A Mini Project Report

on

BEST STREAMING SHOWS SEGMENTATION ANALYSIS

Submitted to CMREC HYDERABAD

In Partial Fulfillment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY IN
COMPUTER SCIENCE AND ENGINEERING

Submitted By

BISSA POOJITHA	**218R1A05K4**
DHOMMATA PRASHANTH	**218R1A05L2**
MEDEPALLI ISHWARYA	**218R1A05N0**
SAI KIRAN SAVANT	**218R1A05P0**

Under the Esteemed guidance of

Mr. Mohammed Azhar

Assistant Professor, Department of CSE

Department of Computer Science & Engineering

CMR ENGINEERING COLLEGE

(UGC AUTUNOMOUS)

(Approved by AICTE, NEW DELHI, Affiliated to JNTU, Hyderabad)
Kandlakoya, Medchal Road, R.R. Dist. Hyderabad-501 401)

2024-2025

CMR ENGINEERING COLLEGE

(Accredited by NBA,Approved by AICTE NEW DELHI, Affiliated to JNTU, Hyderabad)

Kandlakoya, Medchal Road, Hyderabad-501 401

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project entitled **"Best Streaming Shows Segmentation Analysis"** is a bonafide work carried out by

BISSA POOJITHA	**(218R1A05K4)**
DHOMMATA PRASHANTH	**(218R1A05L2)**
MEDEPALLI ISHWARYA	**(218R1A05N0)**
SAI KIRAN SAVANT	**(218R1A05P0)**

in partial fulfillment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING** from CMR Engineering College, affiliated to JNTU, Hyderabad, under our guidance and supervision.

The results presented in this project have been verified and are found to be satisfactory. The results embodied in this project have not been submitted to any other university for the award of any other degree or diploma.

Internal Guide	**Mini Project Coordinator**	**Head of the Department**	**Examiner**
Mr MD.Azhar Assistant Professor CSE, CMREC	**Mr. S. Kiran Kumar** Assistant Professor CSE, CMREC	**Dr. Sheo Kumar** Professor & H.O.D CSE, CMREC	

DECLARATION

This is to certify that the work reported in the present project entitled "**Best Streaming Shows Segmentation Analysis”** is a record of bonafide work done by us in the Department of Computer Science and Engineering, CMR Engineering College, JNTU Hyderabad. The reports are based on the project work done entirely by us and not copied from any other source. We submit our project for further development by any interested students who share similar interests to improve the project in the future.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

BISSA POOJITHA	**(218R1A05K4)**
DHOMMATA PRASHANTH	**(218R1A05L2)**
MEDEPALLI ISHWARYA	**(218R1A05N0)**
SAI KIRAN SAVANT	**(218R1A05P0)**

<u>ACKNOWLEDGMENT</u>

We are extremely grateful to **Dr. A. Srinivasula Reddy**, Principal and **Dr.Sheo Kumar**, HOD, **Department of CSE, CMR Engineering College** for their constant support**.**

I am extremely thankful to **Mr.MD.Azhar,** Assistant Professor, Internal Guide, Department of CSE, for his constant guidance, encouragement and moral support throughout the project.

I will be failing in duty if I do not acknowledge with grateful thanks to the authors of the referencesand other literatures referred in this Project.

I thank **Mr.S. Kiran Kumar** Mini Project Coordinator for his constant support in carrying out the project activities and reviews.

I express my thanks to all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time.

Finally, I am very much thankful to my parents who guided me for every step.

BISSA POOJITHA	**(218R1A05K4)**
DHOMMATA PRASHANTH	**(218R1A05L2)**
MEDEPALLI ISHWARYA	**(218R1A05N0)**
SAI KIRAN SAVANT	**(218R1A05P0)**

