**Topic: Collaborative Filtering Recommendation Systems**

**Proposal:** In the technology 4.0 era, there is an increasing number of online companies utilizing recommendation systems to increase user interaction and enrich shopping potential. Recommendation systems (or recommendation engines) have the potential to change the way websites communicate with users and allow companies to maximize their ROI based on information gathered from customers. However, choosing an appropriate recommendation model to work with large sparse matrices having a high accuracy rate and less executing time is still a challenge. This research compares two collaborative filtering model-based techniques - SVD matrix factorization from Sklearn and ALS recommendation model from Spark - and applies different ways of improving the efficiency of those machine learning models such as tuning hyper-parameter, using advanced train-test split strategy. From that, the paper provides insights about the optimal recommendation system and appropriate ways to tune the algorithms.

1. [**Exploring data splitting strategies for the evaluation of recommendation models**](https://dl.acm.org/doi/abs/10.1145/3383313.3418479)

*Zaiqiao Meng, Richard McCreadie, Craig Macdonal, and Iadh Ounis.2020. Exploring Data Splitting Strategies for the Evaluation of Recommendation Models. In RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020, Rodrygo L. T. Santos, Leandro Balby Marinho, Elizabeth M. Daly, Li Chen, Kim Falk, Noam Koenigstein, and Edleno SIlva de Moura (Eds.) ACM, 681-686*

This paper not only analyzes recent diverse recommendation literature but also compares three out of six common recommendation data splitting strategies and their impact over seven state-of-the-art recommendation models, including:

* Leave One Last: two commons Leave One Last strategy based on the type of transaction are Leave One Last Item and Leave One Last Basket/Session. Leave One Last is the most popular method to maximize the number of transactions in the dataset that can be used for training. However, the weakness is causing the 'leaking' phenomenon than some undesirable training.
* Temporal Split: there are two variations of this strategy which denotes temporal user and temporal global. Temporal user-based splitting may have future interactions' leaking' into the model during the training due to the variance between train/test boundary users. Meanwhile, temporal global has a much smaller number of users and items after calculating the intersection between the training and testing sets comparing to the temporal user.
* Random Split: this strategy randomly selects the training/test boundary per user. However, it is not reproducible unless the data splits used are release by the author(s)
* User Split: this strategy is a less common approach splitting the dataset by user rather than by interaction. Moreover, the work also suffers from the same leaking issue as the temporal user-based strategy.

Using two commonly metrics (NDCG@10 and Recall@10) for leave one last item, leave one last basket, and global temporal split for seven recommendation models under four scenarios (the combination of two datasets and two evaluation metrics), the paper has two observations:

* The splitting strategy is an essential factor impacting the recommendation performance
* The splitting strategy may favor particular systems.

This paper is related to my research as it provides insight into which data splitting strategy I should use for my Amazon dataset.

1. [**Improving regularized singular value decomposition for collaborative filtering**](https://www.mimuw.edu.pl/~paterek/ap_kdd.pdf)

*Arkadiusz Paterek. 2007. Improving regularized singular value decomposition for collaborative filtering. In Proceedings of KDD Cup and Workshop. 5--8.*

This paper introduces different collaborative filtering techniques to produce a good prediction of users' preferences and combine predictions with linear regression to get better evaluation metric results. The author describes the framework for six predictors used in the evaluation, including regularized SVD of data with missing values, improved regularized SVD, K-means, post-processing results of regularized SVD with KNN, regularized SVD with biases, and post-processing result of SVD with kernel ridge regression. The steps for combining prediction is:

+ Drawing random 1.5%-15% of pair users-ratings dataset as a test set (hold-out set)

+ Using the remaining ratings from the training dataset as a training set where all algorithms are trained

+ Combining the predictions made by each algorithm with linear regression on the test set with selected two-way interactions to create a new method

+ Applying the new idea of decreasing the number of parameters using gradient descent with regularization and early stopping to build two new method

This paper lists out the different types of SVD and two new ways to improve the performance of SVD for collaborative filtering, which is one of the two principal algorithms I apply in my research (SVD from Sklearn and ALS from PySpark).

1. [**Application of improved recommendation system based on spark platform in big data analysis**](https://cit.iict.bas.bg/CIT_2016/v-16-6s/20_paper.pdf)

*Xie, Li, Wenbo Zhou, and Yaosen Li. "Application of Improved Recommendation System Based on Spark Platform in Big Data Analysis." Cybernetics and Information Technologies 16.6 (2016): 245-255.*

This paper designs the parallel implementation process of the recommendation algorithm based on the Spark platform, including collaborative filtering based on users, collaborative filtering based on terms and recommendation algorithm based on the ALS model, and related technology research of recommendation systems. The paper introduces four categories of computing framework including Streaming Spark, Graphx, MLbase, and SparkSQL Implementing the Spark storage algorithm, the comparison between Spark and MapReduce, and a new loss function to improve the shortcomings of the recommendation algorithm based on the ALS model. This paper supports my research on:

* Providing detailed explanations about collaborative filtering algorithms both in general and based on (improved) ALS
* Providing a new loss function to upgrade the ALS model, leading to the better performance of recommendation systems which potentially become my approach to the ALS model in my research
* Using Root Mean Square Error (RMSE) metric to evaluate the accuracy of the prediction aligning with the metric of my SVD model

