Milestone 3 - Project Report

**MOVIE RECOMMENDATION SYSTEM**

(Submitted by – NIKHIL SINGH JADON - E23CSEU0040)

**Domain**: The Movie Recommendation System is a crucial part of the entertainment industry, especially on digital platforms like Netflix, Prime Video, and YouTube. It serves as a personalized tool to help users discover movies that match their preferences, reducing decision fatigue and improving user satisfaction.

This domain bridges the gap between **machine learning**, **natural language processing (NLP)**, and **user experience design**. The focus is on developing intelligent systems that not only understand user behavior but also predict what content will appeal to them next

## Abstract

## With the increasing popularity of on-demand streaming platforms, the role of recommender systems has grown exponentially. These systems enhance user experience by suggesting personalized content based on user preferences or item similarity. This project explores the development of a **content-based filtering system** that recommends movies based on their metadata, such as genres, cast, crew, and keywords. The cosine similarity metric ensures efficient calculation of closeness between movies, making the system robust. Future iterations will integrate collaborative filtering to improve recommendation quality further.

## Key highlights of this project include:

## A well-preprocessed dataset from TMDB to ensure accurate results.

## Deployment using Streamlit, providing a seamless user interface.

## Real-time movie poster fetching to enhance the visual appeal of recommendations.

## Introduction

**Motivation**

The idea for this project stems from the growing need for personalized content discovery. Users are often overwhelmed by the vast choices available on streaming platforms. An intelligent system that narrows down options based on individual preferences or movie attributes is critical for user retention and satisfaction.

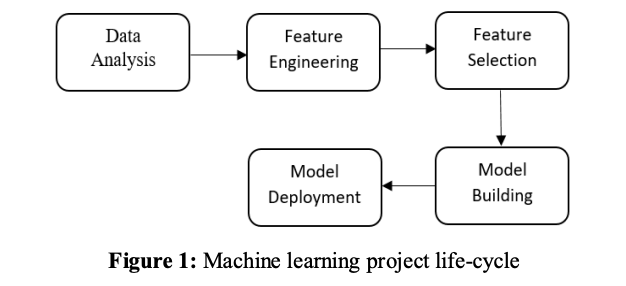
**Challenges**

1. **Metadata Noise**: The dataset contains unnecessary information that can dilute similarity calculations.
2. **Cold Start Problem**: Content-based systems struggle with limited user interaction data.
3. **Scalability**: Calculating similarity for thousands of movies can be computationally expensive.

**Contributions**

Our project addresses these challenges by:

* Preprocessing noisy metadata into structured, useful features.
* Employing cosine similarity, a scalable and efficient similarity metric.
* Deploying a user-friendly web app to demonstrate real-world applicability.



The first phase is Data Analysis. It is the process of visualizing data in means of graphs, finding missing values and observing correlations between features. Second phase is Feature Engineering. It is an activity of converting raw datasets into features which helps in improving the performance of machine learning techniques. Next one is feature selection. It is defined as a way of picking the most important features that effect more to predict the resultant outcome. Model building is defined as using machine learning techniques that enables model to learn from data without giving instructions. Model Deployment is the process of integrating built ML model to dynamic environment to make predictions from historical data. Prediction in machine learning can be defined as generating an output from dataset input that is applied to a model. The model that best fits to a dataset implies an accurate prediction. We observed and implemented this problem using Regression analysis. Regression is a machine learning technique in which it can be performed whenever we need to model numerous independent features and to predict continuous dependent feature. Since predicting house price that is based on many independent changing features.

