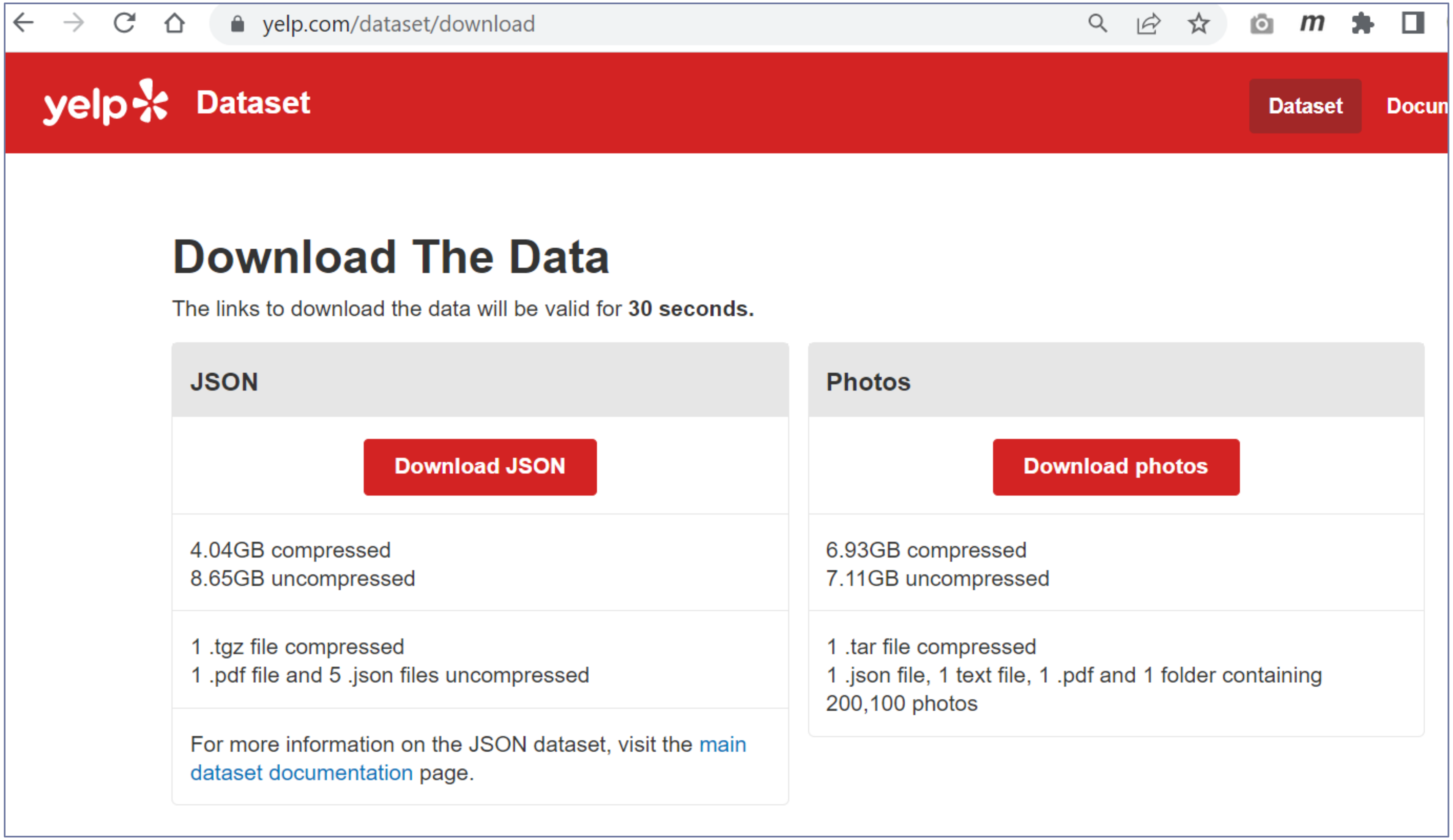
Na-Rae Han

Objectives

- Big data considerations
  - Explore on command line
  - Making Python code more efficient
- OnDemand platform & JNB at CRC (GUI!)
  - Clustering, topic modelling

The Yelp Dataset Challenge

- https://www.yelp.com/dataset



Download The Data

The links to download the data will be valid for **30 seconds.**

JSON

4.04GB compressed
8.65GB uncompressed

1 .tgz file compressed
1 .pdf file and 5 .json files uncompressed

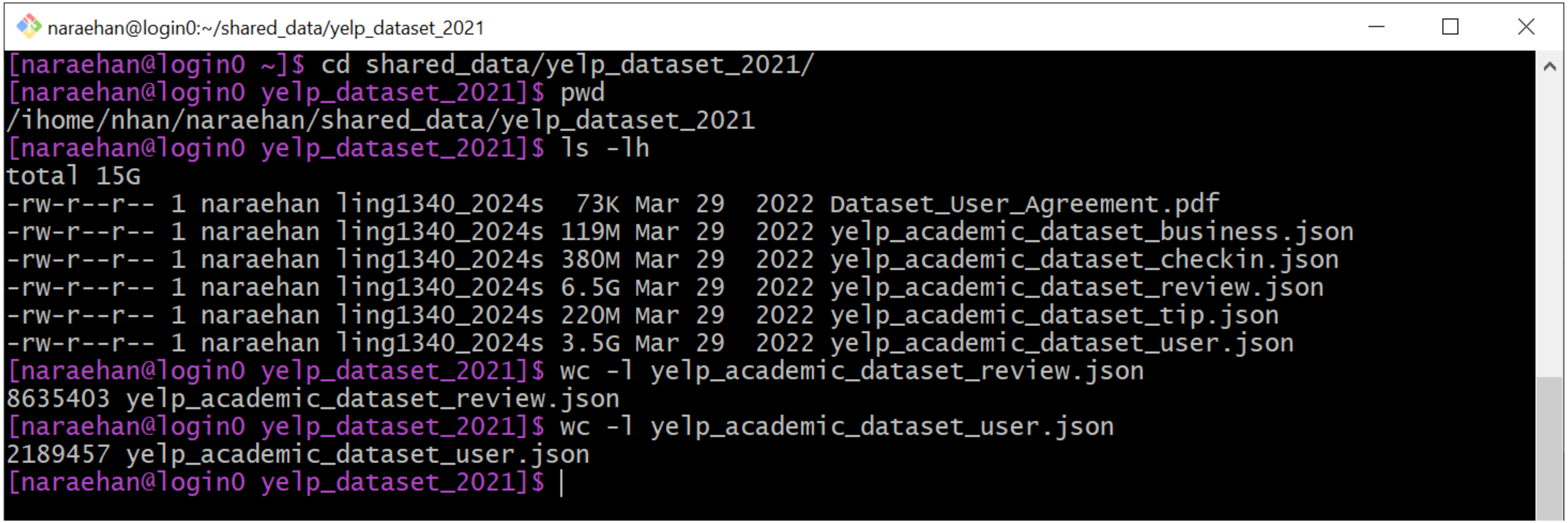
For more information on the JSON dataset, visit the main dataset documentation page.

Photos

6.93GB compressed
7.11GB uncompressed

1 .tar file compressed
1 .json file, 1 text file, 1 .pdf and 1 folder containing 200,100 photos