1. [**An improved ALS recommendation model based on apache spark**](https://link.springer.com/chapter/10.1007/978-981-13-1936-5_33)

*Aljunid MF, Manjaiah DH. An Improved ALS Recommendation Model Based on Apache Spark. International Conference on Soft Computing Systems 2018 Apr 19 (pp. 302-311). Springer, Singapore.*

This paper provides me with a new way to approach ALS (Alternating Least Squares) to have a better performance, using RMSE as the performance metric. Instead of following the original existing flowchart training - running - tuning - evaluating of ALS model, the enhanced model uses a kfoldALS function which makes difference in the first step of splitting data. The more times people run the model with kfoldALS function, the more precise performance score we get. This method is applicable with a big dataset - the two datasets chosen in the paper are the same size as mine in the Capstone Project. Moreover, using a flexible in-memory framework - Apache Spark- leads to more efficient data analyzing and processing.

Moreover, in the Related Work, the authors list a number of new techniques improving different recommendation systems models which become the baseline to compare the effect of the kfoldALS function approach. 

1. [**A Recommendation Engine for Predicting Movie Ratings Using a Big Data Approach**](https://www.mdpi.com/2079-9292/10/10/1215)

*Awan, M.J.; Khan, R.A.; Nobanee, H.; Yasin, A.; Anwar, S.M.; Naseem, U.; Singh, V.P. A Recommendation Engine for Predicting Movie Ratings Using a Big Data Approach. Electronics* ***2021****, 10, 1215.*

This paper introduces a collaborative filtering recommendation system built on matrix factorization and alternating least squared (ALS) model in Spark machine learning libraries applied for a big dataset (Movielens) which improves prediction accuracy to 97% and minimizes the root mean squared errors (RMSE). To avoid and solve the cold-start, scalability, and sparsity in recommendation system, the technique utilizes elements extracted from the new user matrix by multiplying the user’s item-based matrix to attain the required point in less time. Specifically, from the original users-items matrix (Matrix R),  how ALS algorithm operates:

* Factoring Matrix R  into two smaller matrices (Matrices U & P)
* Filling the sparse value in matrice U & P by using random numbers and calculating the error term according to the error formula
* Alternating back and forth between matrix U and matrix P to gradually minimize the error
* From that, all the blank space in the old matrix R has been filled. One strength of ALS is it seeks to leverage ratings from similar user while working with sparse matrix

The paper emphasizes the effectiveness of using ALS algorithm and fetching data through Spark ML which supports my choice in using ALS model for the recommendation system capstone project.

1. [**SVD-based incremental approaches for recommender systems**](https://www.sciencedirect.com/science/article/pii/S0022000014001706)

*Zhou, X.; He, J.; Huang, G.; Zhang, Y. SVD-based incremental approaches for recommender systems. J. Comput. Syst. Sci. 2015, 81, 717–733.*

This paper introduces an incremental recommendation system model based on singular value decomposition (SVD) algorithm called Incremental ApproSVD with the aim of improving running time and accuracy of predicting new items entered dynamically. The new algorithm is the combination between ApproSVD algorithm and Incremenral SVD one. Comparing to other clustering or data dimensionality reduction methods including K-means clustering, minimum spanning tree or partition around medoids (PAM) which solve the huge amount of data with quick response time and the sparsity problem as offline computation, Incremental ApproSVD can handle online and dynamic problems more efficiently. The most important technique of Incremental ApproSVD is choosing column sampling probabilities, specifically adopting column sampling to reduce the column number.

As my Capstone Project focuses on comparing SVD and ALS algorithm in two aspects: accuracy prediction rate measure by the RSME and running time which are the same metrics to measure the improvement of applying Incremental ApproSVD in this paper, it gives me insights to apply the changing in sampling column sizes and  Kfold split in my SVD algorithm. 

1. [**Algorithmic acceleration of parallel ALS for collaborative filtering: Speeding up distributed big data recommendation in spark**](https://ieeexplore.ieee.org/abstract/document/7384354/)

*M. Winlaw, M.B. Hynes, A. Caterini, H. De Sterck, Algorithmic acceleration of parallel ALS for collaborative filtering: speeding up distributed big data recom- mendation in spark, in: Proceedings of the IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS), 2015, IEEE, 2015, pp. 682–691.*

This paper solves the optimization problem of collaborative filtering recommendation system mode with an approach to accelerate the convergence of parallel ALS-based algorithm using a nonlinear conjugate gradient (NCG)  in the Apache Spark distributed data processing environment. The reasons why this paper approach the combined version of ALS-NCG instead of using them separately:

* Nonlinear conjugate gradient (NCG) algorithm use by itself is slow to converge to a solution of optimization problem; therefore, cannot work as an alternative for ALS algorithm.
* Parallel versions of collaborative filtering and recommendation are great interest in the era of bid data. When combine, ALS works as a nonlinear preconditioner for NCG.
* Using ALS standalone, it takes 100s to increase the ranking accuracy from 75% ro 100% while it is seven times faster if we use ALS-NCG (14s)

This paper provide a possible way to future recommendation for my paper when accurate solution are desired, we can apply additional nonlinear conjugate gradient wrapper to improve the running time of  ALS-based model.