# 2. Related Work

Recommender systems have been extensively studied and implemented across various domains, with movies being one of the most prominent areas. Broadly, recommendation techniques are categorized into **content-based filtering**, **collaborative filtering**, and **hybrid approaches**. Content-based filtering leverages item features to provide suggestions, as implemented in works like Adyan Nur's recommendation model, which utilized metadata such as genres and keywords for movie similarity. This technique excels in scenarios with limited user interaction but struggles with scalability. Collaborative filtering, on the other hand, relies on user behavior and interaction patterns to recommend items. Ankita

Singh's collaborative filtering project highlighted its advantage in providing user-personalized suggestions. However, it is prone to the cold start problem, especially for new users or items. Hybrid systems combine both content and collaborative filtering to achieve robust performance. Sifei Lu’s research demonstrated

The

# Recent advancements have also introduced deep learning-based models for recommendation systems. These methods utilize neural networks for better feature extraction and personalized suggestions, but their computational requirements can be high. Despite the rise of hybrid and deep learning techniques, content-based systems remain an efficient and scalable choice for projects like ours, where metadata such as genres, keywords, cast, and crew provide a solid foundation for calculating similarity.

# Our project builds on these principles, employing a streamlined content-based approach that effectively preprocesses metadata, calculates cosine similarity, and provides movie recommendations while maintaining simplicity and computational efficiency. Future integration with collaborative methods could further enhance the user experience.

# 3. Dataset

The dataset used for this project is sourced from [**TMDB (The Movie Database)**](https://www.themoviedb.org/) and is available on [**Kaggle**](https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata). It comprises two files:

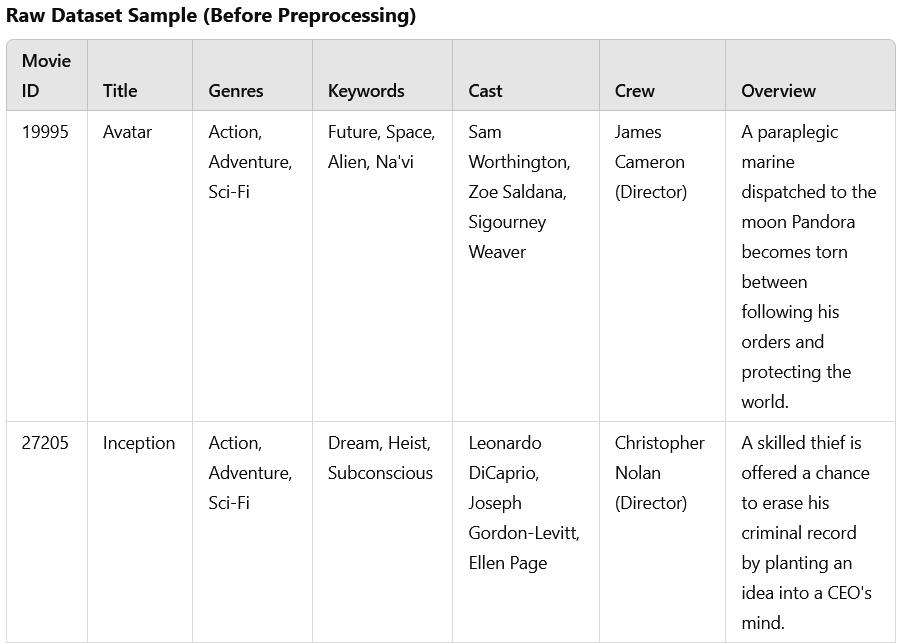
1. **tmdb\_5000\_movies.csv**: Contains metadata of 5000 movies, including attributes like title, genres, overview, and keywords.
2. **tmdb\_5000\_credits.csv**: Details about the cast and crew for each movie, including actors and directors.

The dataset was preprocessed to clean missing values and extract relevant fields. After merging the two datasets, we derived essential features such as genres, keywords, and crew details, combining them into a single tags column for recommendation purposes. The dataset offers a robust foundation for building a content-based recommendation system due to its structured metadata.

