Coursework for ECS765P - Big Data Processing

Task 2: Analysis of Movie Ratings Data (25 points)

References:

1. Spark: The Definitive Guide: Big Data Processing Made Simple, 1st Edition by Bill Chambers and Matei Zaharia, published by O'Reilly Media, 2018. Available at `https://www.oreilly.com/library/view/spark-the-definitive/9781491912201/`.
2. Week 3, 4 and 5 Lectures, Labs and related resources.

Dataset Description

The dataset describes 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service. These data were created by users between January 09, 1995 and July 20, 2023. This dataset was generated on July 20, 2023. The dataset used in this coursework are movies.csv and ratings.csv in the //data-repository-bkt/ECS765/MovieLens/ directory.

Users were selected at random for inclusion. All selected users had rated at least 1 movie. No demographic information is included. Each user is represented by an id, and no other information is provided.

Purpose

The purpose of this task is to analyze movie ratings to gain insights into user preferences, rating patterns, and movie popularity.

Sample Records

Here are some sample records from the dataset to explain the fields:

Movies Dataset

movieId	title	genres
5173	Divorcing Jack (1998)	Comedy|Thriller
174959	50 Kilo Albaloo (2016)	Children|Comedy|Drama
146690	Enter Laughing (1967)	(no genres listed)
175439	The Abduction Club (2002)	Comedy
223075	The Dalton Gang (2020)	(no genres listed)
235383	Loved (1997)	Drama|Thriller
273131	Firestarter (2022)	Fantasy|Horror|Thriller
124408	Last of the Comanches (1953)	(no genres listed)
277021	Radical Evil (2014)	Documentary
123298	The Lost Continent (1968)	Adventure|Fantasy

Ratings Dataset

userId	movieId	rating	timestamp
324684	3898	1.5	1144760809
158793	922	4	1450700720
313695	2423	4	1133582037
59202	5218	3	1033884899
158269	1779	2	1660913035
264464	1957	4	953067620
12727	106782	3.5	1653959620
319929	9018	3.5	1469934763
289257	166643	3.5	1515856802
66339	2105	4	1553843526

Explanation of Fields

Movies Dataset

- **movieId**: The unique identifier for each movie.
- **title**: The title of the movie.
- **genres**: The genres associated with the movie, separated by a pipe (|) character.

Ratings Dataset

- **userId**: The unique identifier for each user.
- **movieId**: The unique identifier for each movie.
- **rating**: The rating given by the user to the movie, on a scale from 0.5 to 5.0.

- **timestamp**: The timestamp when the rating was given, in Unix epoch format.

Questions

1. **Load Data (2 points) File:** `//data-repository-bkt/ECS765/MovieLens/ratings.csv`
   - Load the Movie Ratings dataset into a dataframe.
   - Determine and print the number of unique users who have rated movies in the dataset.
   - Include a screenshot of your results in your report. For example:

```
2025-02-07 12:09:45,823 INFO scheduler.DAGScheduler: Job 2 is finished. Cancelling potential speculative or zombie tasks for this job
2025-02-07 12:09:45,823 INFO scheduler.TaskSchedulerImpl: Killing all running tasks in stage 4: Stage finished
2025-02-07 12:09:45,823 INFO scheduler.DAGScheduler: Job 2 finished: count at NativeMethodAccessorImpl.java:0, took 20.415551 s
Number of unique users: [redacted]
2025-02-07 12:09:45,879 INFO spark.SparkContext: Invoking stop() from shutdown hook
2025-02-07 12:09:45,895 INFO server.AbstractConnector: Stopped Spark@3c323e62{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
2025-02-07 12:09:45,897 INFO ui.SparkUI: Stopped Spark web UI at http://task2-1-42dd1094e04fc898-driver-svc.data-science-eex654.svc:4040
```

2. **Filter Data (3 points) File:** `//data-repository-bkt/ECS765/MovieLens/ratings.csv`
   - Convert the timestamp field to "YYYY-MM-DD" format.
   - Sort your results by date.
   - Include a screenshot of your results in your report. For example:

```
2025-02-07 12:23:25,977 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
2025-02-07 12:23:25,979 INFO scheduler.DAGScheduler: ResultStage 2 (showString at NativeMethodAccessorImpl.java:0) finished in 41.434 s
2025-02-07 12:23:25,980 INFO scheduler.DAGScheduler: Job 2 is finished. Cancelling potential speculative or zombie tasks for this job
2025-02-07 12:23:25,980 INFO scheduler.TaskSchedulerImpl: Killing all running tasks in stage 2: Stage finished
2025-02-07 12:23:25,981 INFO scheduler.DAGScheduler: Job 2 finished: showString at NativeMethodAccessorImpl.java:0, took 41.446136 s
2025-02-07 12:23:26,025 INFO codegen.CodeGenerator: Code generated in 19.741348 ms
2025-02-07 12:23:26,048 INFO codegen.CodeGenerator: Code generated in 14.468842 ms
+------+-------+------+---------+----------+
|userId|movieId|rating|timestamp|      date|
+------+-------+------+---------+----------+
|*****|  ****|  ***|*******|yyyy-mm-dd|
|*****|  ****|  ***|*******|yyyy-mm-dd|
|*****|  ****|  ***|*******|yyyy-mm-dd|
|*****|  ****|  ***|*******|yyyy-mm-dd|
|*****|  ****|  ***|*******|yyyy-mm-dd|
```

