Northeastern University, Khoury College of Computer Science

CS 6220 Data Mining — Assignment 1

Due: January 18, 2023(100 points)

Haoran Zhang
https://github.com/hrcheung
zhang.haoran1@northeastern.edu

1 Getting Started - 10 points

1.1 Using Docker

Different companies use different tools for development and different work environments. For future assignments, we won't be prescriptive, but in this homework, we're going to familiarize ourselves with some of the most useful and common delivery and development environment tools in industry today.
Docker http://www.Docker.com is a useful mechanism for delivering software or scaling it up. For example, say we want to run a multi-computer job, passing *Docker containers* to each of the nodes in the cluster is one way to have repetitive and predictable behavior when doing large scale compute.
There are two essential Docker units: a **container** and a **container image**.

1. A **container** is a sandboxed process on your machine that is isolated from all other processes on the host machine. That isolation leverages kernel namespaces and cgroups, features that have been in Linux for a long time. Docker has worked to make these capabilities approachable and easy to use. To summarize, a container:
   a) is a runnable instance of an image. You can create, start, stop, move, or delete a container using the DockerAPI or CLI.
   b) can be run on local machines, virtual machines or deployed to the cloud.
   c) is portable (can be run on any OS).
   d) is isolated from other containers and runs its own software, binaries, and configurations.
2. When running a container, it uses an isolated filesystem. This custom filesystem is provided by a **container image**. Since the image contains the container's filesystem, it

must contain everything needed to run an application - all dependencies, configurations, scripts, binaries, etc. The image also contains other configuration for the container, such as environment variables, a default command to run, and other metadata.

Go ahead and download and install Docker. The getting started guide on Docker has detailed instructions for setting up Docker on

- Mac https://docs.docker.com/desktop/install/mac-install/,
- Linux https://docs.docker.com/install/linux/docker-ce/ubuntu
- Windows https://docs.docker.com/docker-for-windows/install.

1.2 Executing Your "Hello World"

For this assignment, we'll start with creating a Dockerfile in your submission folder. Specify the operating system and version of Python in the Dockerfile. You will subsequently need to install Python and libraries that you anticipate importing. Do not add the data into the image; you will need to pass that into the container with the `-v` Docker option.
For example, here's the most basic `Dockerfile`:

```
FROM ubuntu:20.04

RUN apt update && apt install -y sbcl
WORKDIR /usr/src
```

For this assignment, you'll set up your Docker environment and the appropriate versions of Python. Specifically,

1. Download and install Docker
2. Create your Dockerfile
3. Compile your Docker image
4. Screenshot a list of the Docker images available
5. Screenshot a list of the running Docker containers that include one with the image you created
6. Include both screenshots and the command you used in your write up

The Dockerfile is created as follows:

```
# syntax=docker/dockerfile:1
FROM python:3.8-slim-buster
WORKDIR /app
COPY requirements.txt requirements.txt
RUN pip3 install -r requirements.txt
COPY . .
CMD [ "python3", "-m" , "flask", "run", "--host=0.0.0.0"]
```

It specifies the official base image I use (python 3.8) and extra packages I need to install. Then based on this Dockerfile I build image using

```
docker build --tag python-docker .
```

and I can check all running images with Docker Desktop.

```
(.venv) (base) haoranzhang@haorandembp python-docker % docker images
REPOSITORY          TAG       IMAGE ID       CREATED              SIZE
python-docker       latest    a288427b4220   51 seconds ago       146MB
getting-started     latest    8f913da094d1   33 minutes ago       157MB
<none>              <none>    061ca78fb84d   About an hour ago    261MB
<none>              <none>    bca64ce4770a   About an hour ago    261MB
docker101tutorial   latest    1890c195bbc3   3 hours ago          47MB
alpine/git          latest    22d84a66cda4   8 weeks ago          43.6MB
```

Figure 1.1: docker images

To run image as a container, I run

```
docker run -e FLASK_APP=assignment1.py --publish 8000:5000 python-docker
```

in which -e sets the ENV and 8000:5000 allows us visiting 5000 using 8000 port. and it shows as:

```
(.venv) (base) haoranzhang@haorandembp python-docker % docker run --publish 8000:5000 python-docker
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WS
GI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://172.17.0.2:5000
Press CTRL+C to quit
172.17.0.1 - - [17/Jan/2023 03:24:44] "GET / HTTP/1.1" 200 -
```

