BUAN 5315 01 / 30 May, 2023 / Amrapali Samanta

Use Case Name

Ripe Pumpkins - a movie review-aggregation service

Use Case URL (web site URL, aka the source project)

https://www.codementor.io/@jadianes/building-a-recommender-with-apache-spark-python-example-app-part1-du1083qbw

Introduction - what is this use case

Our start-up business 'Ripe Pumpkins' is building a collaborative recommendation measurement system called 'Pumpkinmeter' for millions of fans. This use case focuses on building a movie recommendation system using collaborative filtering with Spark. The goal is to provide personalized movie recommendations to users based on their preferences and historical movie ratings. The system utilizes Spark's distributed computing capabilities to handle large-scale datasets and deliver real-time recommendations.

Dataset used – name, source address, summary, (list of all fields of dataset, as appendix)

Name: GroupLens MovieLens Dataset

Small: 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users. Last updated 9/2018.

Full: 27,000,000 ratings and 1,100,000 tag applications applied to 58,000 movies by 280,000 users. Includes tag genome data with 14 million relevance scores across 1,100 tags. Last updated 9/2018.

Source address:
Complete dataset URL
http://files.grouplens.org/datasets/movielens/ml-latest-small.zip

Small dataset URL
http://files.grouplens.org/datasets/movielens/ml-latest.zip

Summary: The GroupLens MovieLens dataset contains movie ratings and user reviews collected from the MovieLens website. It includes information such as user ratings, movie titles, genres, timestamps, and demographic data. This dataset serves as a valuable resource for training and evaluating recommendation systems.
[See Appendix]

Technical Details:

The system utilizes Spark's distributed computing capabilities to handle large-scale datasets and Collaborative Filtering to deliver real-time recommendations. Collaborative recommendation, also known as collaborative filtering, involves predicting user interests by gathering preferences or taste information from a group of users. The core idea is that if user A shares a similar opinion with user B on one topic, it is more probable that A will also have B's opinion on a different topic, rather than randomly selecting another user's opinion on that topic.

A new movie recommender is being built using collaborative filtering with Spark's Alternating Least Squares implementation. The initial step focuses on acquiring and parsing movie and ratings data into Spark RDDs. The objective is to obtain the necessary data and format it appropriately for use in the subsequent stages. The following step involves constructing the recommender system and utilizing it to generate recommendations. Additionally, the recommender is stored for future usage in our online recommendation system.

Debugging Details: Any Challenges That Had To Be Overcome? What did you update/modify to run the project? Any achievements you have made? What are key features of coding practices in this project?

Challenges: One challenge was in the initial days it was taking long time to process and train the model. Hence, handling the resources was needed for running the complete dataset which consists of huge volumes of data. I addressed this issue by changing the instance type to t2.large to speed up the processing time of the algorithm. Also, we were not satisfied with having recommendations with more than 25 reviews only, hence we took the decision to also get results for more than 1000 reviews.

Updates/Modifications: The project involved updating and modifying the data preprocessing steps to handle specific requirements of the MovieLens dataset. From starting the project with one user data , we have used two user data to get recommendations as it creates more diverse user input.

Each line in the ratings dataset (ratings.csv) is formatted as: userId, movieId, rating, timestamp. We drop the timestamp because we do not need it for this recommender. Each line in the movies (movies.csv) dataset is formatted as: movieId, title, genresFor each line in the movies dataset, we create a tuple of (MovieID, Title). We drop the genres because we do not use them for this recommender at this stage.

Achievements: Changing instance type to t2.large has significantly helped in achieving a very shorter time to get results of the model prediction even though working on such larger dataset. The implementation of the collaborative filtering recommendation system using

Spark achieved movie recommendations and improved user satisfaction. The results from having more than 100 reviews for both users gave better results when compared with 25 or more reviews.

Results: Discuss Spark output with above test sets and scenarios

After doing the collaborative filtering on a complete dataset consisting of 70% as testing data, we were able to get the Root Mean Squared Error (RMSE) of 0.8318265262101795 which indicates the average prediction error of a recommendation model. In the context of a movie recommendation system, this RMSE value suggests that the model's predictions have an average deviation of approximately 0.83 from the actual ratings in the testing data. We have also tested on a smaller dataset where the RMSE was 0.91. Hence, we confirm that in comparison to a smaller dataset, the complete dataset which is much larger and diverse testing dataset is giving better results, as a lower RMSE indicates better accuracy, meaning that the model's predictions are closer to the actual ratings.

Now coming to the results of recommendations from our Collaborative Filtering algorithm, we have done the testing of the model on two types of User data- myself and my friend so that there is no biasness while I am training the model. We have given our movie ratings to our liking. I like to watch movies high on crime, drama, thriller, adventure, action more than romance and drama, and my friend has listed movies which fall widely under romance, drama, adventure and few which are of crime, action and adventure. From the top recommendations results as shown below we observed that the recommendation engine suggests results tailored to mine or my friend’s interests, and also much better results for the ones with more than 100 reviews. Cross-Genre Recommendations were also observed which is essential to expand customer preferences. As we saw even though user rated a particular movie low, and listed only one or two movies of different genres amongst others which were similar, our recommendation engine suggested cross-genre recommendations and we believe this can lead to wider range of content suggestions.