CONTENTS

ABSTRACT

In the rapidly evolving landscape of Over-the-Top (OTT) streaming platforms, the sheer abundance of TV show choices presents both a challenge and an opportunity. This abstract delves into the necessity of data analysis on OTT TV shows, highlighting the pivotal role it plays in unraveling viewer patterns and shaping the future of content delivery. OTT platforms host an extensive catalog of TV shows spanning genres, languages, and platforms. The diverse preferences of viewers, influenced by factors such as age, genre affinity, and platform accessibility, necessitate a nuanced understanding for platform providers, content creators, and advertisers alike. Data analysis emerges as a crucial tool to dissect and interpret the multifaceted landscape of viewer behavior. The rapid proliferation of Over-the-Top (OTT) platforms has significantly transformed the entertainment landscape, offering viewers a plethora of choices in TV show selections. In this project, we conducted an in-depth data analysis on an OTT TV shows dataset, leveraging the K-means clustering algorithm to unveil underlying patterns in viewer preferences. The dataset encompassed key features such as age suitability, IMDb ratings, Rotten Tomatoes scores, show titles, and the availability on popular streaming platforms including Netflix, Hulu, Prime Video, Disney Plus, and Hotstar.

LIST OF FIGURES

2. INTRODUCTION

1.1. Introduction to Project:

The landscape of home entertainment has undergone a seismic shift with the advent of Over-the-Top (OTT) streaming platforms, providing viewers with an unprecedented array of choices. In this era of digital content consumption, understanding the diverse preferences of viewers has become paramount for contentcreators, platforms, and advertisers. This project embarks on a comprehensive exploration of viewer behavior within the OTT domain, employing data analysis techniques, specifically the K-means clustering algorithm, to unravel intricate patterns within a rich dataset.The dataset under examination encapsulates a multifaceted view of OTT TV shows, featuring essential attributes such as age suitability, IMDb ratings, Rotten Tomatoes scores, show titles, and the availability on prominent streaming platforms including Netflix, Hulu, Prime Video, Disney Plus, and Hotstar. Each of these features contributes a unique dimension to the viewer experience, shaping the dynamics of content consumption and platform engagement. The overarching goal of this project is to decipher hidden structures within the dataset using the K-means clustering algorithm, an unsupervised machine learning technique. By categorizing TV shows into distinct clusters based on shared characteristics, the analysis aims to reveal nuanced insights into viewer preferences. The algorithm will group shows with similar attributes, allowing for the identification of discernible patterns related to age-specific content, correlations between IMDb ratings and platform availability, and trends in Rotten Tomatoes scores. As the project unfolds, it seeks to answer pivotal questions about viewer segmentation, content quality perception, and the influence of streaming platforms on content accessibility. The K-means clustering algorithm, known for its efficiencyin identifying patterns in unlabeled datasets, stands as a powerful tool in unlocking the intricacies of viewer behavior within the ever-expanding OTT universe. Through this project, we anticipate shedding light on the intricate interplay of factors that influence viewer choices, ultimately contributing to the optimization of content curation, platform strategies, and the overall enhancement of the viewer experience in the dynamic realm of OTT TV shows.

1.1. Purpose of the Project

The scope of the data analysis project on OTT TV shows, utilizing the K-means clustering algorithm, is to gain profound insights into viewer behavior and preferences within the ever-expanding landscape of digital content consumption.By categorizing TV shows into distinct clusters based on shared characteristics, the analysis aims to reveal nuanced insights into viewer preferences.

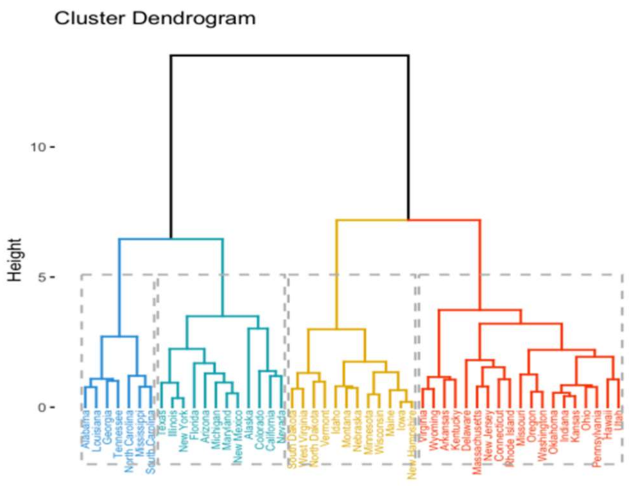
1.2. Proposed systems with FeaturesProposed system