Figure 1.2: run image as a container

The running containers can be displayed by

```
docker ps
```

or using Docker desktop as shown in **Figure 1.3**.
Visiting http://127.0.0.1:8000/ we will see

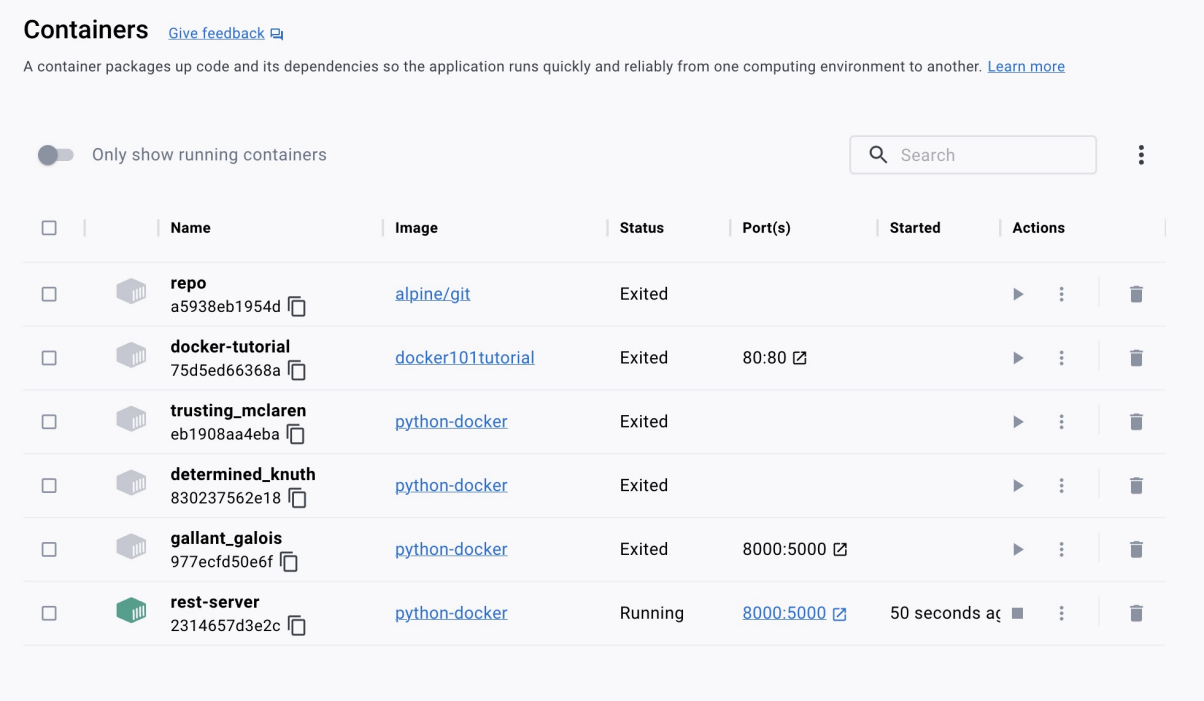

Figure 1.3: running containers

1.3 Github

Software version control at companies is essential for every software company in the industry. There are several types, including *Subversion/SVN* (which Google uses its in-house version branched from SVN). The most popular tool of choice is Github, which Microsoft recently bought.
At the end of this assignment, your submission will point to a repository, where the following files will be reviewed and subsequently graded:

- `Dockerfile` specifying what packages that you've used
- `assignment1.tex` file with your homework writeup
- `assignment1.pdf` file of the compiled version of your *.tex file
- `assignment1.py` file of your working code

None of the other files in that repository will be reviewed. We've provided a LaTeXtemplate that you can use for submission, provided here:

- https://github.com/kni-neu/homework-1/blob/main/assignment1-questions.tex

Do *NOT* include data into your Git repository. If you need help with LaTeX, the program that creates a PDF file from a coded text file (with extension `*.tex`), you may wish to use the online site `overleaf.com`. There is a helpful guide at this url:
https://www.overleaf.com/learn