Working with big data files



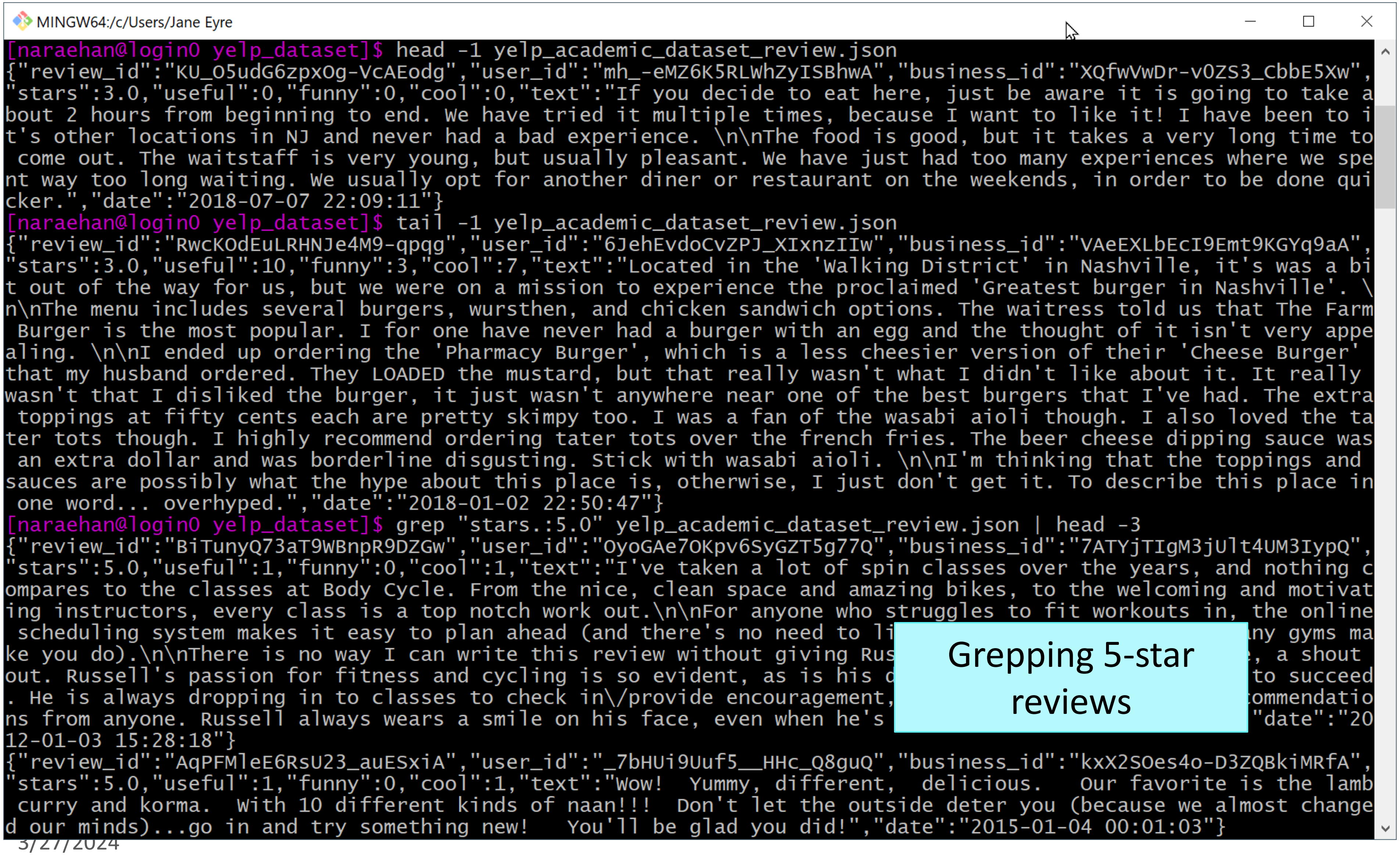
naraehan@login0:~/shared_data/yelp_dataset_2021

```
[naraehan@login0 ~]$ cd shared_data/yelp_dataset_2021/
[naraehan@login0 yelp_dataset_2021]$ pwd
/ihome/nhan/naraehan/shared_data/yelp_dataset_2021
[naraehan@login0 yelp_dataset_2021]$ ls -lh
total 15G
-rw-r--r-- 1 naraehan ling1340_2024s  73K Mar 29  2022 Dataset_User_Agreement.pdf
-rw-r--r-- 1 naraehan ling1340_2024s 119M Mar 29  2022 yelp_academic_dataset_business.json
-rw-r--r-- 1 naraehan ling1340_2024s 380M Mar 29  2022 yelp_academic_dataset_checkin.json
-rw-r--r-- 1 naraehan ling1340_2024s 6.5G Mar 29  2022 yelp_academic_dataset_review.json
-rw-r--r-- 1 naraehan ling1340_2024s 220M Mar 29  2022 yelp_academic_dataset_tip.json
-rw-r--r-- 1 naraehan ling1340_2024s 3.5G Mar 29  2022 yelp_academic_dataset_user.json
[naraehan@login0 yelp_dataset_2021]$ wc -l yelp_academic_dataset_review.json
8635403 yelp_academic_dataset_review.json
[naraehan@login0 yelp_dataset_2021]$ wc -l yelp_academic_dataset_user.json
2189457 yelp_academic_dataset_user.json
[naraehan@login0 yelp_dataset_2021]$ |
```

- Each file is in JSON format, and they are huge:
  - review.json is 6.5GB with 8.6 million records
  - user.json is 3.5GB with 2 million records
  - ← Too big to open in most text editors (Notepad++ couldn't.)
  - ← How to explore them?

    In command line. head/tail, grep and regular expression-based searching.

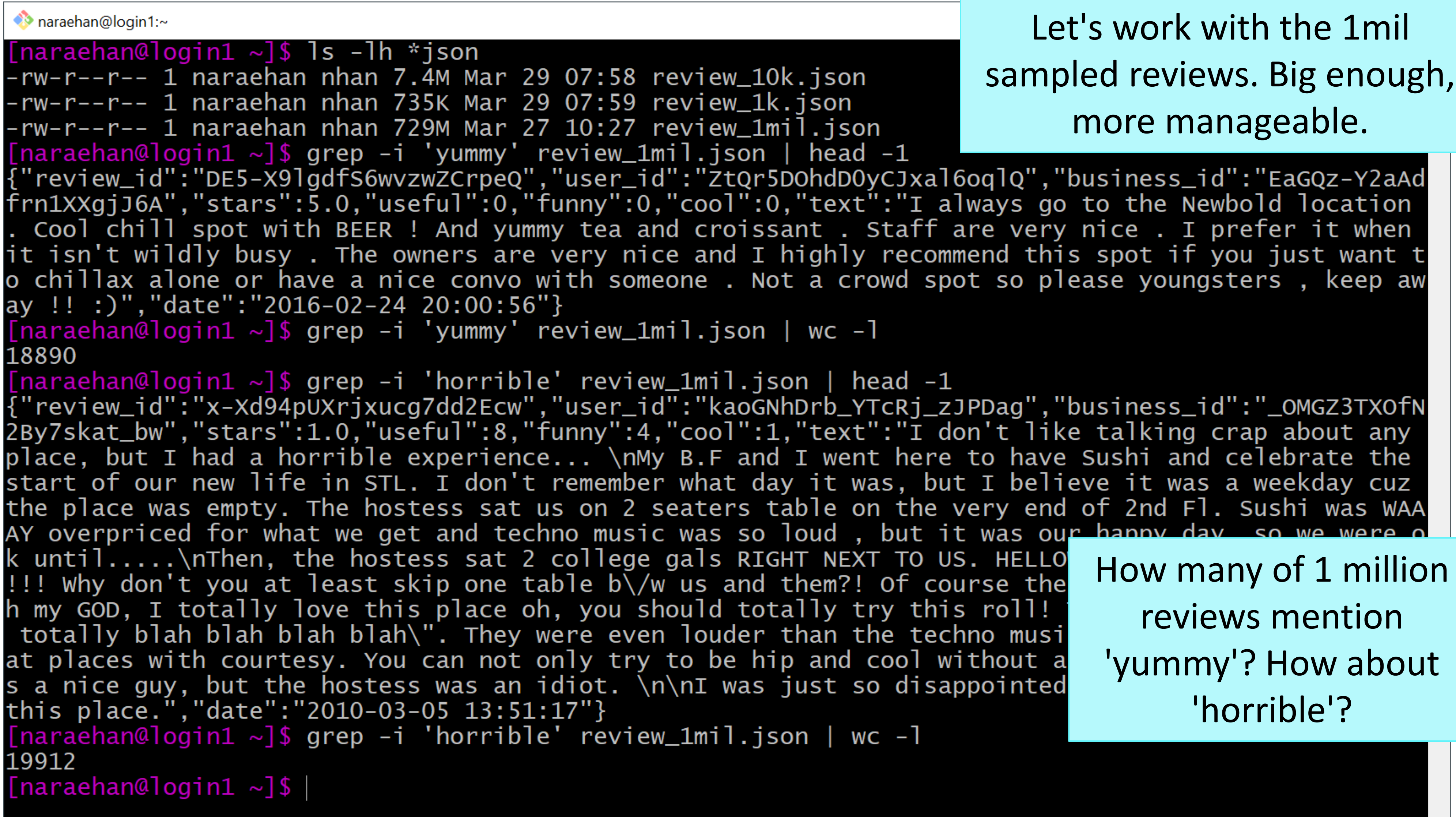
Command line exploration: 5-star reviews?



```
[naraehan@login0 yelp_dataset]$ head -1 yelp_academic_dataset_review.json
{"review_id":"KU_O5udG6zpxOg-VcAEodg","user_id":"mh_-eMZ6K5RLWhZyISBhwA","business_id":"XQfwVwDr-v0ZS3_CbbE5Xw",
"stars":3.0,"useful":0,"funny":0,"cool":0,"text":"If you decide to eat here, just be aware it is going to take a
bout 2 hours from beginning to end. We have tried it multiple times, because I want to like it! I have been to i
t's other locations in NJ and never had a bad experience. \n\nThe food is good, but it takes a very long time to
 come out. The waitstaff is very young, but usually pleasant. We have just had too many experiences where we spe
nt way too long waiting. We usually opt for another diner or restaurant on the weekends, in order to be done qui
cker.","date":"2018-07-07 22:09:11"}
[naraehan@login0 yelp_dataset]$ tail -1 yelp_academic_dataset_review.json
{"review_id":"RwcKOdEuLRHNJe4M9-qpqg","user_id":"6JehEvdoCvZPJ_XIxnzIIw","business_id":"VAeEXLbEcI9Emt9KGYq9aA",
"stars":3.0,"useful":10,"funny":3,"cool":7,"text":"Located in the 'Walking District' in Nashville, it's was a bi
t out of the way for us, but we were on a mission to experience the proclaimed 'Greatest burger in Nashville'. \
n\nThe menu includes several burgers, wursthen, and chicken sandwich options. The waitress told us that The Farm
 Burger is the most popular. I for one have never had a burger with an egg and the thought of it isn't very appe
aling. \n\nI ended up ordering the 'Pharmacy Burger', which is a less cheesier version of their 'Cheese Burger'
that my husband ordered. They LOADED the mustard, but that really wasn't what I didn't like about it. It really
wasn't that I disliked the burger, it just wasn't anywhere near one of the best burgers that I've had. The extra
 toppings at fifty cents each are pretty skimpy too. I was a fan of the wasabi aioli though. I also loved the ta
ter tots though. I highly recommend ordering tater tots over the french fries. The beer cheese dipping sauce was
 an extra dollar and was borderline disgusting. Stick with wasabi aioli. \n\nI'm thinking that the toppings and
sauces are possibly what the hype about this place is, otherwise, I just don't get it. To describe this place in
 one word... overhyped.","date":"2018-01-02 22:50:47"}
[naraehan@login0 yelp_dataset]$ grep "stars.:5.0" yelp_academic_dataset_review.json | head -3
{"review_id":"BiTunyQ73aT9WBnpR9DZGw","user_id":"OyoGAe7OKpv6SyGZT5g77Q","business_id":"7ATYjTIgM3jUlt4UM3IypQ",
"stars":5.0,"useful":1,"funny":0,"cool":1,"text":"I've taken a lot of spin classes over the years, and nothing c
ompares to the classes at Body Cycle. From the nice, clean space and amazing bikes, to the welcoming and motivat
ing instructors, every class is a top notch work out.\n\nFor anyone who struggles to fit workouts in, the online
 scheduling system makes it easy to plan ahead (and there's no need to li                    ny gyms ma
ke you do).\n\nThere is no way I can write this review without giving Rus                    , a shout
out. Russell's passion for fitness and cycling is so evident, as is his d                    to succeed
. He is always dropping in to classes to check in\/provide encouragement,                    ommendatio
ns from anyone. Russell always wears a smile on his face, even when he's                     "date":"20
12-01-03 15:28:18"}
{"review_id":"AqPFMleE6RsU23_auESxiA","user_id":"_7bHUi9Uuf5__HHc_Q8guQ","business_id":"kxX2SOes4o-D3ZQBkiMRfA",
"stars":5.0,"useful":1,"funny":0,"cool":1,"text":"Wow!  Yummy, different,  delicious.   Our favorite is the lamb
 curry and korma.  With 10 different kinds of naan!!!  Don't let the outside deter you (because we almost change
d our minds)...go in and try something new!   You'll be glad you did!","date":"2015-01-04 00:01:03"}
```