To begin, data preprocessing was performed to clean and format the dataset, handling missing values and standardizing numerical features. The K-means clustering algorithm was then applied to group TV shows based on similarities in the specified features, allowing us to categorize shows into distinct clusters reflective of viewer preferences.The selection of the optimal number of clusters was a crucial step in our analysis.The K-means algorithm was executed, assigning each TV show to a specific cluster based on its feature profile and the Elbow Curve method.The results of the clustering analysis provided valuable insights into the diverse preferences of viewers.Clusters showcased trends such as age-specific preferences, correlations between IMDb ratings and platform availability, and distinctive patterns in Rotten Tomatoes scores across different clusters. This information is not only beneficial for content recommendation systems but also for OTT platforms seeking to tailor their content strategy to thepreferences of specific viewer segments. To categorize TV shows into distinct segments based on shared characteristics, enabling a nuanced understanding of viewer preferences.To inform content creators and platforms about the types of content that resonate with specific viewer segments.To contribute to a more personalized and satisfying viewer experience.

1.3 EXISTING SYSTEM

AGGLOMERATIVE HIERARCHICAL CLUSTERING ANALYSIS

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabelled datasets into a cluster and also known as hierarchical cluster analysis or HCA. Sometimes the results of K-means clustering and hierarchical clustering may look similar, but they both differ depending on how they work. As there is no requirement to predetermine the number of clusters as we did in the K-Means algorithm.



DRAWBACKS OF EXISTING SYSTEM

1. Hierarchical clustering is computationally more expensive than K-Means, especially for large datasets. The time complexity is O (n^2 log n).
2. Hierarchical clustering may not scale well for very large datasets, making it less suitable for applications with extensive data.
3. Hierarchical clustering can be sensitive to noise and outliers, impacting the structure of the dendrogram.
4. Unlike K-Means, hierarchical clustering produces a fixed number of clusters once the dendrogram is cut, which might not be suitable for all datasets.

3. LITERATURE SURVEY

2.1. Movie Popularity and Target Audience Prediction Using the Content-Based Recommender System(2022): The movie is one of the integral components of our everyday entertainment. The worldwide movie industry is one of the most growing and significant industries and seizing the attention of people of all ages. It has been observed in the recent study that only a few of the movies achieve success.Uncertainty in the sector has created immense pressure on the film production stakeholder. Moviemakers and researchers continuously feel it necessary to have some expert systems predicting the movie success probability preceding its production with reasonable accuracy. A maximum of the research work has been conducted to predict the movie popularity in the post-production stage. To help the movie maker estimate the upcoming film and make necessary changes, we need to conduct the prediction at the early stage of movie production and provide specific observations about the upcoming movie. This study has proposed a content-based (CB) movierecommendation system (RS) using preliminary movie features like genre, cast, director, keywords, and movie description. Using RS output and movie rating and voting information of similar movies, we created a new feature set and proposed a CNN deep learning (DL) model to build a multiclass movie popularity prediction system. We also proposed a system to predict the popularity of the upcoming movie among different audience groups. We have divided the audience group into four age groups junior, teenage, mid-age and senior. This study has used publicly available Internet Movie Database (IMDb) data and The Movie Database (TMDb) data. We had implemented a multiclass classification model and achieved 96.8% accuracy, which outperforms all the benchmark models. This study highlights the potential of predictive and prescriptive data analytics in information systems to support industry decisions.

2.2 An Intelligent Movies Recommendation System Based Facial Attributes Using Machine Learning(2023): Movie theaters and platforms offer a wide selection of movies that require filtering to match the preferences of individual users. Recommender systems are an effective tool for this task. This study introduces a hybridrecommender system that combines collaborative filtering and content-based approaches to provide personalized movie recommendations. The proposed system considers age, gender, emotion, and genre attributes to ensure the suitability of the recommended movies. The proposed system targets the visitors of cinema theaters to try a new experience of choosing the next movie to watch by recognizing the visitor’s face to determine his/her age, gender and emotion using camera. Therefore, our system is expected to achieve better results than traditional approaches. The system's performance is evaluated using standard metrics such as precision, recall, and F1-measure. The findings indicated that the proposed system outperforms the benchmark system in the most tested scenarios. However, the precision of the benchmark work is slightly higher in (8-14 and 15-24) age groups.