Please see my github repo here: https://github.com/hrcheung/python-docker

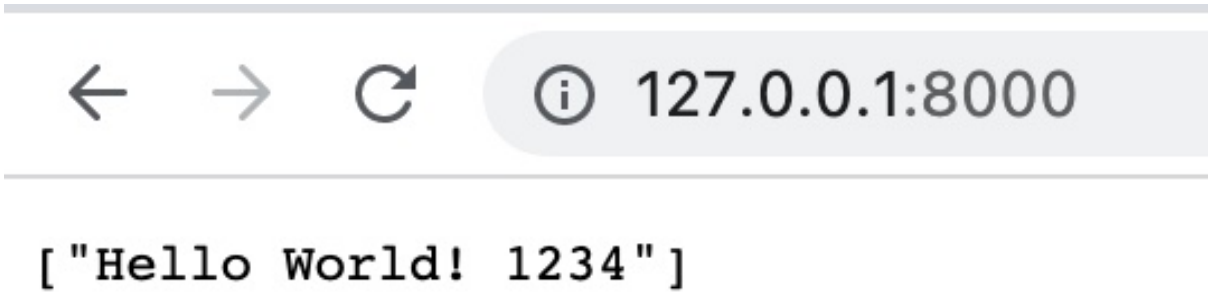

Figure 1.4: hello world

2 Identifying All Sets - 40 points

In subsequent lectures, you'll learn about frequent item sets, where relationships between items are learned by observing how often they co-occur in a set of data. This information is useful for making recommendations in a rule based manner. Before looking at frequent item sets, it is worth understanding the space of all possible sets and get a sense for how quickly the number of sets with unique items grows.
Suppose that we've received only a hundred records of items bought by customers at a market. Each line in the file represents the items an individual customer bought, i.e. their basket. For example, consider the following rows.

```
ham, cheese, bread
dates, bananas
celery, chocolate bars
```

Customer 1 has a basket of ham, cheese, and bread. Customer 2 has a basket of dates and bananas. Customer 3 has a basket of celery and chocolate bars. Each of these records is the receipt of a given customer, identifying what they bought.

1. What is the cardinality of the full set of unique items? Write a function called `cardinality_items` that takes a `.csv` text string file as input, where the format is as the above, and calculates the cardinality of the dataset.

   Answer: The cardinality is 21. I used a set() to traverse all items in the csv file and only keep those unique. Finally return the length of the set.

2. Taking any `.csv` file as a sample of a larger dataset, we'd occasionally like to understand the space of all possible subsets comprised of unique items. If there are N unique items (i.e., the cardinality of the entire dataset is N), how many sets with unique items can there possibly be? (Ignore the null set.) NOTE: I only expect the formula, and there is no code associated with this question.