Grepping 5-star reviews

Command line exploration: 'yummy' vs. 'horrible'

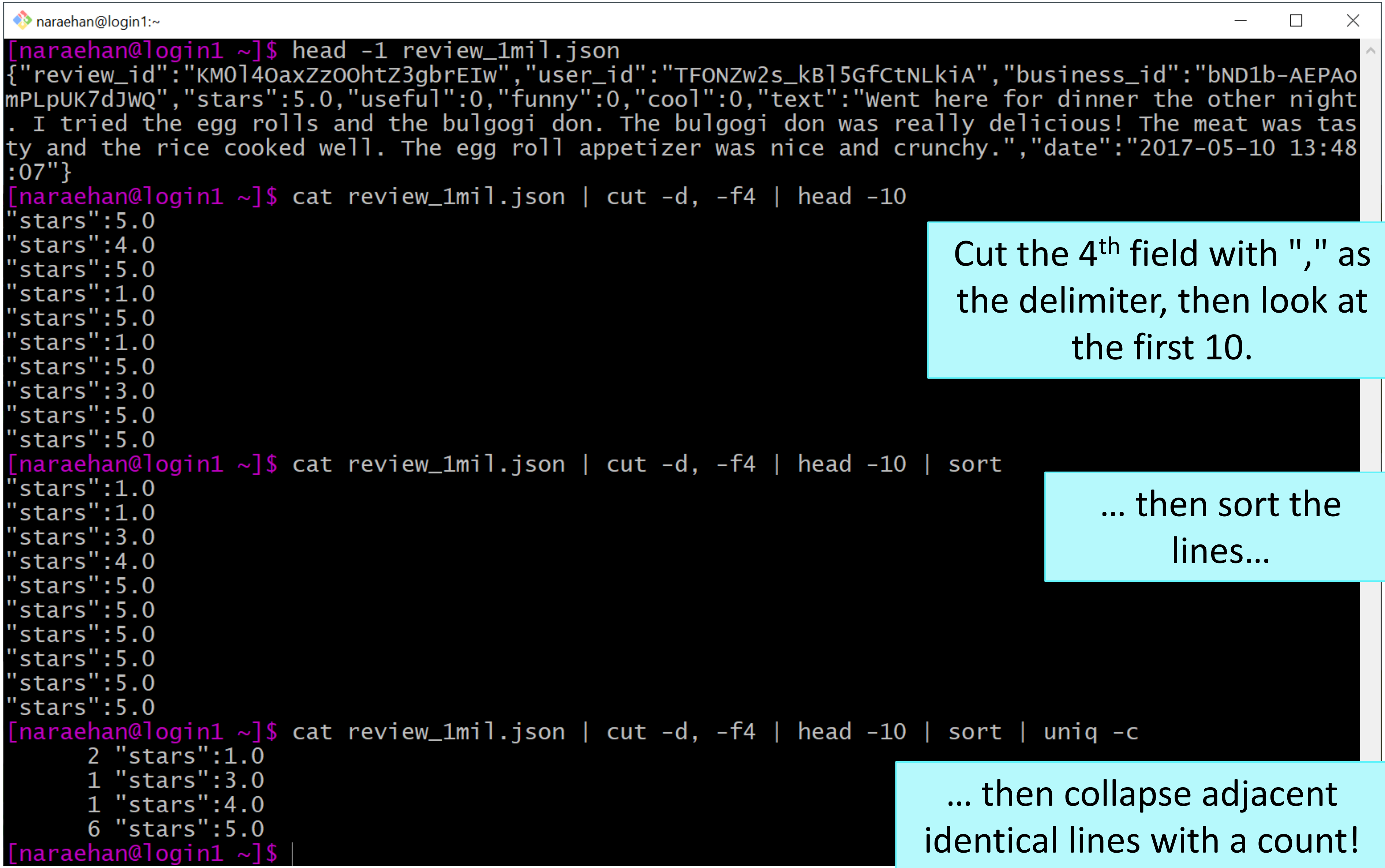


```
[naraehan@login1 ~]$ ls -lh *json
-rw-r--r-- 1 naraehan nhan 7.4M Mar 29 07:58 review_10k.json
-rw-r--r-- 1 naraehan nhan 735K Mar 29 07:59 review_1k.json
-rw-r--r-- 1 naraehan nhan 729M Mar 27 10:27 review_1mil.json
[naraehan@login1 ~]$ grep -i 'yummy' review_1mil.json | head -1
{"review_id":"DE5-X9lgdfS6wvzwZCrpeQ","user_id":"ZtQr5DOhdDOyCJxal6oqlQ","business_id":"EaGQz-Y2aAd
frn1XXgjJ6A","stars":5.0,"useful":0,"funny":0,"cool":0,"text":"I always go to the Newbold location
. Cool chill spot with BEER ! And yummy tea and croissant . Staff are very nice . I prefer it when
it isn't wildly busy . The owners are very nice and I highly recommend this spot if you just want t
o chillax alone or have a nice convo with someone . Not a crowd spot so please youngsters , keep aw
ay !! :)","date":"2016-02-24 20:00:56"}
[naraehan@login1 ~]$ grep -i 'yummy' review_1mil.json | wc -l
18890
[naraehan@login1 ~]$ grep -i 'horrible' review_1mil.json | head -1
{"review_id":"x-Xd94pUXrjxucg7dd2Ecw","user_id":"kaoGNhDrb_YTcRj_zJPDag","business_id":"_OMGZ3TXOfN
2By7skat_bw","stars":1.0,"useful":8,"funny":4,"cool":1,"text":"I don't like talking crap about any
place, but I had a horrible experience... \nMy B.F and I went here to have Sushi and celebrate the
start of our new life in STL. I don't remember what day it was, but I believe it was a weekday cuz
the place was empty. The hostess sat us on 2 seaters table on the very end of 2nd Fl. Sushi was WAA
AY overpriced for what we get and techno music was so loud , but it was our happy day , so we were o
k until.....\nThen, the hostess sat 2 college gals RIGHT NEXT TO US. HELLO
!!! Why don't you at least skip one table b\/w us and them?! Of course the
h my GOD, I totally love this place oh, you should totally try this roll!
 totally blah blah blah blah\". They were even louder than the techno musi
at places with courtesy. You can not only try to be hip and cool without a
s a nice guy, but the hostess was an idiot. \n\nI was just so disappointed
this place.","date":"2010-03-05 13:51:17"}
[naraehan@login1 ~]$ grep -i 'horrible' review_1mil.json | wc -l
19912
[naraehan@login1 ~]$ |
```

