CSCI 5015 Assignment 5
Due March 23, 11:59 PM.
50 points

Objective

Create graphs with matplotlib and seaborn.

Data set

You've been given a ZIP file named **ml-20m.zip**. This file contains the MovieLens 20M Dataset, which comes from https://grouplens.org/datasets/movielens/. It contains "20 million ratings and 465,000 tag applications applied to 27,000 movies by 138,000 users." Although there are a lot of records, the file formats are simple and easily understood.

The ZIP contains a **README.txt** and several CSV files

- genome-scores.csv
- genome-tags.csv
- links.csv
- movies.csv
- ratings.csv
- tags.csv

We will primarily be using the movies.csv, ratings.csv, and tags.csv files.

Coding instructions

Create a Jupyter notebook file named **assignment5.ipynb**. Unzip the file, **ml-20m.zip** and place the three files **movies.csv**, **ratings.csv**, and **tags.csv** in the same folder as your notebook.

Make the first cell in your notebook a Markdown cell and create a header that says "Movie Analysis" and then on the next line, have a plain paragraph that says "Written by John Nicholson" (hopefully, it is obvious that you should replace my name with your name). It should look roughly like this after you render the Markdown:

> # Movie Analysis
>
> Written by John Nicholson

Use your notebook to answer the questions below. Your notebook should have both code cells and Markdown cells. The code cells should read and process the data, and the Markdown cell should explain what the associated code cell does and why you wrote the code the way you did. If you wish, add extra cells to make your main code easier to read and use.

Requirements

- No single code cell should produce all the output for more than one question. At a minimum, you will need 9 code cells, one for each question.
- If you make additional code cells, they do not have to display data. For example, you may want to write a helper function in a code cell. You still need a Markdown cell describing the function and why you wrote the function in the first place.
- All code cells need a Markdown cell explaining what the code does and why you wrote it the way you did. Explanations should be complete sentences and paragraphs, use good grammar, and be spell-checked.
- Before turning in your notebook, ensure that all code cells that produce output have their output displayed. I should not have to run your notebook in order to see the output. You should all render all Markdown cells. That is, I should not see the special Markdown characters like the `#`. I should see nicely rendered text.

Questions

For all questions, you must

- plot a graph in addition to the required Markdown cell that explains why you wrote the code the way you did
- Use a different set of colors for each graph
- Give each graph an appropriate title. For seaborn assign the plot to a variable and then use that variable to call `set_title`

```
ax = sns.countplot(...
ax.set_title('My awesome title')
```

For matplotlib, give the plot function a title argument, e.g. `title='My awesome title'`

1. How often are movies classified by more than one genre?

   In **movies.csv**, the third column contains the genres of each movie. Movies may have more than one genre. In that case each genre is separated by a pipe symbol (|). Create a countplot that shows how many movies have 0 genres in the data, 1 genre, 2 genres, 3 genres, up to the highest number of genres (whatever that is). Movies with no genres can be identified with **(no genres listed)**.

2. Which genres are the most popular?

   Graph the count of each individual genre with a countplot. Do not graph the count for movies with no genre, which can be identified with **(no genres listed)**. Sort the counts from the genre with the lowest count to the highest count.

3. Which tags are the most popular?

   In **tags.csv**, the third column contains tags for movies. A tag may consist of more than one word. Each row represents a tag an individual gave to a movie. Create a graph that shows the count of the top 10 most used tags sorted from lowest count to highest count. Make the tags lowercase for constancy .

4. How many ratings where given each year?

   In **ratings.csv**, the last column contains timestamp. The timestamp is a Unix timestamp that represents seconds from the Unix epoch. To convert this to a datetime object, you use `datetime.fromtimestamp`. Note that it needs an int, not a string. Then to get the year, you call the datetime's `strftime()`

   ```
   from datetime import datetime

   timestamp = '1112486027'
   dt = datetime.fromtimestamp(int(timestamp))
   year = dt.strftime('%Y')
   ```

   Using matplotlib, plot the number of ratings made each year. Rather than a countplot, use matplotlib directly. Plot each count for a year as a dot and connect all the dots as a line. Basically, combine the techniques in https://matplotlib.org/gallery/lines_bars_and_markers/simple_plot.html and https://matplotlib.org/gallery/subplots_axes_and_figures/subplot.html

5. How are ratings distributed?

   In **ratings.csv**, the third column contains ratings for movies. Values range from 0 - 5. Each row represents a rating an individual gave to a movie. Create a countplot that shows the count of how often a rating is used. Compute the average of all ratings, and use that to color the bars so that the bars for all ratings below the average are colored **red** and the bars for all ratings above the average are colored **blue**.

6. Compare ratings for four different movies.

   To answer this you will need to use two files, **ratings.csv** and **movies.csv**. In the **movies.csv**, the first column is the movie id. In the **ratings.csv**, the second column is the movie id. A single movie in **movies.csv** can have 0 or more matching rows in **ratings.csv**. If you are familiar with databases, this is similar to joining two tables.

   Pick four movies in **movies.csv** and retrieve their movidId. Create countplots for each movie's ratings. Graph it as one plot with 4 subplots, similar to https://seaborn.pydata.org/examples/anscombes_quartet.html. Be aware that example uses a scatter plot, you are doing a count plot. Make sure each subplot identifies which movie it represents.

Tips and help

Remember, we talked about files and listing directories in a previous lecture.

When you are running code, you may not see results right away. You should look to the left of the code cell for the `In [ ]:` marker. If you see a number, for example, `In [8]:`, the code in the cell is not currently running. But if you see an asterisk, for example, `In [*]:`, the code in the cell is running on the kernel, and results may not be available yet. The asterisk should change to a number when the code is done.

Remember, you can stop a running cell by choosing the "stop" button in the menu. It is the one that looks like a black square.

You can always come see me in my office or send me email. If you send me email, you should send your code as an attachment. Don't copy/paste your code into the message because that will make it harder for me to debug your code. Send your file as a `.ipynb` file

Submission

When your program is correct, log into D2L and locate the Dropbox for assignment 5. Upload the file

- assignment5.ipynb

into the D2L dropbox.

Contact me if you have any problems.