   Answer: I will use combination to denote. k means the number of items in a set

$$^{N}C_{k} = \frac{N!}{k!(n-k)!}$$

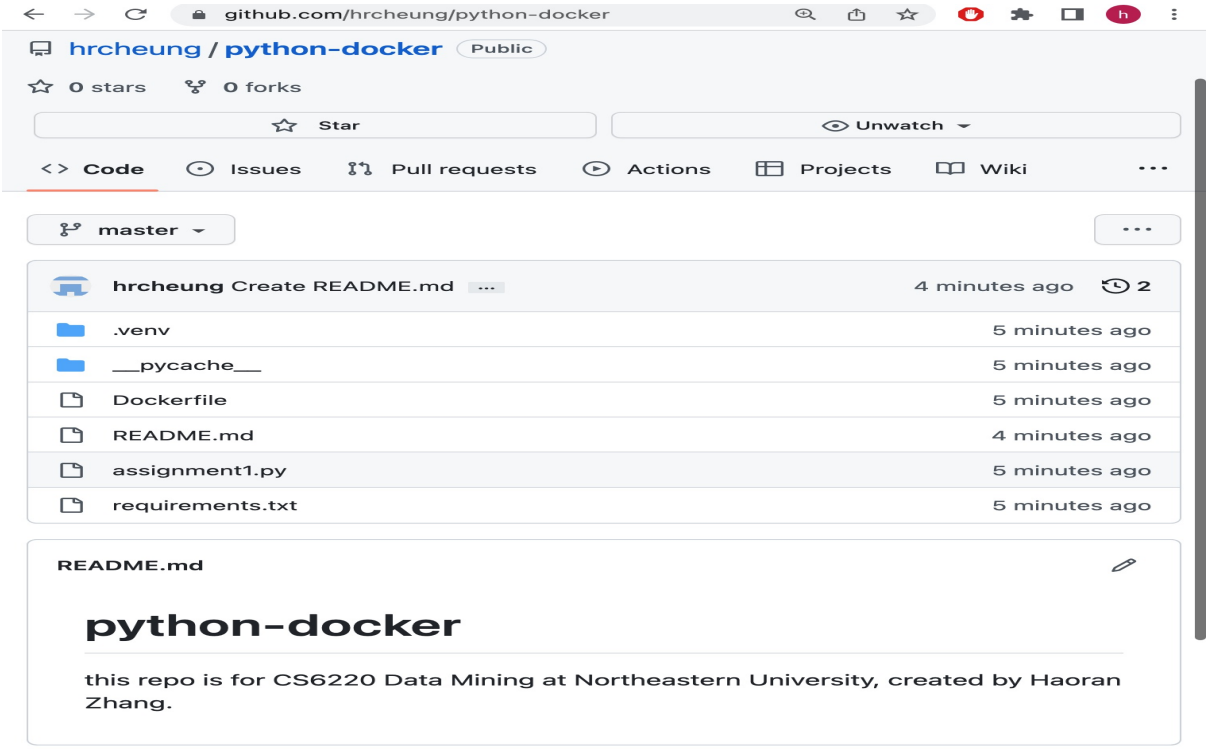

Figure 1.5: Github Repo

so we have

$$^{N}C_1 + {}^{N}C_2 + ... + {}^{N}C_N$$

3. Write a module called `all_itemsets()` with the following input/output:
   a) Input: $filename$ = the `.csv` text string file, where the format is as the above.
   b) Output: $L = [S_1, S_2, \cdots S_M]$, which is a list of all possible sets of with unique items N

```
all_itemsets.py:
  def all_itemsets(filename):
      import pandas as pd
      df=pd.read_csv(filename,sep='delimiter', \
                     header=None,engine='python')
      df=df[0].str.split(',',expand=True)
      rows=df.shape[0]
      cols=df.shape[1]
      car_set=set()
      for i in range(100):
          for j in range(4):
              if df[j][i]:
                  car_set.add(df[j][i].strip())
  #     print(car_set)
      res=[set()]
      for item in car_set:
```

Q2-1

```
In [319]: def cardinality_items():
              import pandas as pd
              #df=pd.read_csv("/app/basket_data.csv")
              df=pd.read_csv("/Users/haoranzhang/CS6220/homew
              df=df[0].str.split(',',expand=True)
              rows=df.shape[0]
              cols=df.shape[1]
              car_set=set()
              for i in range(100):
                  for j in range(4):
                      if df[j][i]:
                          car_set.add(df[j][i].strip())
              return(len(car_set))

In [320]: cardinality_items()

Out[320]: 21
```

Figure 2.1: q2-1

```
            for sub_s in res[:]:
                tmp=sub_s
                tmp.add(item)
                res.append(tmp)
        return res
```

This module has a function `all_itemsets`(filename) in it. It parsed csv file and traverse it to put all items into set to maintain items' uniqueness.

4. Let's take the small sample `.csv` provided as reflective of the distribution of the receipts writ large. So, for example, if the set $S = \{\text{bread}, \text{oatmeal}\}$ occurs twice in a dataset with 100 records, then the probability of item set $\{\text{bread}, \text{oatmeal}\}$ occurring is 0.02. Write a module called `prob_S` with the following input/output:

   a) Input:
   S = the set in question
   D = the entire Dataset (which if it's in memory, Python will pass by reference). In this case, D can be a list of lists or a list of sets:

   - `[ [A, B], [A, C], [C, D] , ... ]`
   - `[ {A, B}, {A, C}, {C, D} , ... ]`

   b) Output: $P(S)$ = the probability that S occurs

```
In [150]: import all_itemsets

In [151]: all_itemsets.all_itemsets("/Users/haoranzhang/CS622

Out[151]: [{'asparagus',
            'beans',
            'beer',
            'bread',
            'butter',
            'chips',
            'corn',
            'diapers',
            'ketchup',
            'leeks',
            'macaroni',
```

Figure 2.2: q2-3

```
prob_S.py:
    def calculateProb(S,D):
        cnt=0
        for i_set in D:
            if S==i_set:
                cnt+=1
        return cnt/len(D)
```