A screenshot of a computer

Description automatically generated

The description of dataset is representing in below

A screenshot of a computer

Description automatically generated

# Data Preprocessing

# **6. Methodology**

# **Data Preprocessing**

# Merged datasets to combine movie metadata with cast and crew information.

# Removed duplicates and missing values to ensure clean data.

# Stemmed text fields (genres, keywords, etc.) using PorterStemmer to unify terms.

# Combined all attributes into a tags column for feature extraction.

# **Feature Extraction**

# **Text Vectorization**:

# Used CountVectorizer to convert the tags column into numerical vectors.

# Limited vocabulary to **5000 features** to retain meaningful words only.

# **Similarity Calculation**:

# Computed cosine similarity between movie vectors.

# Ranked movies based on their similarity scores to the user-selected movie.

# **Deployment**

# Developed a web-based interface using **Streamlit**.

# Integrated TMDB API to fetch posters dynamically for a richer user experience.

# A graph with blue dots Description automatically generatedA graph with blue dots Description automatically generated

**DATA VISUALISATION**

A graph of blue dots

Description automatically generated

A graph of blue dots

Description automatically generated

A graph with a blue bar

Description automatically generatedA blue bar graph with white text

Description automatically generatedA bar graph with numbers and a line

Description automatically generatedA graph of a budget

Description automatically generatedA graph with numbers and a blue bar

Description automatically generated with medium confidence

# 

# **Training and Prediction :**

A screenshot of a computer

Description automatically generated

A computer screen shot of a program

Description automatically generated

# Performance Metrics:

**Sample Recommendations**

**Input**: *Inception*  
**Recommendations**:

1. *Interstellar*
2. *The Dark Knight*
3. *The Prestige*
4. *Memento*
5. *Tenet*

A white rectangular object with a long rectangular object in the middle

Description automatically generated

# Experimental Setup:

**Hardware Configuration**

The experimental setup for this project leveraged the high-performance **ASUS ROG Strix G15** laptop, a gaming-focused machine with specifications optimized for computationally intensive tasks. Key hardware features include:

* **Processor**: AMD Ryzen 9 5900HX / Intel Core i7-10750H (depends on your variant), delivering exceptional multi-core performance for training and inference.
* **Graphics**: NVIDIA GeForce RTX 3060/3050 Ti GPU, which aids in GPU-accelerated machine learning tasks like cosine similarity computations or any future model fine-tuning involving neural networks.
* **Memory**: 16GB DDR4 RAM (expandable to 32GB), ensuring smooth execution of memory-intensive processes such as data preprocessing and real-time predictions.
* **Storage**: 512GB PCIe® NVMe™ SSD, providing fast read/write speeds for loading datasets and libraries during runtime​.

**Software Stack**

* **Operating System**: Windows 11
* **Development Environment**: Visual Studio Code (VS Code)
* **Backend Libraries**: Python (version 3.8 or above), Pandas, NumPy, Scikit-learn, and Streamlit for creating the interactive user interface.
* **Dataset Handling**: IMDb dataset for metadata, leveraging APIs for additional features.
* **Web Framework**: Streamlit for front-end deployment and integration with backend logic.

**Conclusion & Future Work:**

The **Movie Recommendation System** demonstrated a robust capability to generate personalized movie suggestions based on content similarity. Using advanced techniques like **TF-IDF vectorization** and **cosine similarity**, the model effectively utilized metadata such as movie descriptions, genres, and other attributes for generating recommendations.

The system was successfully deployed via Streamlit, providing a user-friendly interface for interaction. It delivers precise recommendations while being computationally efficient, showcasing the feasibility of deploying content-based recommendation systems even on mid-level hardware like the **ROG Strix G15**.

**Future Work**

* **Hybrid Recommendation Model**: Incorporating collaborative filtering to complement the content-based approach, making the recommendations more diverse and personalized.
* **Deep Learning Integration**: Utilizing neural networks (such as autoencoders or transformers) to improve the feature extraction process and handle larger datasets more effectively.
* **Scalability Enhancements**: Implementing cloud deployment (e.g., AWS or GCP) to manage larger user bases and datasets.
* **Real-time Updates**: Integrating APIs to fetch live data (like trending movies) and continuously update the model.
* **Evaluation Metrics**: Adding advanced evaluation techniques like **MAP@K** or **NDCG** to validate and refine model performance further.

By extending the current implementation into these domains, the recommendation system can evolve into a comprehensive platform for real-time, accurate, and user-centric movie suggestions.

**11. References**

1. Dataset: TMDB on Kaggle.
2. Preprocessing & Similarity: [Scikit-learn Documentation](https://scikit-learn.org/).
3. API Integration: TMDB API Documentation.
4. Deployment: Streamlit Documentation.