# REPORT

Implemented a Movie recommendation system on (Movielens dataset) that used the previous implementation of KNN algorithm. Dataset used is ml-latest (size 265MB) https://grouplens.org/datasets/movielens/latest/. This dataset describes 5-star rating and free-text tagging activity from [MovieLens](http://movielens.org), a movie recommendation service. It contains 27753444 ratings and 1108997 tag applications across 58098 movies. The data are contained in the files `genome-scores.csv`, `genome-tags.csv`, `links.csv`, `movies.csv`, `ratings.csv` and `tags.csv`. We use only the `movies.csv`, `ratings.csv`. We need to take only movies that have been rated with a frequency of greater than 50 times which is the threshold so that only most popular movies are recommended. We drop the inactive users and unpopular movies which include many movies with 0 rating. A movie is randomly selected and compared with 10 nearest neighbors based on the user input distance measure. It then ranks its distances and returns the top k nearest neighbor movies as the most similar movie recommendations.

## Steps to run code:

- Download the dataset from the link mentioned above, unzip the dataset, and use the (movies.csv and ratings.csv).
- Run the Recommender.ipynb using Jupyter Notebook.
- Kindly enter distance metric to be used like Euclidean, Minkowski, Cosine, Chebyshev, Jaccard and Manhattan as a string format. I have used my previous implementation of KNN.

- ## Your approach to use KNN as a recommender system:

  KNN algorithm relies on item feature similarity rather than data distribution. When a KNN makes a prediction about a movie, it will calculate the “distance” (distance metrics) between the target movie and every other movie in its database. It then ranks its distances and returns the top k nearest neighbor movies as the most similar movie recommendations.

- ## What is the maximum dataset that your recommender system can use?

  The entire 27m dataset cannot be used as it gives memory error hence, I have used threshold so that only most popular movies are recommended. The shape of flattened dataset is (4802, 7758)

- ## What is the time complexity of your recommender system?

  O(n×m), where n is the number of training examples and m is the number of dimensions in the training set. Assuming n >> m, the complexity of the brute-force nearest neighbor search is O(n).

- ## What is the performance of your recommender system?

  The performance of the system is based on distance measures.

- ## Is there a way to scale-up your recommender system to work with very large datasets?

  Storing sparse matrix wastes space when database accommodates millions of users. So, compression of matrix is needed. For large datasets organizations like Amazon or Netflix, which rely heavily on recommender systems to suggest items and movies to their users use single value decomposition. We also require better system requirements.

References given below:

1. https://github.com/nikitaa30/Recommender-Systems/blob/master/knn_recommender.py
2. Dataset: https://grouplens.org/datasets/movielens/latest/
3. https://sebastianraschka.com/pdf/lecture-notes/stat479fs18/02_knn_notes.pdf
4. https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from-scratch/
5. https://machinelearningmastery.com/k-fold-cross-validation/
6. https://machinelearningmastery.com/distance-measures-for-machine-learning/
7. https://aiaspirant.com/distance-similarity-measures-in-machine-learning/