This module contains a function calculateProb(S,D). It traversed every item in D to see if it matches set S to count. Return the number of S in D and the total number of sets in D.

```
In [159]: import prob_S

In [161]: S= {"bread", "oatmeal"}
          D=[ {"bread", "oatmeal"}, {"bread", "oatmeal"}]
          for _ in range(98):
              D.append({"placeholder"})
          print("P(S)=",prob_S.calculateProb(S,D))

          P(S)= 0.02
```

Figure 2.3: q2-4

3 The Netflix Challenge - 50 points

One of the most famous challenges in data science and machine learning is Netflix's Grand Prize Challenge, where Netflix held an open competition for the best algorithm to predict user ratings for films. The grand prize was $1,000,000 and was won by BellKor's Pragmatic Chaos team. This is the dataset that was used in that competition.

- https://www.kaggle.com/datasets/netflix-inc/netflix-prize-data

In this exercise, we're going to do a bit of exploring in the Netflix Data. Start by downloading the data. If all worked out well, you should have the files in Fig. 3.7. The Kaggle dataset is close to 700MB large, and may take a long time to download. Do *not* include this data in your Docker container, but rather, mount the folder with the data.

3.1 Data Verification

Data integrity tends to be a problem in large scale processing, especially if there is little to no support. Therefore, it's important to verify the quality of the file download.

1. A large part of machine learning and data science is about getting data in the right format. Verify that the schema is the same as the Kaggle Dataset's description. Add screenshots to your assignment.

    - **ANSWER:** The first thing I want to verify is to verify if we have 17770 movies. I use "Contains" to check the number of movieId in files. I checked every `combined_data` file and sum up their movieId. It turns out we have only 17708 movies. Less than 17770. But this sample size is still good enough to do any predicition.

```
In [197]: combined_data_1.loc[combined_data_1[0].str.contains(":")]
```

Out[197]:

	0
0	1:
548	2:
694	3:
2707	4:
2850	5:
...	...
24046714	4495:
24047329	4496:
24056849	4497:
24057564	4498:
24057834	4499:

4499 rows × 1 columns

Figure 3.1: q3-1

- The second thing I do is to verify the form of datasets and whether they contains features they claimed by checking their head records. I verified that these datasets follow the form they claims in Kaggle.

3.2 Data Analysis

Let's answer the following questions in your writeup:

1. How many total records are there?
   - **Answer**: For `combined_data` files we have 104596333 records: number of records in `combined_data` - number of rows that are actually movieId.
   - There are 17708 movies in the `combined_data` files. In `movie_titles` we have 17770.
   - I skipped probe and qualifying as they are subsets of the above databsets.
2. Can you plot the distribution of star ratings over users and time? The granularity of the sliding window is at your discretion. Are there any trends?
   - **Answer**: Please note from this point I am using SAMPLING (n=100,000) to load in an acceptable timely manner.
   - We can plot the distribution over time since we do have timestamps. To plot, I use time as the x-axis and average(rating) as y-axis. For trend of ratings over time, we see that the ratings are closer in recent years. In early 2000, it ranged from 1 to 5 but recently it goes approximately 3.5-4.0.
   - We can plot the same distribution over CustomerID, BUT i don't think it makes any sense since id is arbitrarily set when user creating account. For trend of ratings over customers, as we expect, it means nothing since customer id is automatically generated so no meaning to check its trend.
3. What percentage of the films have gotten *more* popular over time?
   - **Answer:**
   - Since I sampled the dataset, the percentage might be inaccurate. But the idea is to parse the `combined_data` datasets, so every movieId comes with many ratings and timestamp the rating was given.
   - I will set a sliding window of 1 year, and aggregate all ratings of a movie in that year to an average number.
   - Then we traverse the dataset to check each movie if the rating is increasing, cnt++ if it is.
   - return cnt/the number of movies we have.
4. How many films have been re-released? How do you know?
   - **Answer:** If the re-released means Theater release, then we don't have enough information to answer this. Time in these records are not necessarily their release date. For example the time in `combined_data` files are the date the rating was given, it is not the date it released. Also, the Year of Release in `movie_titles`.txt are the release date of DVD, not theaterical.

