BT4222 Midterm Proposal

Recommendation System 7



| **Name** | **Matriculation Number** |
| --- | --- |
| Aarushi Mundra | A0246207J |
| Dylan Wo Qiying | A0233634J |
| Kaavya Senthil Kumar | A0246227E |
| Kenny Low Sheng Wei | A0233869N |
| Ng Wei Han | A0234338E |
| Tasneem Benazir | A0238339W |

**Content Page**

**Section 1: Existing Project Reviews**

**1.1 Business Problems and ML Goals**

**1.2 Highlights on data processing, feature engineering and sampling**

**1.3 Typical Machine Learning methods that have been adopted**

**1.4 Main ML related challenges of topic**

**Section 2: Defining the ML Problem**

**2.1 ML Problem, ML Goal and Linkage**

**2.2 Justification of datasets**

**2.3 Subtasks to be Accomplished**

**2.4 Connection to Existing Project Reviews and Potential Contributions**

**1.1 Business Problems and ML Goals**

**Movie1**

Business Goal: Evaluate the quality of search results and offer user items that are highly relevant to their queries or aligned with their browsing history.

ML Goal: Predict individual preferences and behavior and provide tailored recommendations.

Main Dataset :

1. `[movies\_metadata.csv](https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system/input)`: Contains information on 45,000 movies featured in the MovieLens dataset. Features include posters, backdrops, budget, revenue, release dates, languages, production countries and companies.

**eCommerce2**

Business Goal: Boost sales through having more tailored recommendations according to consumer’s on-site activity and purchase history.

ML Goal: Predict e-commerce clicks, cart additions and orders and provide apt recommendations through these events.

Main Dataset:

1. [OTTO Kaggle competition](https://www.kaggle.com/competitions/otto-recommender-system/overview): `train.jsonl` contains data of “click”, “add” and “ordered” events in ecommerce sites.

**1.2 Highlights on data processing, feature engineering and sampling**

**Weighted Rating (Movie)3**

* Involves the use of calculating a movie’s rating based on number of votes.
* Since average ratings would not be an accurate gauge of popularity for movies with comparatively low number of ratings, we can look into applying different weighted rating scores and test for the best performance.

**Minimum number of votes required for the movie to be used in model (Movie)**

* Can be used to filter out more rarely rated movies in our dataset.
* Avoids cold starts where infrequently rated movies have limited data to inform recommendations. Also increases relevance and data quality.

**Composite feature generation (eCommerce)**

* First place solution of the OTTO Kaggle competition included the creation of new composite features e.g. time between events based on the given time stamps.
* We can experiment on whether such techniques could improve our model accuracy, since we also have similar movie review timestamp data.

**1.3 Typical Machine Learning methods that have been adopted**

**Demographic Filtering (DF) (Movie)**

* Recommends movies to users with similar demographic features.
  + Although this might perform well, it is often overly simplistic as it only takes into account more popular and critically acclaimed films.

**Content Based Filtering (CBF) (Movie)**

* Recommendations are made based on each user’s historical preferences. Some of the models used were TF-IDF (descriptions/content), CountVectorizer (genres/keywords) and cosine similarity.
  + Performs well within genres but not across genres.

**Memory-based Collaborative Filtering (CF) (Movie)**

* Recommendations are made based on sets of similarity scores computed using cosine similarity or pearson correlation. There are 2 types of such:

1. User-based CF: Recommends items to a user that users with similar preferences have liked.
2. Item-based CF: Recommends items to a user based on their similarity with the items that the target user rated.
   * Performs well for complex items such as music and movies where variations in taste are dependent on the variation in preferences.

**Model-Based Collaborative Filtering (eCommerce / YouTube / Netflix)**

* Highest accuracy models in the OTTO Kaggle competition4 generated co-visitation matrices based on the unique sequences of events, then fed these matrices into a DNN for training, maximizing the recall@k.
  + Despite having good performances, such models require large amounts of computing power and data, which might not be suitable for our dataset.
* YouTube’s candidate generation engine5 also utilizes a DNN trained on embedded watch, search and demographic inputs to shrink the item corpus before re-ranking them for its user base.
  + Similarly, this requires large amounts of computing power.
* K nearest neighbors6 (KNN) approach was also frequently mentioned in many model based CF articles.
  + This model requires the least computing power and is also the most explainable, and hence could be most applicable for our dataset.
* Matrix Factorization using Singular Value Decomposition (SVD)/Alternative Least Squares (ALS)7,8
  + SVD is effective in capturing latent factors and patterns in dense matrices, while ALS handles sparse matrices well and scales well with large datasets.

