Universität Freiburg
Institut für Informatik
Dr. Fang Wei-Kleiner
Victor Anthony Arrascue Ayala

Georges-Köhler Allee, Geb. 51
D-79110 Freiburg
fwei@informatik.uni-freiburg.de
arrascue@informatik.uni-freiburg.de

Lab Course: Distributed Computing Using Spark
Summer Term 2021

2. Experiment: Collaborative filtering Recommender with Spark

Deadline: 06.06.2021

Prolog. The goal of this experiment is to build your first recommender system using Spark. For that, you will implement a Collaborative Filtering (CF) Recommender on the citeulike dataset[1]. This recommender applies the Alternating Least Squares (ALS) algorithm to predict missing entries of the users-papers ratings matrix. ALS is a collaborative filtering algorithm based on matrix factorization. Spark ML library provides an implementation for ALS.
The following sources can help you learn more about matrix factorization methods for CF recommenders and the ALS algorithms:

- The DAQL lecture SS2018 (section 3 Model-based Recommender Systems)
- The book: Machine learning with Spark, (chapter 4)
- The Spark's documentation for collaborative filtering:
  `http://spark.apache.org/docs/latest/ml-collaborative-filtering.html`

Exercise 2. 1 (Advanced analysis, 10 points)
Before developing the recommender system, you need to get more understanding about the citeulike dataset. This helps you to better choose, design and evaluate the suitable recommendation algorithm. Write a python program that uses Spark to calculate the following information:

a) Examine how many missing ratings are in the citeulike dataset by calculating the sparsity of the ratings matrix.
b) Study the users' behavior by examining the number of ratings given by the users. For that, calculate and plot the (number of users, number of ratings) rank-frequency distribution.
c) Study the papers popularity by looking at the number or ratings each paper gets. For that, calculate and plot the (number of items, number of ratings) rank-frequency distribution. Does it form a power-law, do you notice a long tail?

[1] In this experiment we will use the dataset *citeulike_dbis_exp2.zip* which can be found on ILIAS

Exercise 2. 2 (Rating matrix preparation, 10 points)
In this exercise, you need to build the ratings matrix as a DataFrame with the following columns:

- *user_id*: a unique positive integer ID for each user.
- *paper_id*: a unique positive integer ID for each item.
- *rating*: a number that represents the ratings given by the user (*user_id*) to the item (*item_id*). It can be one of the following:
  a) Positive rating: an integer with value of 1. We consider all papers appear in the user's library are relevant and have the same "rating": 1.
  b) Unknown rating: an integer with value of 0. Papers which don't appear in the user's library are "unrated papers". They are either (a) irrelevant papers or (b) possibly relevant papers, but the user was not aware of them. For simplicity, we assume "unrated papers" are irrelevant papers and we assign them a rating of 0.

Write a python program that uses spark to load users ratings from the file (*users_libraries.txt*) into a DataFrame structured as described above. Consider the following hints:

a) The *user_hash_id* is a string value. In order to generate the required *user_id*, you need to map each *user_hash_id* to a unique integer number.
b) The unknown ratings are not recorded in the file, but it can be implied. Theoretically, for each user we need to assign a rating of 0 to all unrated papers. But, because the set of unrated papers is very large, it is enough to add the unknown ratings for only a subset of the unrated papers as following: For each user u, add q unknown ratings, where q equals to the number of positive ratings of u. The papers which will receive the unknown ratings (the zeros) should be chosen randomly from the set of u's unrated papers.

Exercise 2. 3 (ALS algorithm, 10 points)
Familiarize yourself with the usage of the ALS algorithm provided by the spark ML library and understand the meaning of the expected parameters. Then, write a python program that employs ALS to fit a model from the dataframe you generated in the last task. Use the learned model to generate top 10 recommendations for all users using the *recommendForAllUsers* method. In addition to that, output the top 10 recommendations for user with user_hash_id = 1eac022a97d683eace8815545ce3153f)

Exercise 2. 4 (Recommender System Model Evaluation, 10 points)
Spark ML provides functionality for evaluating machine learning algorithms. Write the python program that uses Spark to evaluate your ALS-based recommender that you built in the previous task on a held-out test data following the following steps:

a) Split the ratings randomly into training set and test set with 70% and 30% of the ratings respectively.
b) Fit a model on the training set.
c) Calculate the Root Mean Squared Error (RMSE) over the test set.

Exercise 2. 5 (Hyperparameter tuning, 10 points)
ALS algorithm expects a set of hyper-parameters. Investigate and understand the role of the following parameters:

- rank
- maxIter

a) Fix the value of *maxIter* to its default and determine the optimal value for *rank* that optimizes RMSE. To achieve this, use the hyperparameter tuning tools supported by Spark (`CrossValidator`) to search for the optimal value in the following set: $rank \in \{10, 25, 50\}$. Report the best value you find along with the corresponding RMSE.
b) Try to increase the *maxIter*, do you get better results? report your observation.