2.2 An Efficient Machine Learning System using Sentiment Analysis for Movie Recommendations(2022): In this fast-paced modern world, everyone needs some kind of entertainment in order to maintain a positive attitude and maintain his or her energy levels. The ability to rebuild our self-confidence and feel happy about our profession is made possible by the use of entertainment. When a person needs to refresh himself, he may watch either movies, web series, or anime of his choice, or can listen to his favorite music. The movie suggestion systems can be applied in order to watch the chosen movies online, and these systems are dependable since it reduces the time required looking for desired movies, which cannot be afforded to squander. This paper implements content-based filtering using cosine in order to enhance the performance of a movie recommendation system. Comparative results have been demonstrated, and similarities in the suggested methodology are presented. Both of these findings reveal that the proposed strategy outperforms pure approaches in terms of accuracy, quality, and scalability of the movie recommendation system across three different datasets. The "preferred" that a user would give to a system is predicted by a recommendation system, which is a subclass of information filtering systems. These systems base their predictions on information from users. Their most common applications are in the corporate world. The Movie Recommendation System helps users quickly and effectively find movies that interest them based on either their own prior movie experiences or the movie experiences of other users, all without having to waste time looking for or scrolling through films that are irrelevant to them.

2.4 Enhancing Performance of Movie Recommendations Using LSTM With Meta Path Analysis(2023): Movie recommendation algorithms play an important role in assisting consumers in identifying films that match their likes. Deep Learning, particularly Long Short-

Term Memory (LSTM) networks, has shown substantial promise in collecting sequential patterns to improve movie recommendations among the different techniques used for this purpose. Long Short-Term Memory-Inter Intra- meta path Aggregation (LSTM-IIMA) in movie recommendation systems is proposed in this study, with a specific focus on incorporating intra and inter-meta path analysis. The intra-meta path analysis investigates interactions within a single meta path, whereas the inter-meta path analysis investigates links between numerous meta paths. Intra and inter-meta path analyses are used in the LSTM-based movie recommendation system LSTM-IIMA to capitalize on these rich linkages. Each meta path sequence records the dependencies of a user's interactions with films and other things. The LSTM architecture has been modified to handle these meta path sequences, processing them to record temporal dependencies and entity interactions. To optimize the parameters and minimize prediction errors, the model is trained using supervised learning techniques. To measure the quality and usefulness of the recommendations,the LSTM-IIMA evaluation incorporates metrics such as precision, recall, ablation analysis, time efficiency and Area Under the Curve (AUC). The performance of the system is compared to that of alternative recommendation techniques HAN and MAGNN. Overall, incorporating intra and inter meta path analysis into the LSTM- IIMA improves its ability to capture complex linkages and dependencies between movies, users, and other things.

4. SOFTWARE REQUIREMENT ANALYSIS

a. SDLC

The **Systems Development Life Cycle (SDLC)** or Software Development Life Cycle in systems engineering, information systems and software engineering, is the process ofcreating or altering systems, and the models and methodologies use to develop these systems.

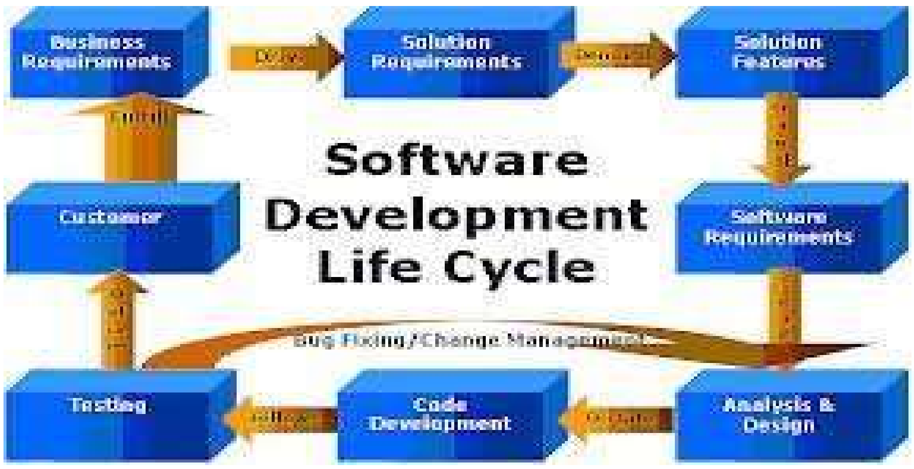


Figure 3.1(a): Software Development Life Cycle

Requirement Analysis and Design:

In the requirement analysis phase, the process starts by understanding the core objectives and stakeholders' expectations. The goal is to identify key requirements for gathering, processing, and analyzing data related to TV shows or movies to determine patterns and insights. First, stakeholders such as content analysts, data scientists, and business managers are consulted to gather detailed requirements. These discussions aimto understand what kind of data is important (e.g., ratings, genres, viewership statistics, and demographic information) and how this data will be sourced, whether from databases, streaming platforms, or third-party APIs.

The collected requirements are documented in a structured format. This documentation includes functional requirements, such as the need to aggregate data on shows, preprocess the data for clustering, and visualize the clustered results. Non-functional requirements might include performance metrics like processing speed, scalability to handle large datasets, and security measures to protect sensitive information.

In the design phase, the focus shifts to creating a detailed blueprint for implementingthe system. This starts with defining the high-level architecture, outlining the major components such as data collection modules, preprocessing pipelines, clustering algorithms, and visualization tools. The design phase culminates in a set of detailed design documents that serve as a blueprint for the development team. These documents provide clear guidelines on how to implement each component, ensuring that thesystem meets the specified requirements and functions as intended.

Implementation:

In the implementation phase, the detailed designs and plans created during the design phase are translated into actual code. This phase begins with setting up the developmentenvironment, ensuring that all necessary tools, frameworks, and libraries are installed and configured correctly. Developers start by writing the code for the different modulesand components as specified in the design documents. This often involves creating the data collection module first, which handles the ingestion of data from various sources. The data collection code needs to be robust, capable of handling different data formats, and able to deal with potential issues such as missing or corrupted data. The implementation of the clustering algorithm is the next major step. The k-means clustering algorithm is coded to accept the preprocessed data and group it into clusters based on the specified number of clusters (k). This involves initializing cluster centroids, assigning data points to the nearest centroids, and iteratively updating the centroids until convergence is achieved. The implementation ensures that the algorithm runs efficiently, even on large datasets.

Testing:

In the testing phase of the project, the primary goal is to ensure that the system functions correctly and meets the specified requirements before it is fully deployed. This phase involves a series of activities designed to identify and fix any defects or issues in the system. The first step in the testing phase is unit testing, where individual components of the system are tested in isolation. Each module, such as the data collection, preprocessing, clustering, and UI components, is evaluated to ensure itperforms its specific function correctly. Once unit testing is complete, integrationtesting is performed to ensure that the different modules work together seamlessly. Thisinvolves testing the interactions between components, such as how the data collected by the data collection module flows through the preprocessing pipeline and into the clustering algorithm.

Maintenance:

In the maintenance phase of the project, the focus shifts to ensuring that the system continues to operate effectively after it has been deployed. This phase involves ongoingactivities to address issues, enhance functionality, and ensure the system remains reliable and useful over time.

- Bug Fixes and Issue Resolution: Address and resolve any bugs or issues that ariseafter deployment based on user reports or system monitoring.
- Performance Monitoring: Continuously monitor system performance to identify andaddress any inefficiencies or bottlenecks.
- Enhancements and Upgrades: Implement new features and improvements based onuser feedback and evolving needs.
- Security Updates: Apply regular security patches and updates to protect against newthreats and vulnerabilities.

3.2 SDLC METHDOLOGIES:

This document play a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.Several Software Development Life Cycle (SDLC) methodologies can be employed, each with its own approach to planning,development, and delivery.

Agile Methodology

Agile emphasizes iterative development, flexibility, and collaboration. It involves breaking the project into small, manageable increments called sprints or iterations.

Application to the Project:

- Iterative Development: Develop and test small portions of the system in short cycles. For example, one sprint might focus on data collection and preprocessing, while another focuses on implementing the clustering algorithm.
- Continuous Feedback: Regularly gather feedback from stakeholders to adjust requirements and improve the system incrementally.
- Flexibility: Adapt to changes in requirements or technology as the project progresses.

