**Recommender Systems (Virtual Presentation)**

# Introduction

This project focuses on *recommender systems*, a data science application widely used in the real world. This project requires development and implementation of appropriate solutions, and present the results (virtually).

# Integrity

All projects will be checked with plagiarism-detection software; any project found to have plagiarised will be subject to disciplinary actions. Plagiarism includes, e.g., submitting code that is not your own or submitting text that is not your own. All plagiarisms will be penalised; there are no exceptions and no excuses.

# General Requirements

This section contains information about the general requirements that your project submission must meet. **Please read all requirements carefully before you start.**

* You *must* do the assignment in **Jupyter Notebook** (available in Anaconda).
* You *must* use the appropriate Python functions (or methods) to complete the tasks and you must figure out (and determine) what functions should be used by yourself. You may refer to relevant Python (or Python Library) official documentations as needed. You may use any Python Library and any function as appropriate.
* Please ensure that your submission follows the file naming rules specified in the tasks below. File names are case sensitive, i.e. if it is specified that the file name is gryphon, then that is exactly the file name you should submit; Gryphon, GRYPHON, griffin, and anything else but gryphon will not be accepted.

# The Dataset

This assignment deals with movie recommendation. The dataset to be used throughout the assignment is the MovieLens 1M Dataset. A copy of this dataset ml-1m.zip is provided. The README.txt file therein provides details about the dataset. Besides, different users may rate different groups of movies.

# Task 1: User-based Collaborative Filtering

In this task you need to develop, implement, and evaluate user-based (i.e., user-user) collaborative filtering that uses KNN (k-nearest neighbour), i.e., KNN-based Collaborative Filtering. **Randomly** choose one user (as the active user) and predict this user’s ratings on movies. Note that in the given dataset, the user might have only rated some of the movies.

Specific requirements include:

* Choose an appropriate similarity metric (and other parameters).
* Implement the approach in Python. Add detailed comments for the code (to explain your implementation).
* Study the impact of the parameter K (of KNN), with at least 5 different values.
* Use RMSE (root-mean-square error) as the metric for evaluation.
* Summarise the results/findings concisely in word document & slides (and presentation).

Tips on creating comments in Python: [https://www.w3schools.com/python/python](https://www.w3schools.com/python/python_comments.asp) [comments.asp.](https://www.w3schools.com/python/python_comments.asp)

# Task 2: Item-based Filtering

In this task you need to develop, implement, and evaluate item-based (i.e., item-item) (collaborative) filtering that uses KNN. **Randomly** choose one item (i.e., movie) and predict users’ ratings on this movie. Note that in the given dataset, it is possible that some users didn’t rate this movie. Specific requirements include:

* Choose an appropriate value for the parameter K (of KNN) (and other parameters).
* Implement the approach in Python. Add detailed comments for the code (to explain your implementation).
* Compare the performance of at least 2 similarity metrics.
* Use RMSE (root-mean-square error) as the metric for evaluation.
* Summarise the results/findings concisely in word document and slides (presentation).

# Task 3: A Better Recommender System

In this task you are required to identify/propose a better solution to improve the recommendation quality and evaluate its performance in comparison with two baseline methods.

**Task 3.1.** Develop the “new” solution for movie recommendation. This can be done in either of the following ways:

* Option 1: Search and read related publications on recommender systems. Identify/choose one solution that can be used for movie recommendation and will potentially deliver better recommendation quality. The solution must have been published in a peer-reviewed journal or conference. It can be based on KNN or not.
* Option 2: Based on an extensive literature review (conducted by yourself), propose a new algorithm by yourself. In this case, the idea is yours. You need to justify its originality and novelty. A strong list of references must be provided. Note that parameter tuning only cannot be considered as a (new) approach.

Specific requirements include:

* Describe the idea (in your own words) clearly and precisely in word document and slides (presentation). Cite references wherever necessary.
* In slides, explicitly give the source of the solution (if you choose Option 1) or provide a strong list of references (Option 2).
* Name the solution as Option1RecSys if you choose Option 1 or Option2RecSys for Option 2.
* Implement the approach in Python. Add detailed comments for the code (to explain your implementation as well as details of the algorithm).

**Task 3.2.** Randomly choose 5 users (from who have rated more than 100 movies each) and recommend Top-30 movies (to each user). Use AP (Average Precision) and NDCG (Normalized Discounted Cumulative Gain) as evaluation metrics. Compare the performance of three solutions:

* Movie Average: recommends items with the highest average ratings. Name it as MovieAvg.
* KNN-based Collaborative Filtering: the approach developed in the above Task 1. Choose the optimal parameters (based on results of Task 1). Name it as KNNCF.
* The solution developed in the above Task 3.1, i.e., Option1RecSys or Option2RecSys.
* Use one appropriate graph/chart to visualise the results of each metric, respectively (i.e., one graph for AP and another for NDCG). Summarise the results/findings concisely in slides (presentation).

**It is required that the solution developed in Task 3.1 (Option1RecSys or Option2RecSys) must achieve the best (overall) performance.**

# Task 4: Presentation

In this assignment, you need to create word document and design slides (for the above tasks and results). Your slides could include, e.g., but not limited to:

* a cover page/slide & project info,
* a concise outline, key results and findings of Task 1,
* a concise outline, key results and findings of Task 2,
* clear and complete description of the “new” solution developed in Task 3.1 (with proper citations),
* literature review (with proper citations), if applicable,
* necessary/key details of the algorithm of Option1RecSys or Option2RecSys,
* key results, visulisation, and findings of Task 3.2,
* a list of references.

**The following requirements must be strictly met:**

* The slides should be no more than **10** pages in total (that is, no more than **10** slides)
* There is no template for slides.

You need to submit the following files:

**Jupyter Notebook file** containing your Python commands, named assignment3.ipynb. **Please use the provided solution template to organise your code**: *assignment3 TEMPLATE.ipynb*

**#For the Notebook file, please make sure to clean them and remove any unnecessary lines of code (cells). Comments are required in this assignment. Follow these steps before submission:**

1. Main menu → Kernel → Restart & Run All
2. Wait till you see the output displayed properly. You should see all the data printed and graphs displayed.