5. What other information might we try to extract to better understand the data? For the questions that you may come up with (especially any time series data), make sure you back up your assertions with plots. Go ahead and play around with the data, and explore.
    - **Answer:** I am interested in knowing whether we have more DVD over the years. So on the x-axis we put time and on the y-axis we put the number of DVD released on that year. This helps us understand whether DVD market is shrinking.
    - According to the figure, clearly we can see that the Number of DVD released increased but dropped significantly these years. We can say the market of DVD is shrinking.
6. What are some interesting problems that we might solve? (No need to actually solve them!)
    - **Answer:**
    - I noticed that the Netflix challenge is to predict rating. However it seems that we don't have much information about our customer type, movie type, etc.
    - SO a good question to consider is to add features to movies and customers; for example, we can find Netflix's label on movies. By doing so allows us to see customers' preference towards different types of movies.

4 Grading Criterion

A significant portion of the grading rubric is the presentation of your report. We'll review:

1. the answers to questions.
2. your code and its legibility
3. the clarity of your write-up, including
    a) pipeline and code decisions,
    b) perspectives on the solution,
    c) and algorithmic rationale.

In [204]:
```
combined_data_1.head()
```

Out[204]:

	0
0	1:
1	1488844,3,2005-09-06
2	822109,5,2005-05-13
3	885013,4,2005-10-19
4	30878,4,2005-12-26

In [210]:
```
movie_titles=pd.read_csv("netflix-data/movie_titles.csv",sep='delimiter', h
```

In [211]:
```
movie_titles.head()
```

Out[211]:

	0
0	1,2003,Dinosaur Planet
1	2,2004,Isle of Man TT 2004 Review
2	3,1997,Character
3	4,1994,Paula Abdul's Get Up & Dance
4	5,2004,The Rise and Fall of ECW

In [213]:
```
probe=pd.read_table("netflix-data/probe.txt",header=None)
probe.head()
```

Out[213]:

	0
0	1:
1	30878
2	2647871
3	1283744
4	2488120

In [214]:
```
qualifying=pd.read_table("netflix-data/qualifying.txt",header=None)
qualifying.head()
```

Out[214]:

	0
0	1:
1	1046323,2005-12-19
2	1080030,2005-12-23
3	1830096,2005-03-14
4	368059,2005-05-26

Figure 3.2: q3-1-2

```
In [318]: len(combined_data_1)+len(combined_data_2)+len(combined_data_3)+len(combinec

          records in combined_data files =  104596333
```

Figure 3.3: q3-2-1

```
In [257]: sns.lineplot(x='time',y='ratings',data=movie_ratings_sample)

Out[257]: <matplotlib.axes._subplots.AxesSubplot at 0x7f81d8cd9fd0>
```