Change of venue!
Let's work with the 1mil sampled reviews. Big enough, more manageable.

How many of 1 million reviews mention 'yummy'? How about 'horrible'?

A quick look at "Stars" distribution



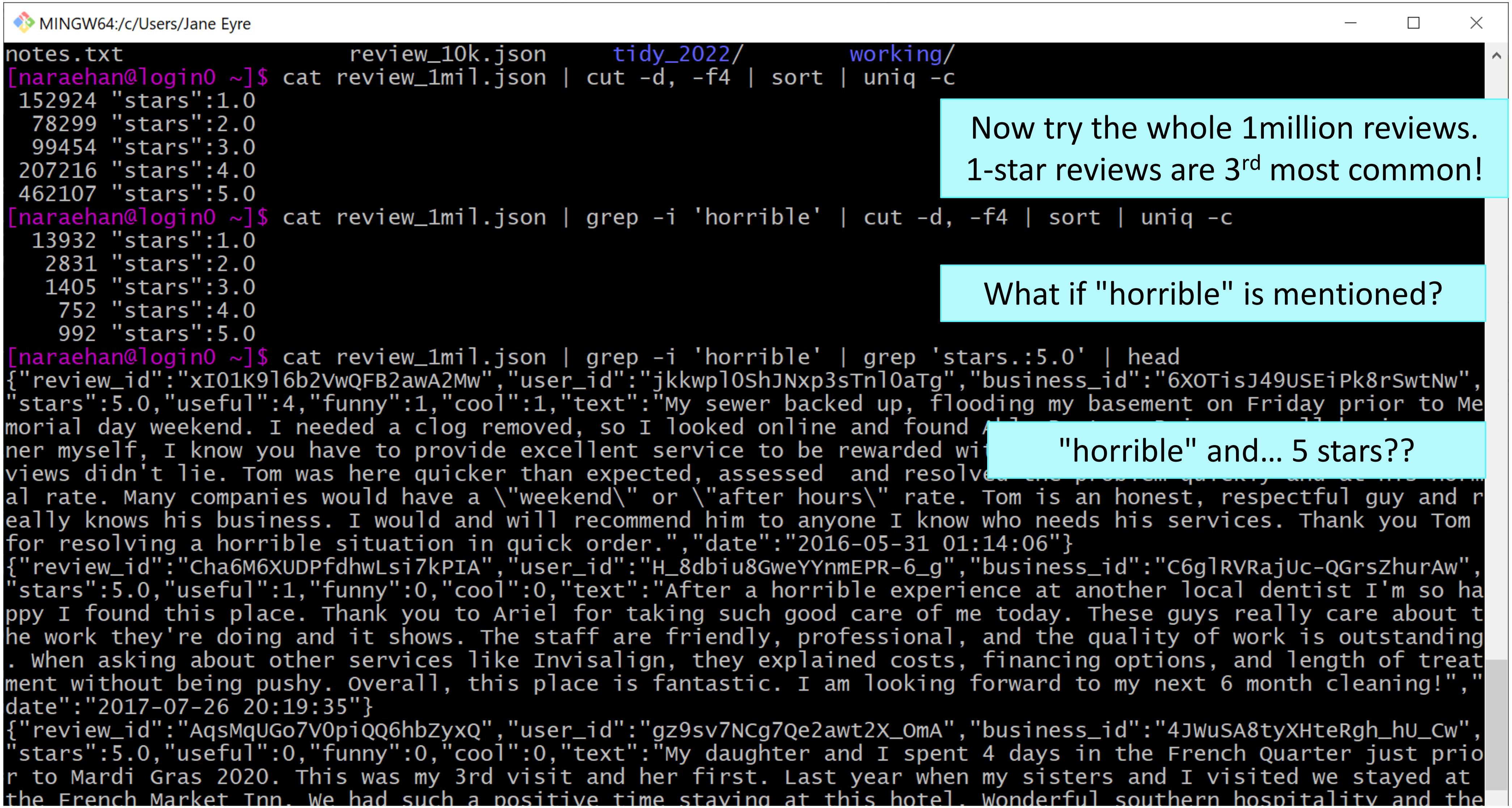
```
[naraehan@login1 ~]$ head -1 review_1mil.json
{"review_id":"KM0l4OaxZzOohtZ3gbrEIw","user_id":"TFONZw2s_kBl5GfCtNLkiA","business_id":"bND1b-AEPAo
mPLpUK7dJWQ","stars":5.0,"useful":0,"funny":0,"cool":0,"text":"Went here for dinner the other night
. I tried the egg rolls and the bulgogi don. The bulgogi don was really delicious! The meat was tas
ty and the rice cooked well. The egg roll appetizer was nice and crunchy.","date":"2017-05-10 13:48
:07"}
[naraehan@login1 ~]$ cat review_1mil.json | cut -d, -f4 | head -10
"stars":5.0
"stars":4.0
"stars":5.0
"stars":1.0
"stars":5.0
"stars":1.0
"stars":5.0
"stars":3.0
"stars":5.0
"stars":5.0
[naraehan@login1 ~]$ cat review_1mil.json | cut -d, -f4 | head -10 | sort
"stars":1.0
"stars":1.0
"stars":3.0
"stars":4.0
"stars":5.0
"stars":5.0
"stars":5.0
"stars":5.0
"stars":5.0
"stars":5.0
[naraehan@login1 ~]$ cat review_1mil.json | cut -d, -f4 | head -10 | sort | uniq -c
      2 "stars":1.0
      1 "stars":3.0
      1 "stars":4.0
      6 "stars":5.0
[naraehan@login1 ~]$ |
```

Cut the 4th field with "," as the delimiter, then look at the first 10.

... then sort the lines...

... then collapse adjacent identical lines with a count!

Which "Stars" most common? With 'horrible'?



```
notes.txt                 review_10k.json     tidy_2022/         working/
[naraehan@login0 ~]$ cat review_1mil.json | cut -d, -f4 | sort | uniq -c
 152924 "stars":1.0
  78299 "stars":2.0
  99454 "stars":3.0
 207216 "stars":4.0
 462107 "stars":5.0
[naraehan@login0 ~]$ cat review_1mil.json | grep -i 'horrible' | cut -d, -f4 | sort | uniq -c
  13932 "stars":1.0
   2831 "stars":2.0
   1405 "stars":3.0
    752 "stars":4.0
    992 "stars":5.0
[naraehan@login0 ~]$ cat review_1mil.json | grep -i 'horrible' | grep 'stars.:5.0' | head
{"review_id":"xI01K9l6b2VwQFB2awA2Mw","user_id":"jkkwpl0ShJNxp3sTnl0aTg","business_id":"6XOTisJ49USEiPk8rSwtNw",
"stars":5.0,"useful":4,"funny":1,"cool":1,"text":"My sewer backed up, flooding my basement on Friday prior to Me
morial day weekend. I needed a clog removed, so I looked online and found A
ner myself, I know you have to provide excellent service to be rewarded wi
views didn't lie. Tom was here quicker than expected, assessed  and resolve
al rate. Many companies would have a \"weekend\" or \"after hours\" rate. Tom is an honest, respectful guy and r
eally knows his business. I would and will recommend him to anyone I know who needs his services. Thank you Tom
for resolving a horrible situation in quick order.","date":"2016-05-31 01:14:06"}
{"review_id":"Cha6M6XUDPfdhwLsi7kPIA","user_id":"H_8dbiu8GweYYnmEPR-6_g","business_id":"C6glRVRajUc-QGrsZhurAw",
"stars":5.0,"useful":1,"funny":0,"cool":0,"text":"After a horrible experience at another local dentist I'm so ha
ppy I found this place. Thank you to Ariel for taking such good care of me today. These guys really care about t
he work they're doing and it shows. The staff are friendly, professional, and the quality of work is outstanding
. When asking about other services like Invisalign, they explained costs, financing options, and length of treat
ment without being pushy. Overall, this place is fantastic. I am looking forward to my next 6 month cleaning!","
date":"2017-07-26 20:19:35"}
{"review_id":"AqsMqUGo7VOpiQQ6hbZyxQ","user_id":"gz9sv7NCg7Qe2awt2X_OmA","business_id":"4JWuSA8tyXHteRgh_hU_Cw",
"stars":5.0,"useful":0,"funny":0,"cool":0,"text":"My daughter and I spent 4 days in the French Quarter just prio
r to Mardi Gras 2020. This was my 3rd visit and her first. Last year when my sisters and I visited we stayed at
the French Market Inn. We had such a positive time staying at this hotel. Wonderful southern hospitality and the
```

Now try the whole 1million reviews. 1-star reviews are 3rd most common!

