Capstone Project Proposal

Recommendation System

1. **Team Information**

| Name | ID | Role |
| --- | --- | --- |
| Vo Dinh Dat | 20214890 | Leader |
| Doan The Vinh | 20210940 | Member |
| Doan Ngoc Cuong | 20210141 | Member |
| Le Trung Kien | 20214907 | Member |
| Pham Quang Trung | 20214935 | Member |

Team 1 Information

1. **Proposal**
2. **Problem Definition**

* Online shoppers have their pick of millions of products from large retailers. While such variety may be impressive, having so many options to explore can be overwhelming, resulting in shoppers leaving with empty carts. This neither benefits shoppers seeking to make a purchase nor retailers that missed out on sales. This is one reason online retailers rely on recommender systems to guide shoppers to products that best match their interests and motivations. Using data science to enhance retailers' ability to predict which products each customer actually wants to see, add to their cart, and order at any given moment of their visit in real-time could improve your customer experience the next time you shop online with your favorite retailer. The goal of this project is to build a multi-objective recommender system based on previous events that the user does in each session.

1. **Dataset**

* The dataset we will be using is: **OTTO – Multi-Objective Recommender System**, here are some key features about the dataset:
  + 12M real-world anonymized user sessions
  + 220M events, consisting of clicks, carts and orders
  + 1.8M unique articles in the catalog
  + Ready to use data in .jsonl format
  + Evaluation metrics for multi-objective optimization

1. **Method**
2. Collaborative Filtering

* **Collaborative filtering** algorithms recommend items based on preference information from many users. This approach uses similarity of user preference behavior, given previous interactions between users and items, recommender algorithms learn to predict future interaction.

1. Content Filtering

* **Content filtering** uses the attributes or features of an item to recommend other items similar to the user’s preferences. This approach is based on similarity of item and user features, given information about a user and items they have interacted with, and model the likelihood of a new interaction.

1. XGBoost Ranker

* **Learning-to-rank** aims to train a model that arranges a set of query results into an ordered list. XGBoost ranker can be used to generate ranked lists of items that users most likely will interact with, and XGBoost will be suitable for large-volume data, such as the data we are using with millions user sessions and millions events.

1. **Evaluation**

* The result will be evaluated based on ***Recall@20*** for each action **type**, and the three recall values are weight-averaged:

where ***R*** is defined as

and 𝑁 is the total number of sessions in the test set, and predicted **aids** are the predictions for each session-type (e.g., each row in the submission file) truncated after the first 20 predictions.

* Our goal is to at least make it to the **top 200** of the Kaggle Private Leaderboard.

1. **Schedule**

| Task | Description | Deadline |
| --- | --- | --- |
| Project Setup and Data Exploration | 1. Set up a development environment (libraries, tools, version control). 2. Explore the OTTO dataset (understand data structure, distribution, event types). 3. Clean and preprocess the dataset (handle missing values, data transformation). | 13th Oct |
| Feature Engineering & Initial Model Selection | 1. Feature engineering (session-level features, interaction types, time-based features). 2. Select baseline models (Collaborative Filtering, Content Filtering). 3. Implement basic collaborative filtering and content-based filtering models. | 20th Oct |
| Model Training & Evaluation (Collaborative and Content Filtering) | 1. Train collaborative filtering and content-based filtering models. 2. Perform cross-validation on recall metrics (Recall@20). 3. Fine-tune hyperparameters for better performance. | 27th Oct |
| Implement XGBoost Ranker & Advanced Techniques | 1. Implement the XGBoost Ranker for learning-to-rank on the dataset. 2. Apply advanced feature selection techniques. 3. Start combining predictions from different models using ensembling methods. | 3rd Nov |
| Model Optimization & Ensembling | 1. Optimize the XGBoost model by tuning hyperparameters. 2. Explore stacking or blending techniques for ensembling the models. 3. Analyze model performance on validation data. | 10th Nov |
| Final Model Testing & Leaderboard Submission | 1. Test final model on the holdout set. 2. Prepare Kaggle submission file. 3. Submit the model to the Kaggle leaderboard for feedback. | 17th Nov |