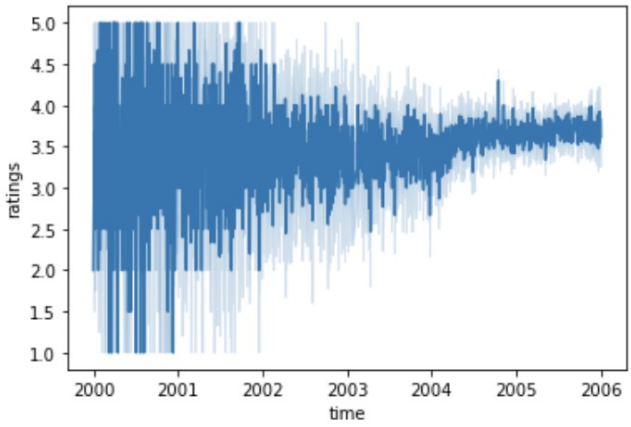

Figure 3.4: q3-2-1

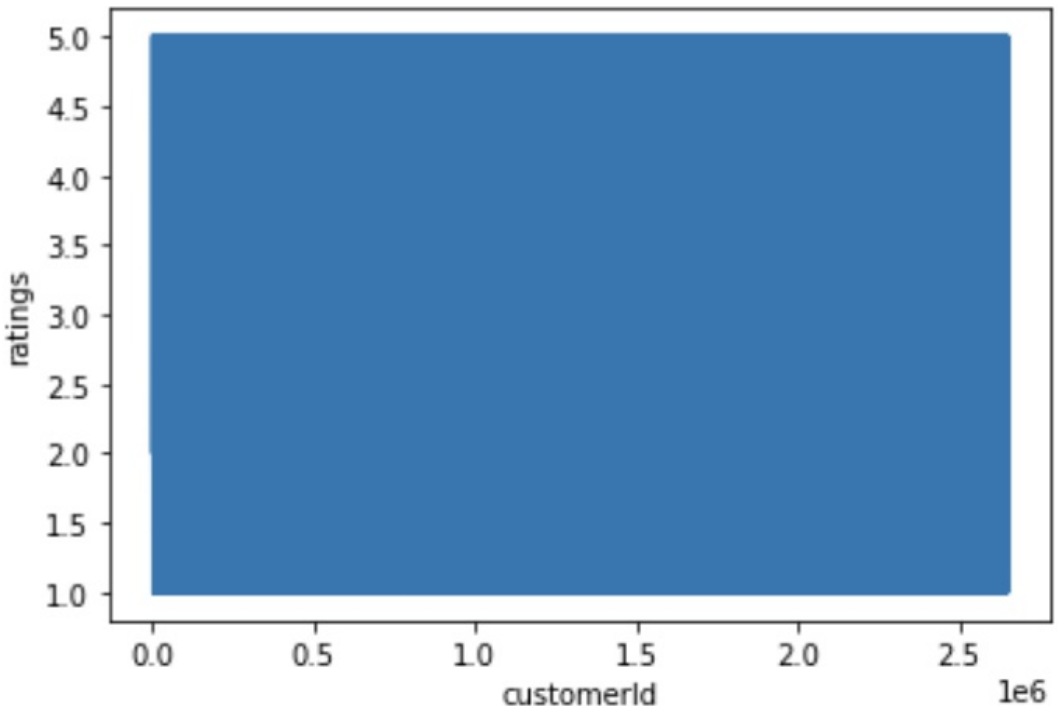

Figure 3.5: q3-2-2

```
In [314]: sns.lineplot(x='YearOfRelease',y='CntDVD',data=cnt_movies)
Out[314]: <matplotlib.axes._subplots.AxesSubplot at 0x7f8208a3a370>
```

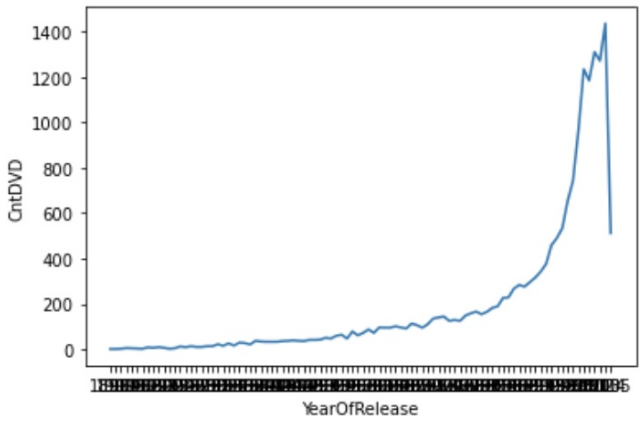

Figure 3.6: q3-2-5

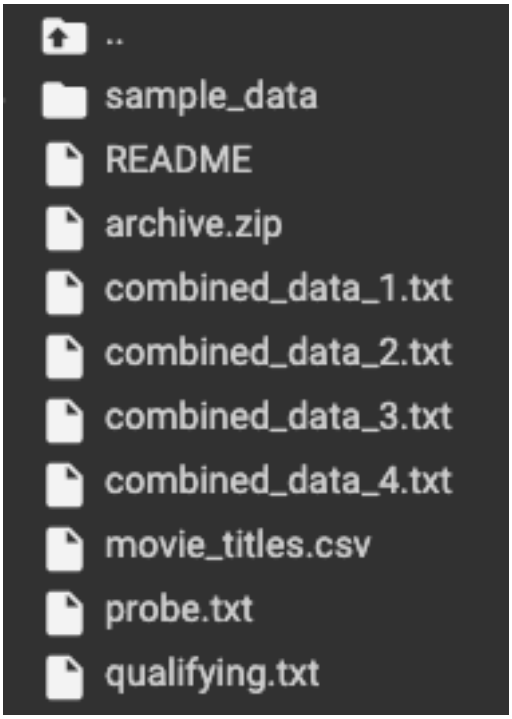

Figure 3.7: If everything worked out well, you should have the above files available to browse and process.