**Re-Ranker Models**

* Top solutions in OTTO competition9 used re-ranking models like Light GBM models, tree-based XGBoost models.
  + These Learning to Rank models require large amounts of computing power, and might not be applicable.
* Other re-ranking models include freshness-based models such as Bayesian Personalized Ranking10, or NN-based re-rankers

**1.4 Main ML related challenges of topic**

**Demographic Filtering**

* Model is unable to take into account user sentiments and provides recommendations based on features of the movie itself.
* The movie recommendation system (RS) involved other models to support the recommendations system.

**Content Based Filtering**

* Quality of searches can be improved with the usage of better metadata and it is not capable of capturing tastes and providing recommendations across genres.
* Movie RS suggested the use of CF which can better capture the personal tastes and biases of a user.

**Collaborative Filtering**

* User based filtering: User's preference can change over time, precomputing the matrix based on their neighboring users may lead to bad performance. To tackle this problem, we can apply item-based CF.
* Item based filtering:
  + Suffers from the issue of scalability. This is because the computation grows with every new customer and new movie in the dataset.
  + Suffers from sparsity. In extreme cases, there could be millions of users and the similarity between two fairly different movies could be very high simply because they have similar rank for the only user who ranked them both.
  + To solve both the issues for item based filtering, the model in the movie RS uses a latent factor model, like Single Value Decomposition, to capture the similarity between users and the items.

**Cold Start Problem (Ecommerce/Movie/Youtube)**

* CF models often have cold start issues, so we might want to employ more of a hybrid approach, similar to the [YouTube Recommendations System](https://www.youtube.com/watch?v=5R0ZMx2kIPw) case study above.

**Many other problems and solutions for Grey Sheep, Shilling attack, Synonym, etc, are not shown in the 2 projects but rather in this reading11.**

**2.1 ML Problem, ML Goal and Linkage**

There are certain drawbacks to CBF, CF and demographic models, when used independently as mentioned in section 1.4, that affect the performance of each RS.

Thus, the decision to make is how to most effectively combine these 3 models into a hybrid system by choosing relevant metrics to address the inherent limitations of each model and provide more precise movie recommendations and at the same time keep the model scalable.

If we are able to find the optimal hybridization, we can achieve the best of all worlds and create a more robust system that is more accurate, similar to how the conventional ensemble method in machine learning works.

**2.2 Justification of dataset(s)**

The dataset we chose consists of the following files:

1. movies\_metadata.csv: The main Movies Metadata file. Contains 45,000 movies featured in the Full MovieLens dataset. Features include posters, backdrops, budget, revenue, release dates, languages, production countries and companies.
2. keywords.csv: Contains the movie plot keywords, in stringified JSON Object form.
3. credits.csv: Consists of cast and crew Information for all our movies, in stringified JSON Object form.
4. ratings\_small.csv: The subset of 100,000 ratings from 700 users on 9,000 movies. Contains user ID, movie ID, rating and timestamp.

For CBF, we need sufficient information about a movie’s content, which can be extracted from movies\_metadata.csv, keywords.csv and credits.csv. We will also need data about each user’s preferences, which can be deduced from their rating and timestamp from ratings\_small.csv.

Likewise, for both user-based and item-based CF to work, we will also need information about a movie’s content and all user’s preferences.

**2.3 Subtasks to be Accomplished**

* **Exploratory Data Analysis**: Gain more understandings from our datasets
* **Data preprocessing**: Data cleaning, missing values, outlier removing
* **Splitting**: 70% Train | 20% Validate | 10% Test
* **Feature engineering**: Experiment with composite feature generation. We will also check for highly correlated features and select only important features to use.
* **Transformation**: Scaling or encoding of some features to best suit our models.
* **CBF**: Implement CBF, exploring the use of cosine similarity, euclidean distance and Pearson’s similarity to see which gives the best performance.
* **Memory-based CF**: Implement user-user and item-item CF using KNN.
* **~~Model-based CF~~**~~: Implement model-based CF, using matrix factorization methods which includes both ALS and SVD models.~~
* **Tuning of Model Hyperparameters**: Perform hyperparameter tuning on the model to find best hyperparameters and optimize its performance.
* **~~Evaluation of model-based CF~~**~~: Conduct a comparative evaluation to determine whether SVD or ALS yields superior performance, with the dataset we have.~~
* **Hybrid RS**: Form a Hybrid RS that leverages CBF, CF and DF systems by balancing the strengths and addressing the inherent limitations in each individual system.
* **Re-ranking**: Employ re-rankers for the candidate recommendations generated by the RS. Re-ranking step aims to refine the initial recommendations by applying additional algorithms to improve their relevance and effectiveness.
* **Evaluate re-ranking result**: The final step in our pipeline would be evaluating our results pre and post re-ranking using evaluation methods like recall@K.