Agile is ideal for projects where requirements are expected to evolve or where early delivery of partial functionality is beneficial**.**

The following diagram shows how an agile model acts like:

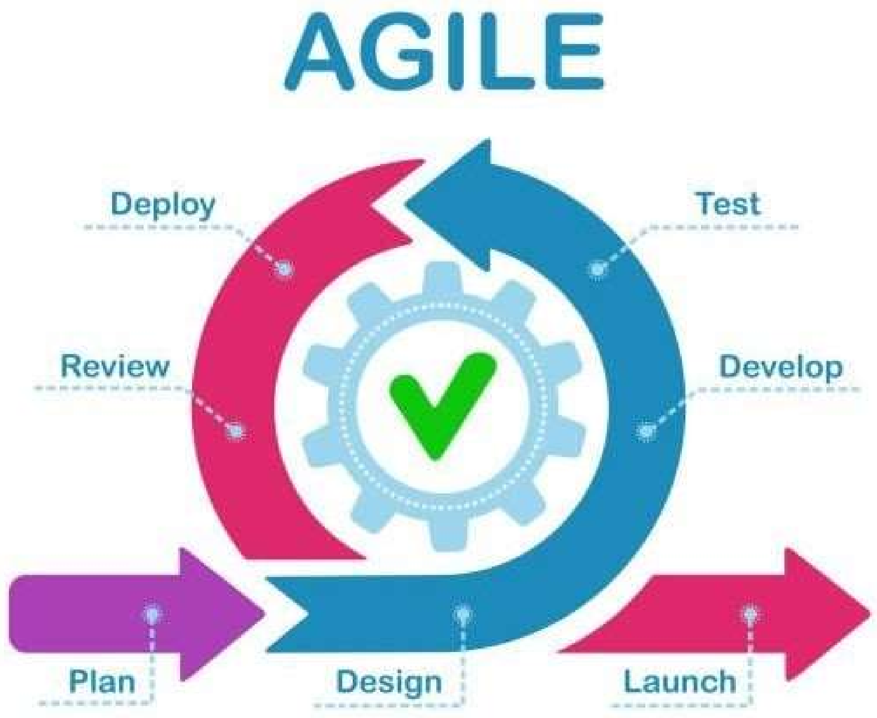


Figure 3.1(b): Agile Model

Spiral Model

The Spiral model combines iterative development with a focus on risk assessment. It involves repeating cycles of planning, risk analysis, engineering, and evaluation.

Application to the Project:

- Iterative Cycles: Each cycle involves revisiting requirements, assessing risks, and refining the design and implementation. For example, one cycle might focus on developing and testing the clustering algorithm, while another focuses on integrating the UI.
- Risk Management: Identify and address risks early in each cycle to minimizepotential issues.

The Spiral model is beneficial for complex projects with high uncertainty and evolving requirements.

The following diagram shows how a spiral model acts like:

