**DEC Mini Project Report**

1. **Project Title**:

**- Movie Recommender System: Content-Based Recommender System**

2. **Team Members**:

- Team Member 1: Dittee Salian, roll no.:43

- Team Member 2: Tejas Redkar, roll no.: 44

- Team Member 3: Gayatrini Neogi, roll no.: 49

- Team Member 4: Atharva Chaher, roll no.:50

3. **Overview of Various Technologies Used and Datasets**:

- Technologies Used:

- Programming Language: Python

- Frontend:

- Streamlit

- Backend:

- Framework: Flask/Django [Choose one]

- Database: Kaggle

- Libraries: scikit-learn, pandas, NumPy, NLTK (Natural Language Toolkit)

- Datasets Used:

- Movie Dataset: [Specify the source, e.g., IMDb dataset, MovieLens dataset]

- [Any additional datasets used for content analysis or feature engineering]

4. **Workflow/Architecture Diagram with Explanation:**

There are mostly three ways to build a recommendation engine:

1. Popularity based recommendation engine
2. Content based recommendation engine
3. Collaborative filtering based recommendation engine.

**Popularity based recommendation engine**: It is the simplest kind of recommendation engine that we will come across. The trending list we see on YouTube or Netflix is based on this algorithm. It keeps a track of view counts for each movie/video and then lists movies based on views in descending order.

**Content based recommendation engine:** This type of recommendation system takes in a movie that a user currently likes as input. Then it analyses the movie’s contents to find other movies with similar content. This system uses item metadata, such as genre, director, description, actors, etc. for movies, to make these recommendations. The general idea behind these recommender systems is that if a person likes a particular item, he or she will also like an item that is like it.

**Collaborative filtering based recommendation engine**: This algorithm at first tries to find similar users based on their activities and preferences. One typical application of this algorithm can be seen in the Amazon e-commerce platform, where you get to see the “Customers who viewed this item also viewed” and “Customers who bought this item also bought” lists.

In this project, we use the content based recommendation engine.

**Flow Process**

1. **Data Sourcing**: The first dataset contains the following features:

* budget — The budget in which the movie was made.
* genre — The genre of the movie, Action, Comedy, Thriller, etc.
* homepage — A link to the homepage of the movie.
* id — This is the movie\_id as in the first dataset.
* keywords — The keywords or tags related to the movie.
* original\_language — The language in which the movie was made.
* original\_title — The title of the movie before translation or adaptation.
* overview — A brief description of the movie.
* popularity — A numeric quantity specifying the movie’s popularity.
* production\_companies — The production house of the movie.
* production\_countries — The country in which it was produced.
* release\_date — The date on which it was released.
* revenue — The worldwide revenue generated by the movie.
* runtime — The running time of the movie in minutes.
* status — “Released” or “Rumoured”.
* tagline — Movie’s tagline.
* title — Title of the movie.
* vote\_average — average ratings the movie received.
* vote\_count — the count of votes received.

The second dataset contains the following features: -

* movie\_id — A unique identifier for each movie.
* title: Title of the movie.
* cast — The name of the lead and supporting actors.
* crew — The name of the Director, Editor, Composer, Writer, etc.

**2. Data Preparation**:

Data preparation includes data wrangling, cleaning, and removal of outliers. First, we merge the two datasets using the merge () method. In the datasets, we had two columns containing null values — homepage and tagline. Later, we used the ast (Abstract Syntax Tree) module to change the datatype of some features with the list of strings to the python list for a better understanding of the dataset. The columns that are unnecessary for the remaining project procedures can finally be removed.

**3. Model Training**: The cleaned data is fed into the model, so that the model could learn the patterns from the dataset. To build the model, we use nltk (Natural Language Toolkit), which is a standard python library that provides a set of diverse algorithms for NLP (Natural Language Processing). We then used the PorterStemmer () module to normalize the feature column in the dataframe. As we need to find a way to represent the texts as vectors, the CountVectorizer () module is used. The CountVectorizer is a tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector based on the frequency of each word that occurs in the entire text. Later, we used the cosine\_similarity () module of scikit-learn to find the cosine similarity between the vectors to find out how similar they are to each other. Cosine Similarity is:

* A measure of similarity between two non-zero vectors of an inner product space.
* The cosine of the trigonometric angle between two vectors.
* Applied to vectors of low and high dimensionality.
* Not a measure of vector magnitude, just the angle between vectors.

Then as a final step, we created a function that takes in a movie title as an input and outputs a list of the 8 most similar movies. This function will match the input movie title with the corresponding index of the similarity matrix and extract the row of similarity values.

**4. Model Evaluation and Validation**: After training the model, the model is used to make some predictions, and hence, its performance is evaluated and validated. Here, the recommendation function we defined is used to get the list of recommended movies that have some similarities with the movie we choose.

We tried to get recommendations for the movie Avatar and look at the list of 8 recommended movies we got as output:

**Results:** We create a recommender using content-based filtering. The result of this analysis shows that the recommender is successfully recommending similar movies.

5. **Future Scope and Conclusion:**

- Future Scope:

- Integration of collaborative filtering for enhanced recommendations.

- Implementation of deep learning models for advanced content analysis.

- Development of an interactive user interface for a seamless user experience.

- **Conclusion**:

- [Summarize the achievements and outcomes of the project. Highlight the significance of the content-based recommender system in providing personalized movie recommendations.]