User 1

Listed 10 movies I have watched in the past and provided my own ratings for them.
I have listed more movies adhering to the genres like crime, drama, mystery, scifi, thriller and one for anime, adventure, romance. The dataset movies file is checked for ID to Tittle assignment. After training, the results were pretty good. To name a few as per my user ratings given, I got suggestions for movies like 'Olive Kitteridge (2014)', 'Lives of Others', ‘Schindler's List (1993) ‘, 'Over the Garden Wall (2013)’ which are related to crime, drama, thriller, mystery or serious documentaries which I would definitely watch. As I saw Godfather, it also suggested 'The Godfather Trilogy’ to see which is correct.

Adding new user ratings

```
new_user1_ratings = [
    (0,260,4), # Star Wars: Episode IV - A New Hope (1977)
    (0,58559,4), # Dark Knight, The (2008)
    (0,48516,3), # Departed, The (2006)
    (0,593,4), # Silence of the Lambs, The (1991)
    (0,32,3), # Twelve Monkeys (a.k.a. 12 Monkeys) (1995)
    (0,79132,4), # Inception (2010)
    (0,318,5), # Shawshank Redemption, The (1994)
    (0,364,2), # Lion King, The (1994)
    (0,858,5), # Godfather, The (1972)
    (0,356,5) # Forrest Gump (1994)
   ]
```

Getting top recommendations for User 1
User 1 : Scenario 1

TOP recommended movies for User 1 (with more than 25 reviews):
('Planet Earth II (2016)', 4.3361926630893315, 853)
('Planet Earth (2006)', 4.312682184100417, 1384)
('Band of Brothers (2001)', 4.253593879561944, 984)
('Blue Planet II (2017)', 4.250443625017688, 349)
('Life (2009)', 4.2419448598883704, 166)
('The Blue Planet (2001)', 4.217744131351459, 421)
('Frozen Planet (2011)', 4.203898463119426, 402)
('Cosmos', 4.193298933313592, 157)
('"Things I Like', 4.188455804323436, 30)
('The Reichenbach Fall (2012)', 4.167207554271638, 48)
("Won't You Be My Neighbor? (2018)", 4.159217682024021, 83)
('The Godfather Trilogy: 1972-1990 (1992)', 4.153614009560567, 421)
('Music for One Apartment and Six Drummers (2001)', 4.145544574312158, 31)
('Over the Garden Wall (2013)', 4.1433365188486135, 377)
('The Farthest (2017)', 4.1273209637729416, 28)

User 1 : Scenario 2

TOP recommended movies for User 1 (with more than 100 reviews):
('Planet Earth II (2016)', 4.3361926630893315, 853)
('Planet Earth (2006)', 4.312682184100417, 1384)
('Band of Brothers (2001)', 4.253593879561944, 984)
('Blue Planet II (2017)', 4.250443625017688, 349)
('Life (2009)', 4.2419448598883704, 166)
('The Blue Planet (2001)', 4.217744131351459, 421)

('Frozen Planet (2011)', 4.203898463119426, 402)
('Cosmos', 4.193298933313592, 157)
('The Godfather Trilogy: 1972-1990 (1992)', 4.153614009560567, 421)
('Over the Garden Wall (2013)', 4.1433365188486135, 377)
('Alone in the Wilderness (2004)', 4.125838776352651, 343)
('"Civil War', 4.117653583820509, 431)
('Olive Kitteridge (2014)', 4.1051077804416956, 211)
('"Lives of Others', 4.097398034313056, 9670)
("Schindler's List (1993)", 4.093065821867768, 71516)

User 2

Adding new user ratings

User 2: Listed 10 movies my friend has watched in the past and provided her own ratings for them. She has listed more movies adhering to the genres like romance, comedy, drama, and very few for action, crime, adventure, war. The dataset movies file is checked for ID to Tittle assignment. After training, the results were pretty good. To name a few as per my user ratings given, I got suggestions for movies like 'Olive Kitteridge (2014)', 'Lives of Others', 'Schindler's List (1993) ', 'Over the Garden Wall (2013)' which are related to crime, drama, thriller, mystery or serious documentaries which I would definitely watch.

```
new_user2_ratings = [
    (0,193886,2), # Leal (2018)
    (0,193868,4), # Dos tipos de cuidado (1953)
    (0,15,2), # Cutthroat Island (1995)
    (0,25,3), # Leaving Las Vegas (1995)
    (0,17,4), # Sense and Sensibility (1995)
    (0,52,4), # Mighty Aphrodite (1995)
    (0,94,3.5), # Beautiful Girls (1996)
    (0,101,4), # Bottle Rocket (1996)
    (0,118,4), # If Lucy Fell (1996)
    (0,920,5) # Gone with the Wind (1939)
   ]
```

Getting top recommendations for User 2

User 2 : Scenario 1

Get the highest rated recommendations for the new user 1, filtering out movies with less than 25 ratings.

