**Project Proposal: Movie Recommendation System Using Collaborative Filtering**

**Members:**

**Salik Ahmed Bhatti 22k-5114**

**Abdullah Nadeem 22k-5033**

**1. Project Overview**

The project aims to develop a personalized movie recommendation system using **Collaborative Filtering (CF)** techniques. This system will predict movies that a user may enjoy based on the preferences of other users with similar tastes. Collaborative filtering will be implemented using two main approaches: **User-based Collaborative Filtering** and **Item-based Collaborative Filtering**. The ultimate goal is to enhance user engagement by recommending movies that align with their interests without requiring explicit knowledge of movie content.

**2. Problem Statement**

With the rapid growth of movie databases, it becomes challenging for users to discover new movies that match their preferences. Traditional recommendation systems often rely on content-based methods, which require deep knowledge about the content itself. Collaborative filtering, on the other hand, leverages user behavior and interactions to generate recommendations, thus avoiding the need for manual content classification.

By implementing collaborative filtering techniques, the goal is to create a scalable and effective recommendation system that can suggest movies tailored to users' preferences based solely on user ratings.

**3. Objectives**

* **Preprocessing of Data**: Clean and transform the dataset into a suitable format for collaborative filtering.
* **Implementation of Collaborative Filtering**: Build user-based and item-based collaborative filtering models to provide movie recommendations.
* **Evaluation**: Evaluate the system's performance based on common metrics like **Precision**, **Recall**, **RMSE**, and **MAE**.
* **Recommendation Delivery**: Develop a basic mechanism to deliver movie recommendations to users.

**4. Methodology**

The proposed solution will follow a step-by-step approach to develop the recommender system:

* **Data Collection**:  
  We will use the **MovieLens dataset**, which includes user ratings for movies, along with movie titles and metadata. This dataset contains millions of ratings across thousands of movies and users, providing an ideal foundation for the recommender system.
* **Data Preprocessing**:
* Load and clean the data, removing any missing or irrelevant entries.
* Merge the movie and rating datasets to create a unified dataset with user ratings and movie titles.
* Create a **user-item matrix**, where rows represent users and columns represent movies. The values will be the ratings given by users to movies.
* **Collaborative Filtering Algorithms**:
* **User-Based Collaborative Filtering**:
* Calculate **cosine similarity** between users based on their rating patterns. Users who have similar ratings for movies will be identified as similar users.
* Recommend movies that similar users have rated highly, but the target user hasn't rated yet.
* **Item-Based Collaborative Filtering**:
* Calculate **cosine similarity** between movies based on user ratings. Movies that are rated similarly by users are considered similar.
* Recommend movies similar to those that the user has rated highly.
* **Model Evaluation**:
* Use **Precision**, **Recall**, **Root Mean Squared Error (RMSE)**, and **Mean Absolute Error (MAE)** to evaluate the performance of the system and the accuracy of the recommendations.
* **Precision and Recall** will help assess how relevant and complete the recommendations are, while **RMSE and MAE** will evaluate the error in predicted ratings.
* **Recommendation Generation**:
* For **user-based collaborative filtering**, we will recommend movies that similar users have rated highly.
* For **item-based collaborative filtering**, we will recommend movies similar to those the user has already rated positively.
* **Interface Development (Optional)**: A simple **Flask-based**
* **web interface** will be created where users can input their preferences, and the system will return personalized movie recommendations in real time.
* The interface will allow users to rate movies, and recommendations will be dynamically updated based on user feedback.

**5. Technologies and Tools**

* **Programming Language**: Python 3.x
* **Libraries**:
* **Pandas, Numpy**: For data manipulation and numerical operations.
* **Scikit-learn**: For calculating cosine similarity and other machine learning tasks.
* **Flask** (optional): For building the web interface.
* **Matplotlib, Seaborn**: For data visualization and evaluation metric plotting.
* **Dataset**: MovieLens Dataset

**6. Expected Challenges and Solutions**

* **Data Sparsity**: User-item matrices often have many missing entries, which can reduce the effectiveness of collaborative filtering. We will handle this by ensuring proper preprocessing, and the use of techniques like **nearest neighbor imputation** or **matrix factorization**.
* **Scalability**: As the dataset grows, calculating similarity matrices can become computationally expensive. We will optimize the solution by using efficient algorithms such as **nearest neighbors** or **approximate nearest neighbors (ANN)** to speed up calculations.
* **Cold Start Problem**: New users or movies with limited ratings pose a challenge. To address this, we will explore hybrid models or integrate content-based filtering as an enhancement.
* **Evaluation and Tuning**: Fine-tuning the parameters of the collaborative filtering models will require careful evaluation and adjustment to improve performance.

**7. Deliverables**

* **Collaborative Filtering Model**: A fully trained and evaluated collaborative filtering recommendation system.
* **Evaluation Metrics**: Performance metrics, including Precision, Recall, RMSE, and MAE, to assess model accuracy.
* **Web Interface (Optional)**: A simple Flask-based interface where users can interact with the recommendation system.