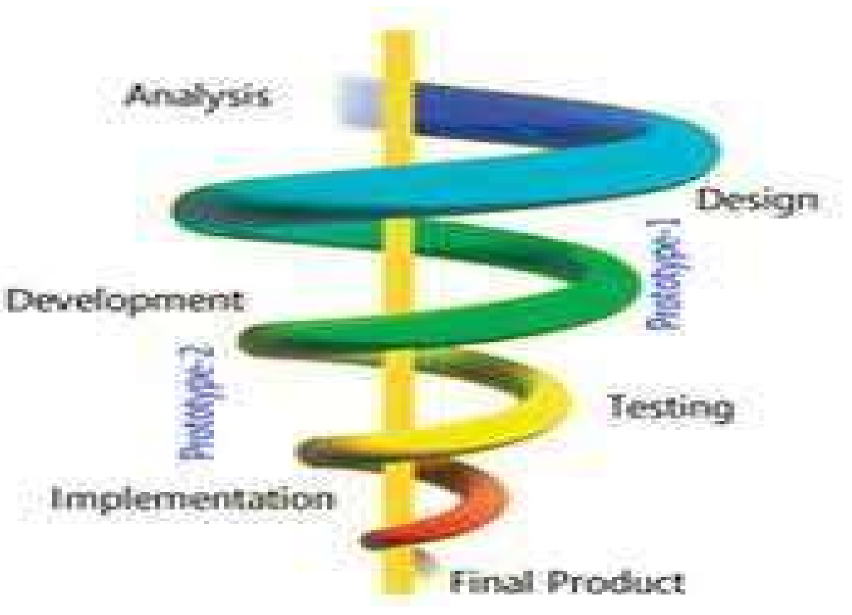


Figure 3.1(c): Spiral Model

3.3. Modules and their Functionalities:

Each module has a specific functionality that contributes to the overall effectiveness ofthe system.

3.3.1. Data Collection Module:

The data collection module ensures that the system has access to comprehensive and up-to-date information about shows and movies. It may involve integrating APIs fromdifferent data sources or scraping data from web pages. The module must handle data extraction, transformation, and loading processes to prepare the data for subsequent analysis.

Functionality:

- Data Sources: This module is responsible for gathering data from various sources such as streaming platforms (Netflix, Hulu), movie databases (IMDb, TMDb), anduser reviews.
- Data Ingestion: It collects data including show titles, genres, ratings, reviews, anduser demographics.

- Data Storage: The collected data is stored in a database or data warehouse forfurther processing.

3.3.2. Data Preprocessing Module:

Preprocessing transforms raw data into a format suitable for analysis. For example, textual data such as show descriptions is processed using techniques like tokenization and stemming, and numerical data such as ratings may be scaled or normalized. Featureextraction may involve creating feature vectors that represent the content of each show based on various attributes.

Functionality:

- Data Cleaning: This module cleans the raw data by removing duplicates, handlingmissing values, and correcting inconsistencies.
- Feature Extraction: It extracts relevant features from the data, such as genres,keywords from descriptions, and user ratings.
- Normalization: The module normalizes the data to ensure consistency in format andscale, which is crucial for accurate clustering.

3.3.3 Feature Engineering Module:

Feature engineering involves creating and refining the input features used for clustering. For example, text data might be converted into numerical vectors using methods suchas TF-IDF or word embeddings. The goal is to ensure that the features used for clustering are meaningful and representative of the data.

- **Functionality:**Feature Selection: This module selects the most relevant features for clustering. This may include genre, average rating, user preferences, and keywords.
- Dimensionality Reduction: Techniques like Principal Component Analysis(PCA) might be used to reduce the number of features while preserving important information.

3.3.4. Clustering Module:

The clustering module is central to the recommendation system. It involves initializingcluster centroids, assigning shows to the nearest centroid, and iteratively updating centroids until convergence. The module also includes functionality to determine the optimal number of clusters (k) and assess the clustering results to ensure they are meaningful.

Functionality:

- K-means Clustering: This module applies the k-means algorithm to group showsinto clusters based on their similarity. The algorithm partitions the data into k clusters, where each cluster contains shows with similar attributes.
- Cluster Evaluation: It evaluates the quality of the clusters using metrics likesilhouette score or within-cluster sum of squares.

3.3.5. Recommendation Engine Module:

The recommendation engine uses the clusters to provide relevant suggestions. For example, if a user prefers action shows, the engine might recommend top-rated shows from the cluster that primarily consists of action genres. Personalization techniques canfurther refine recommendations based on user ratings, viewing history, and demographic information.

Functionality:

Recommendation Logic: Based on the clustering results, this module recommendsshows to users. It identifies which cluster a user's preferences align with and suggests top shows from that cluster.

- Personalization: The module may incorporate user-specific preferences and pastinteractions to tailor recommendations.

3.4 Functional Requirements

3.4.1 Data Collection and Integration:

- Objective: Gather comprehensive data about movies and shows from varioussources, including genres, ratings, and reviews.
- Requirements: The system should integrate with APIs or data sources to fetch and update movie data regularly. It should handle different data formats and ensure dataaccuracy and consistency.

3.4.2 Data Preprocessing:

- Objective: Prepare the collected data for clustering by cleaning, normalizing, andtransforming it into a suitable format.
- Requirements: The system must handle missing values, remove duplicates, andnormalize data to ensure consistency. Textual data needs to be converted into numerical features using techniques like TF-IDF or embeddings.