**2.4 Connections to Existing Project Reviews and Potential Contributions**

**Connections to Existing Project Reviews**

* We will experiment with composite feature engineering functions.
* Similar to the existing Movie project, we will employ the use of DF, CBF, and CF (memory-based and SVD).
* Similar to this article12 on Netflix Recommendations Systems, we plan to use matrix factorization models, like SVD and ALS, as it can readily accept varying confidence levels and keep only the true interaction portion of the data to factor modeling, by trying to identify the explainable biases of the model.

**Potential Contributions**

* Hyperparameter tuning in model-based CF where we plan to optimize key model parameters, in order to improve accuracy and performance.
* Test out for the demeaning of the respective Bayesian averages from all rankings to account for the skew in sparse rankings.
* There are many ways to construct hybrid RS. We would want to test and evaluate these possible systems to find which one fits our dataset best.
* Experimentation with re-ranking models, if resources permits.
* Use KNN taking into account the mean ratings of each user, for user-user CF.

**References**

1. **Ibtesam Ahmed 2020. Getting Started with a Movie Recommendation System. Retrieved September 18 2023, From** [Kaggle - Getting Started with a Movie Recommendation System](https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system/notebook)
2. **OTTO – Multi-Objective Recommender System, 2023. Retrieved September 19 2023, From** [Kaggle - eCommerce OTTO Multi-Objective Recommender System Competition](https://www.kaggle.com/competitions/otto-recommender-system/overview)
3. **Benjamin Bengfort, 2017. Computing Bayesian Estimate of Start Rating Means, Retrieved September 18 2023, From** [Computing Bayesian Estimate of Star Rating Means](https://medium.com/district-data-labs/computing-a-bayesian-estimate-of-star-rating-means-651496a890ab)
4. **OTTO – Multi-Objective Recommender System, 2023, Retrieved September 20 2023, From** [First Place Solution Discussion for the OTTO Competition](https://www.kaggle.com/competitions/otto-recommender-system/discussion/384022)
5. **AI-First, YouTube Recommendation System Case Study, 2019 Retrieved October 1 2023, From** [YouTube Candidate Generation Model Explained](https://www.youtube.com/watch?v=5R0ZMx2kIPw)
6. **Movie Recommendation and Rating Prediction using K-Nearest Neighbors, 2023, Retrieved September 17 2023, from** [K-Nearest-Neighbors for Recommendation Systems](https://www.analyticsvidhya.com/blog/2020/08/recommendation-system-k-nearest-neighbors/)
7. **Matrix Factorization Algorithms, Retrieved September 16 2023, From** [Matrix Factorization Algorithms](https://surprise.readthedocs.io/en/stable/matrix_factorization.html)
8. **MRKMAKR, 2018, Alternating Least Squares Matrix Factorization Explained, Retrieved September 25 2023, From** [Alternating Least Squares Matrix Factorization Explained](https://towardsdatascience.com/prototyping-a-recommender-system-step-by-step-part-2-alternating-least-square-als-matrix-4a76c58714a1)
9. **OTTO – Multi-Objective Recommender System, 2023, Retrieved 29 September 2023, From** [First Place Solution Discussion for the OTTO Competition](https://www.kaggle.com/competitions/otto-recommender-system/discussion/384022)
10. **Re-Ranking Recommendation System, Retrieve 27 September 2023, From** [Re-Ranking Recommendation Systems](https://aman.ai/recsys/re-ranking/#:~:text=Freshness%2Dbased%20re%2Dranking%20can,improve%20the%20diversity%20of%20recommendations.)
11. **Roy, D., Dutta, M., A systematic review and research perspective on recommender systems, Retrieved 3 October 2023, From** [A systematic review and research perspective on recommender systems](https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00592-5#citeas)
12. **Yehuda Koren & Chris Volinsky, 2009. Matrix Factorization Techniques For Recommender Systems, Retrieved 30 September 2023, From** [Netflix Recommendation System Article](https://datajobs.com/data-science-repo/Recommender-Systems-%5BNetflix%5D.pdf)