What if "horrible" is mentioned?

"horrible" and... 5 stars??

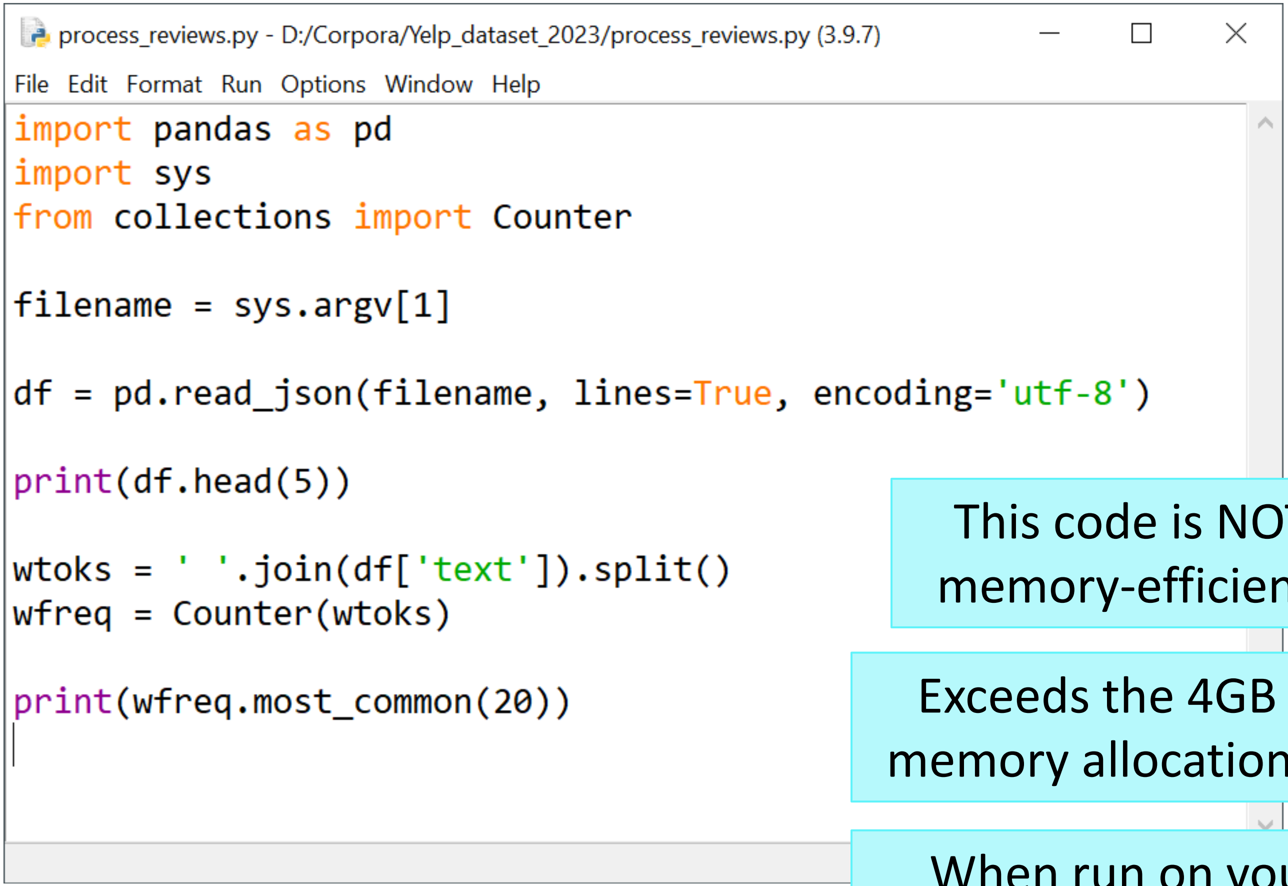
Opening + processing big files

- How much resource does it take to process review.json file (6.5GB)?

There's 6.5 GB

Another 5~ GB

Not as big


process_reviews.py - D:/Corpora/Yelp_dataset_2023/process_reviews.py (3.9.7)
File Edit Format Run Options Window Help

```
import pandas as pd
import sys
from collections import Counter

filename = sys.argv[1]

df = pd.read_json(filename, lines=True, encoding='utf-8')

print(df.head(5))

wtoks = ' '.join(df['text']).split()
wfreq = Counter(wtoks)

print(wfreq.most_common(20))
```

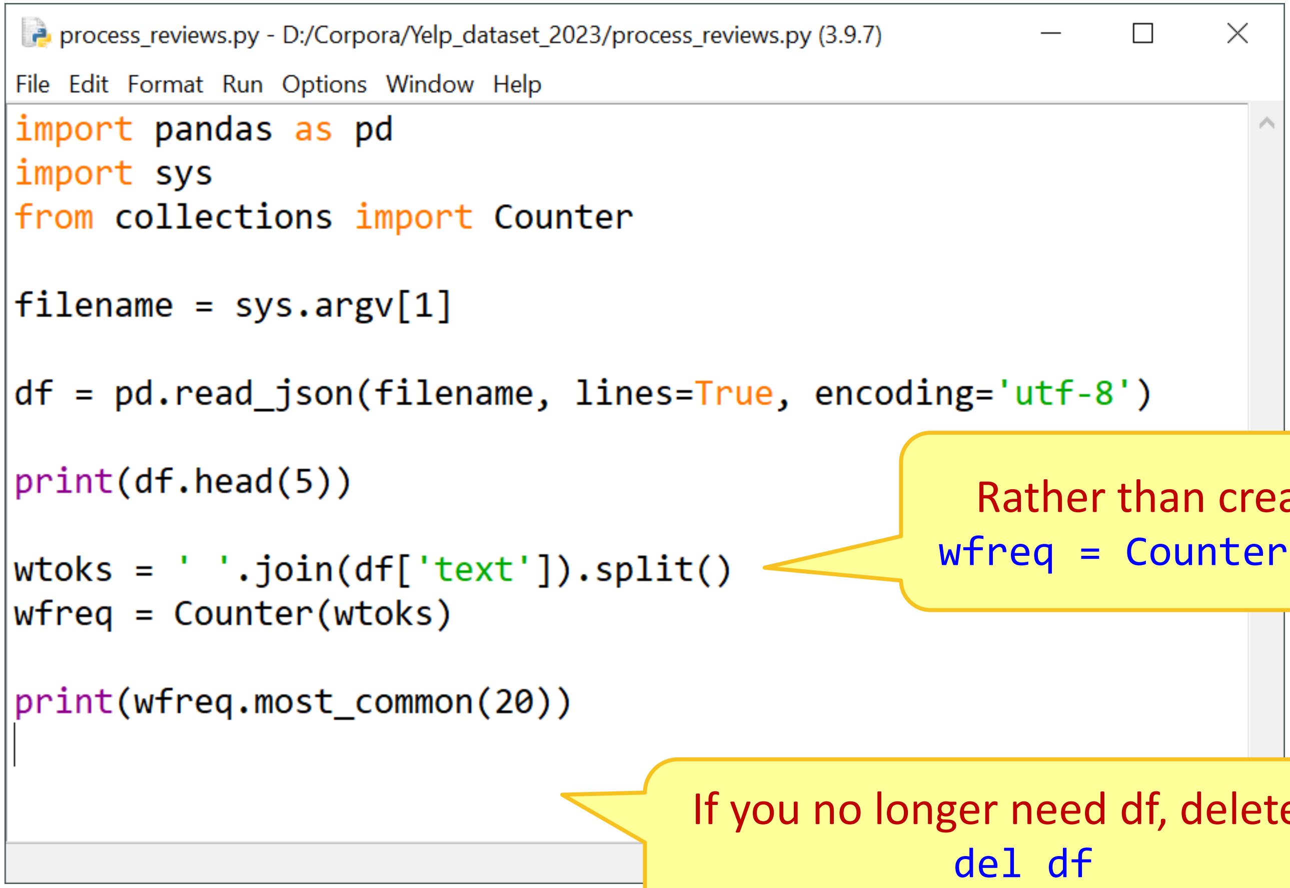
This code is NOT memory-efficient.