TOP recommended movies (with more than 25 reviews):
('Connections (1978)', 4.972784990970419, 49)
("'I', 4.698024321335621, 85)
("'Lonely Wife', 4.689532334301891, 43)
('Он вам не Димон (2017)', 4.663090597476431, 26)
("'Hollow Crown', 4.650107488176603, 36)
('Hamlet (Gamlet) (1964)', 4.601264259880479, 37)
('Winter in Prostokvashino (1984)', 4.589377701769553, 67)
('Between the Folds (2008)', 4.584828588461592, 61)
("Mickey's Trailer (1938)", 4.582243192054774, 27)
("Won't You Be My Neighbor? (2018)", 4.5760024611808445, 83)
('Baseball (1994)', 4.571428071073244, 42)
("'Civil War', 4.565837238856659, 431)
('My Love (2006)', 4.562778882188419, 32)
('Queen: Days of Our Lives (2011)', 4.549557066263571, 32)
('Small Potatoes - Who Killed the USFL? (2009)', 4.539391643687392, 26)

User 2 : Scenario 2
Get the highest rated recommendations for the new user 2, filtering out movies with less than 100 ratings.

TOP recommended movies (with more than 100 reviews):
("'Civil War', 4.565837238856659, 431)
('Casablanca (1942)', 4.495544998551394, 31095)
("Smiley's People (1982)", 4.479628675843423, 116)
('To Kill a Mockingbird (1962)', 4.454131258429257, 17988)
('Death on the Staircase (Soupçons) (2004)', 4.45267974733346, 130)
('All About Eve (1950)', 4.450888303246153, 5632)
("'Philadelphia Story', 4.449410062335382, 7828)
('Trouble in Paradise (1932)', 4.4471237590880115, 468)
('That Munchhausen (1979)', 4.437561953127243, 104)
('Planet Earth (2006)', 4.4363972587016995, 1384)
('The Adventures of Sherlock Holmes and Dr. Watson: Bloody Signature (1979)', 4.436343419948689, 141)
('Rear Window (1954)', 4.432485593585598, 22264)
('It Happened One Night (1934)', 4.431701435291398, 4750)
('Wild China (2008)', 4.42291534335733, 105)
('Witness for the Prosecution (1957)', 4.422858806261685, 2180)

These personalized recommendations demonstrate how the recommendation system takes into account the specific movie preferences and genres favoured by each user, resulting in tailored suggestions that align with their individual tastes.

Another valuable application is obtaining the predicted rating for a specific movie by a particular user. The approach is similar to retrieving top recommendations, but instead of utilizing predictAll for every unrated movie, we only provide the method with a single entry containing the movie for which we desire the rating prediction. Also, in order to save time when starting up the server, we were able to persist some of the RDDs we have generated, specially those that took longer to process.

Insight: What will be insights/actionable items you learned from the case – business implications/impacts?

The insights and actionable items derived from this use case have significant business implications as Ripe Pumpkins' new initiative, Pumpkinmeter score seems to have a lot of potential. Our results of recommendations based on two users, has suggested movies similar to their specific tastes. This insight emphasizes the need for businesses to invest in personalized recommendation algorithms to enhance user satisfaction and engagement. From the recent success rates of other recommendation model in streaming services we know how likely it is that users tend to gravitate towards specific genres when selecting movies. By implementing a movie recommendation system 'Pumpkinmeter' using Spark's collaborative filtering, Ripe Pumpkins can enhance user engagement, increase customer retention, and improve revenue generation. The personalized movie recommendations and also Cross-Genre Recommendations can lead to increased user satisfaction and provide valuable insights into user preferences, enabling targeted marketing strategies and content customization and increased Customer Satisfaction.

Some of the actionable items would be tune our filtering algorithm based on more variety of user data and observe the results. Also, using genre and timestamp as filtering recommendations is something on our upcoming agenda. We believe recommendations based on genre will be more accurate than what it is now. Encouraging users to provide ratings and feedback on recommended movies can further refine the recommendation system. By actively collecting and analysing user feedback, our busines can identify areas for improvement, refine the algorithms, and enhance the accuracy and relevance of future recommendations. We are also planning to modify and build a model which we can use in a web environment to be able to provide on-line movie recommendations.

References (at least 3 references including the main project source, lecture, presentation, etc.)

https://spark.apache.org/docs/latest/mllib-collaborative-filtering.html

https://www.codementor.io/@jadianes/building-a-recommender-with-apache-spark-python-example-app-part1-du1083qbw

Lee, J. (2023, June). *Lecture 8: Data Processing Patterns II*. [BUAN 5315 01 Lecture notes]. *Seattle University.* Retrieved from https://seattleu.instructure.com/courses/1608543/pages/lecture-8-data-processing-patterns-ii

Lee, J,(2023). [BUAN 5315 01 Data Translation Challenge]. *Seattle University.* Retrieved from https://seattleu.instructure.com/courses/1608543/assignments/7112409

Netflix research. Netflix Research. (n.d.). https://research.netflix.com/research-area/recommendations

Appendix – any other contents that can support your project

Columns of movies.csv dataset:
movieId
title
genre

Columns of ratings.csv dataset:
userId
movieId
rating
timestamp