3. **Rating Distribution (4 points) File:** `//data-repository-bkt/ECS765/MovieLens/ratings.csv`
   - Create a new column that categorizes ratings into "Low" (1-2), "Medium" (3-4), and "High" (5).
   - Group the dataset by rating category and count the number of ratings per category.

- Visualize the rating distribution using a pie chart. For an example, see the pie chart Example at `https://matplotlib.org/stable/gallery/pie_and_polar_charts/pie_features.html`
- Include a screenshot of your visualization in your report.

4. **Add Columns (4 points) File:** `//data-repository-bkt/ECS765/MovieLens/ratings.csv`
   - Add new columns by extracting the year and month from the timestamp.
   - Create a new column that categorizes the time of rating into "Early Year" (January - June) and "Late Year" (July - December).
   - Show 10 rows of the results in your report.
   - Plot a bar chart for the number of ratings for each time of year (Early Year, Late Year). For an example, see Bar Chart Example at `https://matplotlib.org/stable/gallery/lines_bars_and_markers/barchart.html`.
5. **Genre Analysis (4 points) Files:** `//data-repository-bkt/ECS765/MovieLens/movies.csv` and `//data-repository-bkt/ECS765/MovieLens/ratings.csv`
   - Create a new column that lists the genres of each movie.
   - Filter out invalid genres from the dataset.
   - Group the dataset by genre and count the number of ratings per genre.
   - Visualize the genre distribution using a bar chart. For an example, see Bar Chart Example at `https://matplotlib.org/stable/gallery/lines_bars_and_markers/barchart.html`.
   - Include a screenshot of your visualization in your report.

   Valid genres are: Action, Adventure, Animation, Children, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, Western.
6. **Temporal Analysis (3 points) File:** `//data-repository-bkt/ECS765/MovieLens/ratings.csv`
   - Group the dataset by year and aggregate the total number of ratings per year.
   - Create new columns based on the aggregated metrics.
   - Show 10 samples of the results in your report.
   - Visualize the aggregated metrics using a line chart. For an example, see Bar Chart Example at `https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html`
   - Include a screenshot of your visualization in your report.

7. **Top Movies (2 points) Files:** `//data-repository-bkt/ECS765/MovieLens/movies.csv` and `//data-repository-bkt/ECS765/MovieLens/ratings.csv`

   - Find the top 10 movies with the highest average ratings.
   - Include a screenshot of your results. For example:

```
2025-02-07 18:14:42,588 INFO scheduler.DAGScheduler: Job 5 is finished. Cancelling potential speculative or zombie tasks for this job
2025-02-07 18:14:42,588 INFO scheduler.TaskSchedulerImpl: Killing all running tasks in stage 7: Stage finished
2025-02-07 18:14:42,589 INFO scheduler.DAGScheduler: Job 5 finished: showString at NativeMethodAccessorImpl.java:0, took 20.273910 s
2025-02-07 18:14:42,620 INFO codegen.CodeGenerator: Code generated in 15.083768 ms
+-------+---------+--------------------+--------------------+
|movieId|avg_rating|               title|              genres|
+-------+---------+--------------------+--------------------+

|******|   ***|***************|   **********|

|******|   ***|***************|   **********|

|******|   ***|***************|   **********|

|******|   ***|***************|   **********|

|******|   ***|***************|   **********|
```

8. **User Analysis (3 points) File:** `//data-repository-bkt/ECS765/MovieLens/ratings.csv`

   - Create a new column that categorizes users into "Frequent Raters" (more than 50 ratings) and "Infrequent Raters" (50 or fewer ratings).
   - Find the top 10 users with the highest number of ratings.
   - Visualize the distribution of frequent and infrequent raters using a bar chart. For an example, see Bar Chart Example at `https://matplotlib.org/stable/gallery/lines_bars_and_markers/barchart.html`.
   - Include a screenshot of your visualization in your report.