Exceeds the 4GB default memory allocation on CRC.

When run on your own laptop, script may crash citing "MemoryError"

Big objects: avoid creating, manually delete

- Try avoiding making big data objects in the first place.
- Manually free up memory by deleting objects when done using. (Advanced users only!)


process_reviews.py - D:/Corpora/Yelp_dataset_2023/process_reviews.py (3.9.7)
File Edit Format Run Options Window Help

```
import pandas as pd
import sys
from collections import Counter

filename = sys.argv[1]

df = pd.read_json(filename, lines=True, encoding='utf-8')

print(df.head(5))

wtoks = ' '.join(df['text']).split()
wfreq = Counter(wtoks)

print(wfreq.most_common(20))
```

Rather than creating wtoks AND wfreq, you could:

```
wfreq = Counter(' '.join(df['text']).split())
```

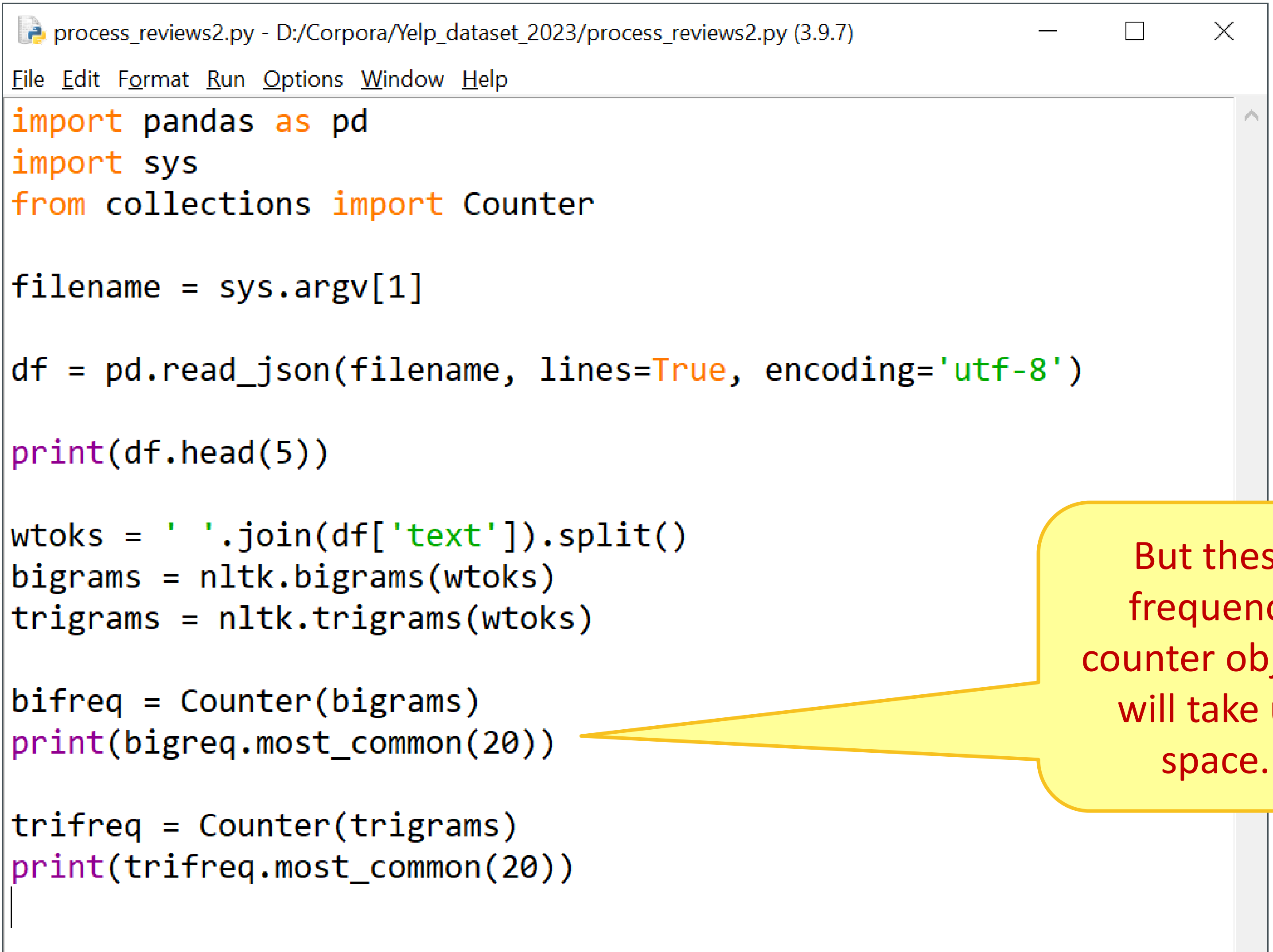
If you no longer need df, delete it:

```
del df
gc.collect()
```

Garbage collection: Python takes care of some memory management on its own.

Memory consideration

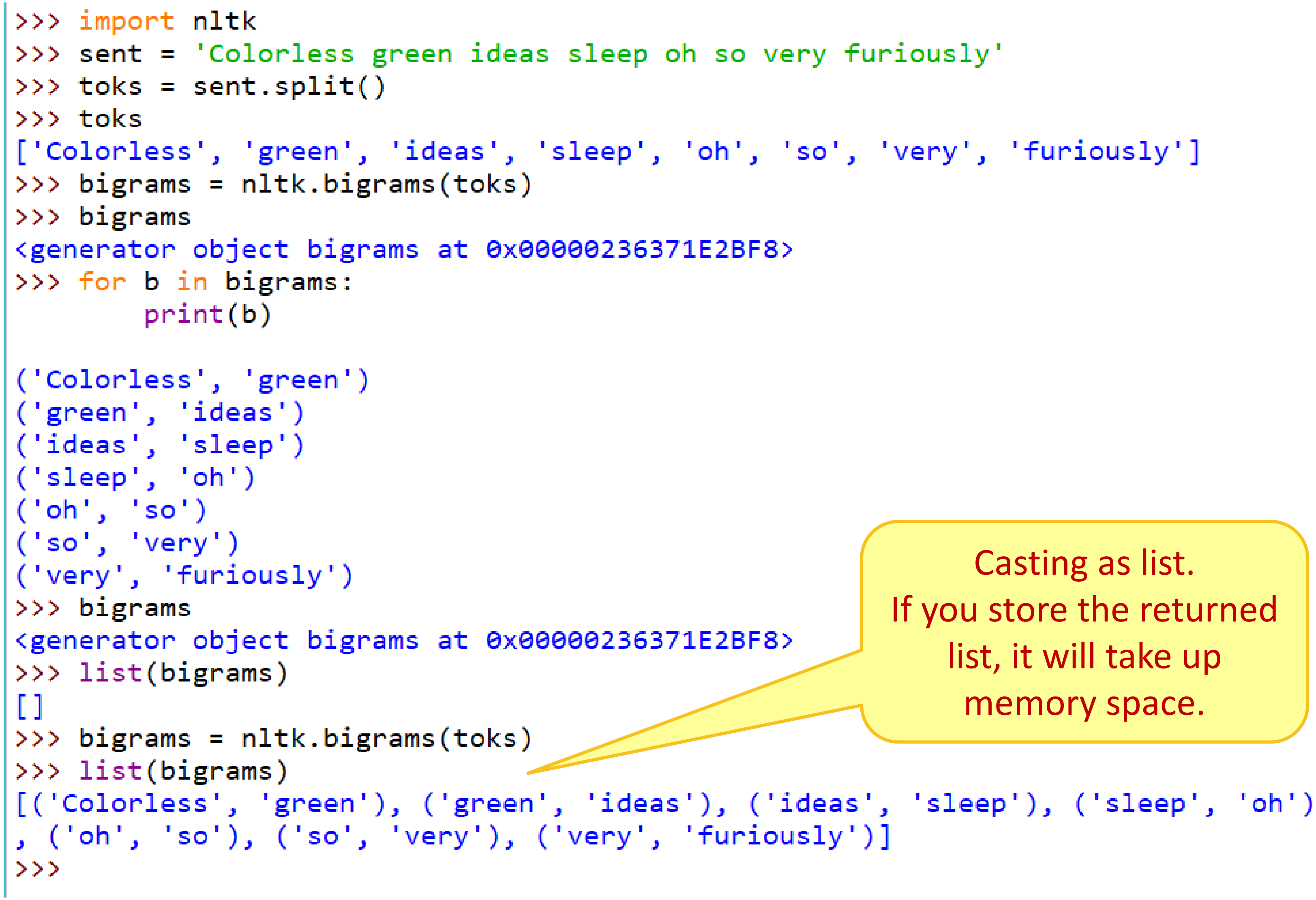
- How much space needed for bigrams? Trigrams?


