COLLABORATIVE FILTERING

Introduction:

In this report, I delve into collaborative filtering, a cornerstone technique in recommendation systems. My focus is on the implementation of the Alternating Least Squares (ALS) algorithm using PySpark. My objective is to apply collaborative filtering to the MovieLens 25M dataset provided by GroupLens. This dataset comprises 25 million movie ratings and one million tag applications, contributed by 162,000 users for 62,000 movies. Additionally, it includes tag genome data with 15 million relevance scores across 1,129 tags. Released in December 2019, this dataset serves as a stable benchmark for recommendation system research.

Given the challenges of setting up PySpark locally, I opt to use Google Colab for coding. This decision ensures a hassle-free environment for experimentation and analysis without compromising our Python environment. Through this exploration, I aim to gain insights into collaborative filtering and its practical application in recommending movies based on user preferences.

Data Preprocessing:

I employed pandas to read the CSV and conducted an initial data exploration to identify any outliers or irrelevant data. Fortunately, the dataset was clean with no missing values or NAs. As all features were on the same scale, data normalization was unnecessary. I dropped the timestamp column as it was deemed irrelevant for our analysis. Before splitting the data into training and testing sets, I grouped it based on the user_ID. This grouping allowed for a more granular analysis of each user's interactions with items, facilitating the identification of patterns, preferences, and similarities among users.

Training:

To train the model, I opted for the PySpark.ML implementation of the Alternating Least Squares (ALS) algorithm. The choice of PySpark.ML over PySpark.MLLIB was motivated by its more recent implementation, DataFrame-centric approach, and user-friendly API. PySpark.ML offers a higher-level abstraction, streamlining the machine learning workflow and making it easier for data scientists and developers to work with. This aligns well with our goal of building an efficient and scalable recommendation system.

Parameters:

Rank	maxIter	regParam	userCol	movieId	ratingCol	coldStart Stratergy	NonNegative	Seed	RMSE
10	5	0.01	userId	movieId	rating	drop	Fale	16	0.810
20	10	0.1	userId	movieId	rating	drop	True	16	0.812
25	10	0.01	userId	movieId	rating	drop	True	16	0.802
25	15	0.01	userId	movieId	rating	nan	True	16	nan
25	15	0.1	userId	movieId	rating	nan	True	16	nan
25	15	0.1	userId	movieId	rating	drop	True	16	0.805
25	15	0.1	userId	movieId	rating	drop	True	16	0.804
30	20	0.001	userId	movieId	rating	drop	True	16	0.864
30	20	0.5	userId	movieId	rating	drop	True	16	0.998

How to run the Code:

1. Open the 017425395.py file and just run the code.

Note:- Since I used google colab for running my code you may need to alter the path of `ratings.csv` file in the code. Make Sure you have uptodate libraries of PySpark, JRE 8, and Keras.