process_reviews2.py - D:/Corpora/Yelp_dataset_2023/process_reviews2.py (3.9.7)
File Edit Format Run Options Window Help

```
import pandas as pd
import sys
from collections import Counter

filename = sys.argv[1]

df = pd.read_json(filename, lines=True, encoding='utf-8')

print(df.head(5))

wtoks = ' '.join(df['text']).split()
bigrams = nltk.bigrams(wtoks)
trigrams = nltk.trigrams(wtoks)

bifreq = Counter(bigrams)
print(bigreq.most_common(20))

trifreq = Counter(trigrams)
print(trifreq.most_common(20))
```

Good news! These are built as *generator* objects and take up almost no space.

But these frequency counter objects will take up space.



```
>>> import nltk
>>> sent = 'Colorless green ideas sleep oh so very furiously'
>>> toks = sent.split()
>>> toks
['Colorless', 'green', 'ideas', 'sleep', 'oh', 'so', 'very', 'furiously']
>>> bigrams = nltk.bigrams(toks)
>>> bigrams
<generator object bigrams at 0x00000236371E2BF8>
>>> for b in bigrams:
        print(b)

('Colorless', 'green')
('green', 'ideas')
('ideas', 'sleep')
('sleep', 'oh')
('oh', 'so')
('so', 'very')
('very', 'furiously')
>>> bigrams
<generator object bigrams at 0x00000236371E2BF8>
>>> list(bigrams)
[]
>>> bigrams = nltk.bigrams(toks)
>>> list(bigrams)
[('Colorless', 'green'), ('green', 'ideas'), ('ideas', 'sleep'), ('sleep', 'oh')
, ('oh', 'so'), ('so', 'very'), ('very', 'furiously')]
>>>
```

Generator type objects take up little memory space; meant to be used in a loop-like environment.

Content has been exhausted

Casting as list.
If you store the returned list, it will take up memory space.

File opening & closing methods

```python
f = open('review.json')
lines = f.readlines()
for l in lines:
    if 'horrible' in l:
        print(l)
f.close()
```

```python
f = open('review.json')
for l in f:
    if 'horrible' in l:
        print(l)
f.close()
```

```python
lines = open('review.json').readlines()
for l in lines :
    if 'horrible' in l:
        print(l)
```

Python will close up this file handle.

```python
with open('review.json') as f:
    for l in f:
        if 'horrible' in l:
            print(l)
```

No need to close f later. Some folks swear by using `with`.

Which methods are memory-efficient?

Handling files in chunks

```
f = open('review.json')
lines1 = f.readlines(1000000000)
lines2 = f.readlines(1000000000)
lines3 = f.readlines(1000000000)
lines4 = f.readlines(1000000000)
lines5 = f.readlines()
f.close()
```

Optional # of bytes to read.
(When used like this without a loop, offers no memory advantage.)

Generator object: takes up 0 space

```
dfs = pd.read_json('review.json', lines=True, chunksize=10000, encoding='utf8')

wfreq = Counter()

for df in dfs:
    wtoks = ' '.join(df['text']).split()
    temp = Counter(wtoks)
    wfreq.update(temp)

print(wfreq.most_common(20))
```

`chunksize` optional parameter in pandas' `read_json` method reads in 10,000 lines at a time…

then, iterate through each small df.

Memory-efficient!
This code uses only **290MB** of memory!

Pandas vs. large data: tips

- "Why and How to Use Pandas with Large (but not big) Data"
  - https://towardsdatascience.com/why-and-how-to-use-pandas-with-large-data-9594dda2ea4c

1. Read CSV file data in chunk size
2. Filter out unimportant columns in DF to save memory
3. Change `dtypes` for columns
   - float64 takes up more space than float32.

Vectorizing and training in chunks

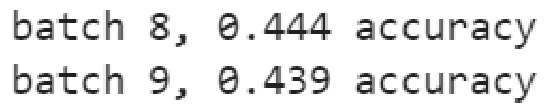
```
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import HashingVectorizer
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)

filename = 'review_10k.json'
length = 10000
chunk_size = 1000
chunks = length/chunk_size

df_chunks = pd.read_json(filename, lines=True, chunksize=chunk_size, encoding="utf-8")

clf = MultinomialNB()
vectorizer = HashingVectorizer(alternate_sign=False)

for i, df in enumerate(df_chunks):
    if i < 0.8 * chunks:
        clf.partial_fit(vectorizer.transform(df['text']), df['stars'], classes=[1,2,3,4,5])
    else:
        pred = clf.predict(vectorizer.transform(df['text']))
        print('batch {}, {} accuracy'.format(i, np.mean(pred == df['stars'])))
```

If vectorizer/ML model depends only on individual row of data, it can be implemented in chunks.

(Caveat: TF-IDF vectorizer and most ML models can't.)

Hashing vectorizer **skips the IDF part** of TF/IDF, can be implemented in chunks!

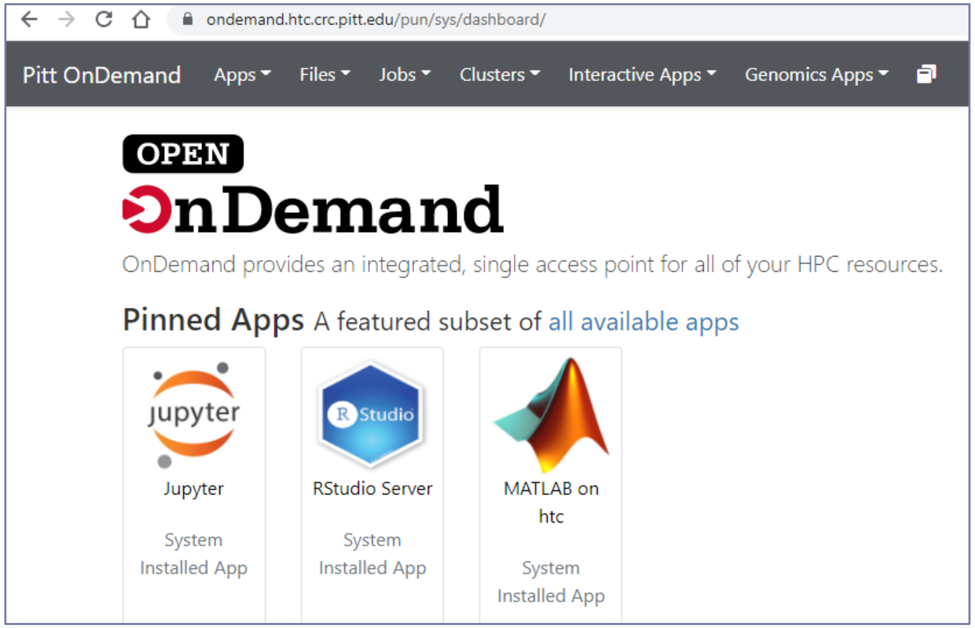
NB classifier can be trained in partial bits!



```
batch 8, 0.444 accuracy
batch 9, 0.439 accuracy
```

OnDemand on CRC!

- Browser-based gateway to CRC resources!
  - https://ondemand.htc.crc.pitt.edu/
- Jupyter Notebook (Lab) etc. are available
- Help documentation:
  - https://crc-pages.pitt.edu/user-manual/web-portals/open-ondemand/

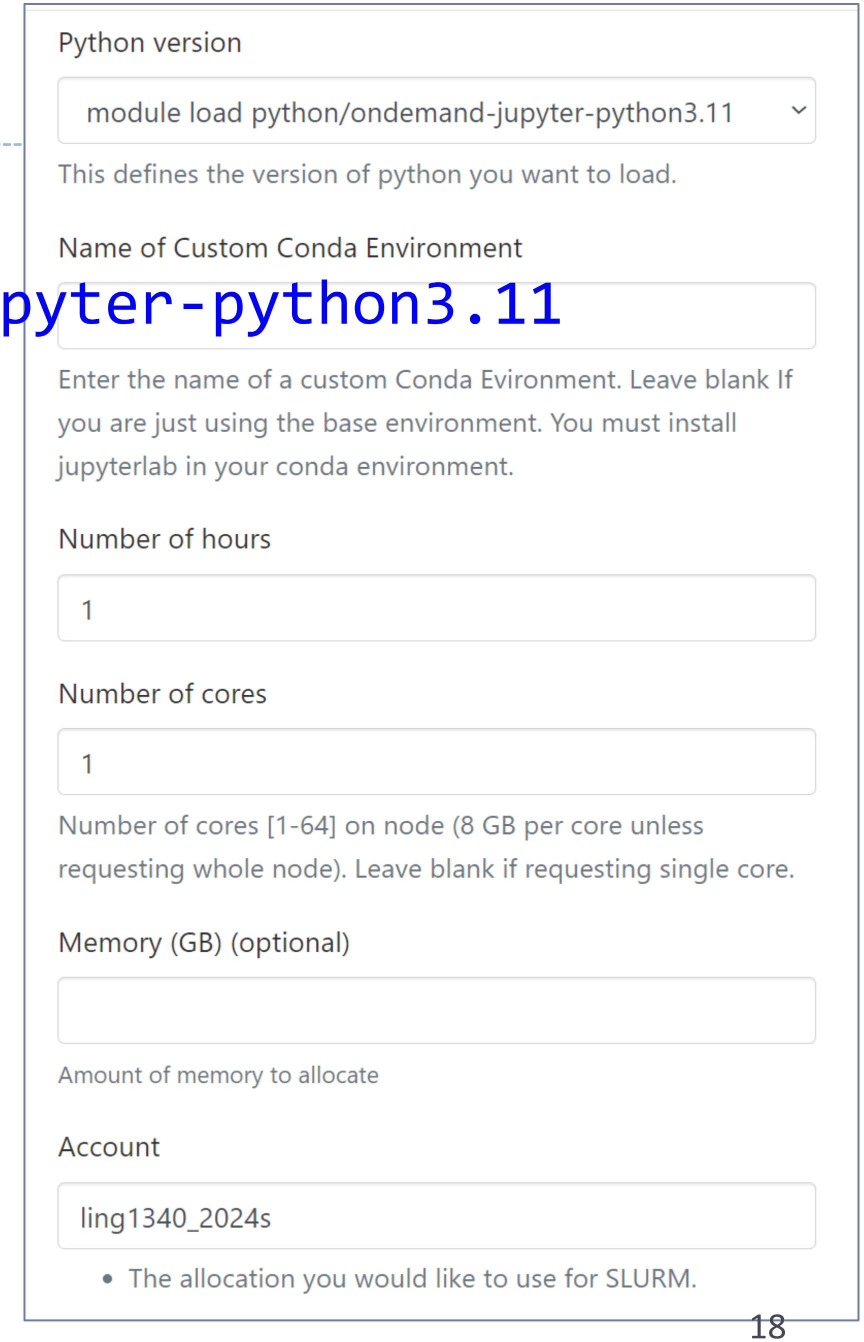


OnDemand provides an integrated, single access point for all of your HPC resources.

Pinned Apps A featured subset of all available apps

Launching a session

- Python version:
  `module load python/ondemand-jupyter-python3.11`
- Account: `ling1340_2024s`
- Memory (GB) (optional)
  - You may need to specify RAM amount
  - Default: 8GB per core.
    Your session will terminate if exceeded!



Python version

module load python/ondemand-jupyter-python3.11

This defines the version of python you want to load.

Name of Custom Conda Environment

Enter the name of a custom Conda Evironment. Leave blank If you are just using the base environment. You must install jupyterlab in your conda environment.

Number of hours

1

Number of cores

1

Number of cores [1-64] on node (8 GB per core unless requesting whole node). Leave blank if requesting single core.

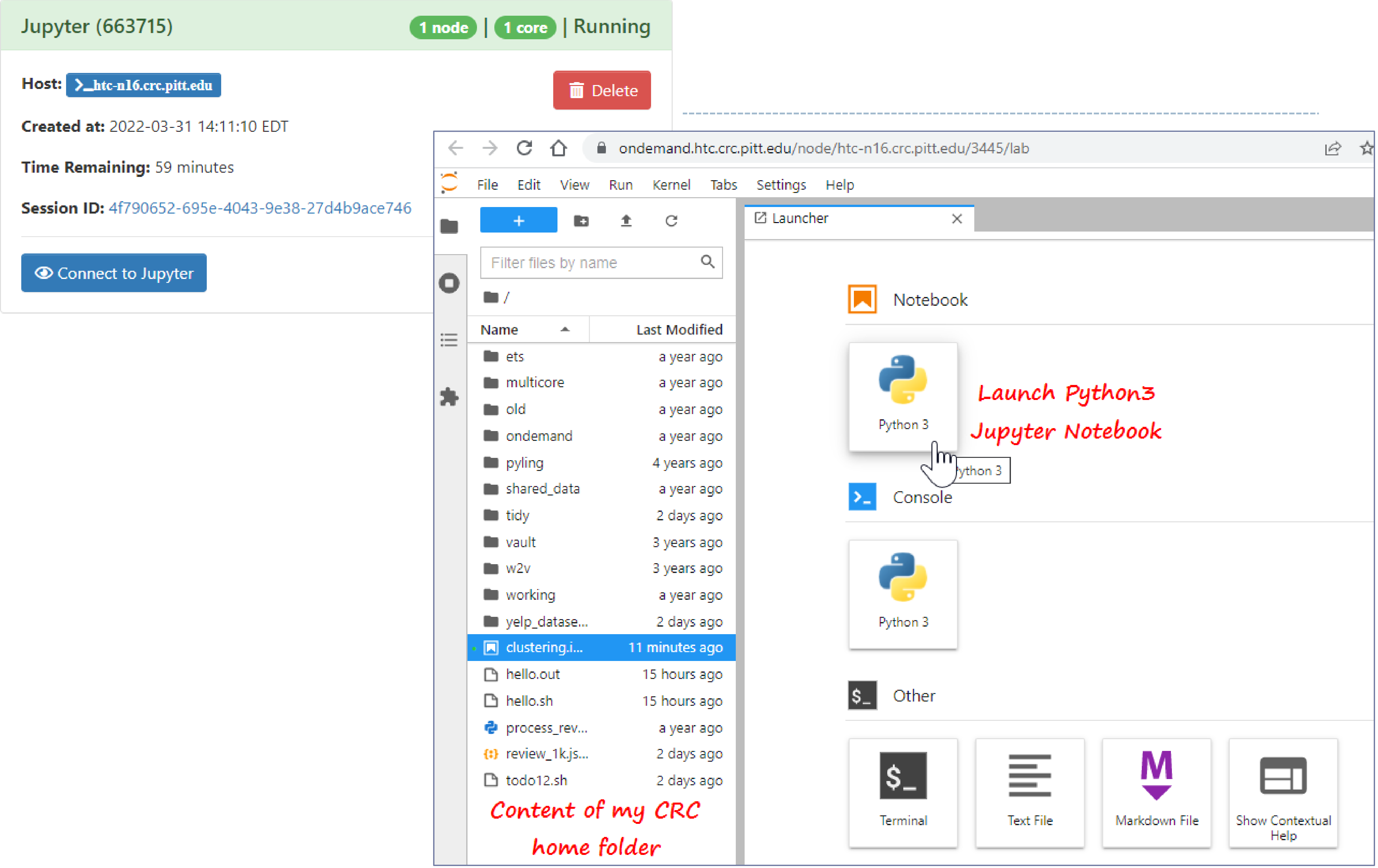
Memory (GB) (optional)

Amount of memory to allocate

Account

ling1340_2024s

- The allocation you would like to use for SLURM.



Jupyter (663715) 1 node | 1 core | Running

Host: >_htc-n16.crc.pitt.edu

Created at: 2022-03-31 14:11:10 EDT

Time Remaining: 59 minutes

Session ID: 4f790652-695e-4043-9e38-27d4b9ace746

ondemand.htc.crc.pitt.edu/node/htc-n16.crc.pitt.edu/3445/lab

Name	Last Modified
ets	a year ago
multicore	a year ago
old	a year ago
ondemand	a year ago
pyling	4 years ago
shared_data	a year ago
tidy	2 days ago
vault	3 years ago
w2v	3 years ago
working	a year ago
yelp_datase...	2 days ago
clustering.i...	11 minutes ago
hello.out	15 hours ago
hello.sh	15 hours ago
process_rev...	a year ago
review_1k.js...	2 days ago
todo12.sh	2 days ago

Launch Python3 Jupyter Notebook

Content of my CRC home folder

Wrap up

- Homework #4 out – don’t be too ambitious!
- Progress report #3, presentation up coming!