3.4.3 Clustering:

- Objective: Group movies into clusters based on their features using the k-meansclustering algorithm.
- Requirements: The system should implement k-means clustering to partition the data into clusters. It must support determining the optimal number of clusters andevaluating clustering quality.

3.4.4 Recommendation Engine:

Objective: Provide users with personalized recommendations based on theclustering results.

- Requirements: The recommendation engine must use clustering results to suggestshows from clusters similar to the user's preferences. It should incorporate user- specific data to refine recommendations.

3.5 Non-Functional RequirementsPerformance:

- The system should provide recommendations with minimal latency, ensuring asmooth user experience.

Scalability:

- The system must handle increasing data volumes and user interactions withoutperformance degradation.

Security:

- The system should implement data protection measures to ensure user data privacyand comply with relevant regulations.

Usability:

- The user interface should be intuitive and accessible, catering to a diverse user baseand providing a seamless experience.

Reliability:

- The system should ensure high availability and robust error handling to maintainconsistent performance.

Maintainability:

The system should be easy to maintain and update, with well-documented code anda modular design

3.6 Feasibility Study

3.6.1 Feasibility Report:

This feasibility report evaluates the practicality of developing a best streaming shows segmentation analysis utilizing k-means clustering. The report covers technical, operational, economic, and schedule feasibility to determine whether the project is viable and how it aligns with business goals. Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Economical Feasibility
- Operation Feasibility

3.6.2 Technical Feasibility:

a. System Requirements:

- Data Collection: Access to comprehensive data sources like streaming platformsand movie databases.
- Technology Stack: Programming languages (e.g., Python, Java), libraries (e.g.,scikit-learn, pandas), and tools for data processing and clustering.
- Infrastructure: Adequate hardware and software infrastructure for data storage,processing, and analysis.

b. Technical Challenges:

- Data Integration: Combining data from multiple sources with varying formats andstructures.
- Feature Engineering: Effectively extracting and selecting features for clustering.
- Scalability: Handling large volumes of data and maintaining system performance.

c. Proposed Solutions:

- Use robust APIs and data extraction tools to integrate data.
- Apply advanced feature extraction techniques and dimensionality reduction.
- Implement scalable cloud-based solutions or high-performance servers to managedata processing needs.

d. Tools and Technologies:

- Data Collection: APIs (e.g., IMDb API), web scraping tools.
- Data Preprocessing: Python libraries (pandas, NumPy).
- Clustering: Scikit-learn for k-means clustering.

3.6.3 Operational Feasibility:

a. System Usability:

- User Interface: Intuitive design for ease of navigation and interaction.
- User Feedback: Mechanisms to collect and incorporate user feedback forcontinuous improvement.

b. Training and Support:

- Documentation: Comprehensive user manuals and technical documentation.
- Training: Providing training sessions for users and administrators.

c. Operational Challenges:

- Data Privacy: Ensuring compliance with data protection regulations (e.g., GDPR).
- Maintenance: Regular updates and bug fixes to ensure system reliability.

d. Proposed Solutions:

- Implement strong data protection measures and encryption.
- Establish a support team and regular maintenance schedules.

3.6.4 Economical Feasibility:

a. Cost Estimation:

- Development Costs: Costs related to hiring developers, data scientists, and UI/UXdesigners.
- Infrastructure Costs: Expenses for servers, databases, and cloud services.
- Operational Costs: Ongoing maintenance, updates, and user support.

b. Budget:

- Develop a detailed budget plan that includes initial development costs and recurringoperational expenses.
- Explore funding options or cost-sharing strategies if needed.

c. Return on Investment (ROI):

- Revenue Potential: Assess potential revenue streams, such as premium features or partnerships with streaming services.
- Cost-Benefit Analysis: Compare the estimated benefits (improved user engagement,increased subscriptions) against the costs.

d. Proposed Solutions:

- Perform a detailed cost-benefit analysis to ensure a positive ROI.
- Explore cost-effective solutions such as open-source tools and cloud-based services.

5. SOFTWARE & HARDWARE REQUIREMENTS

4.1 Requirement Specification:

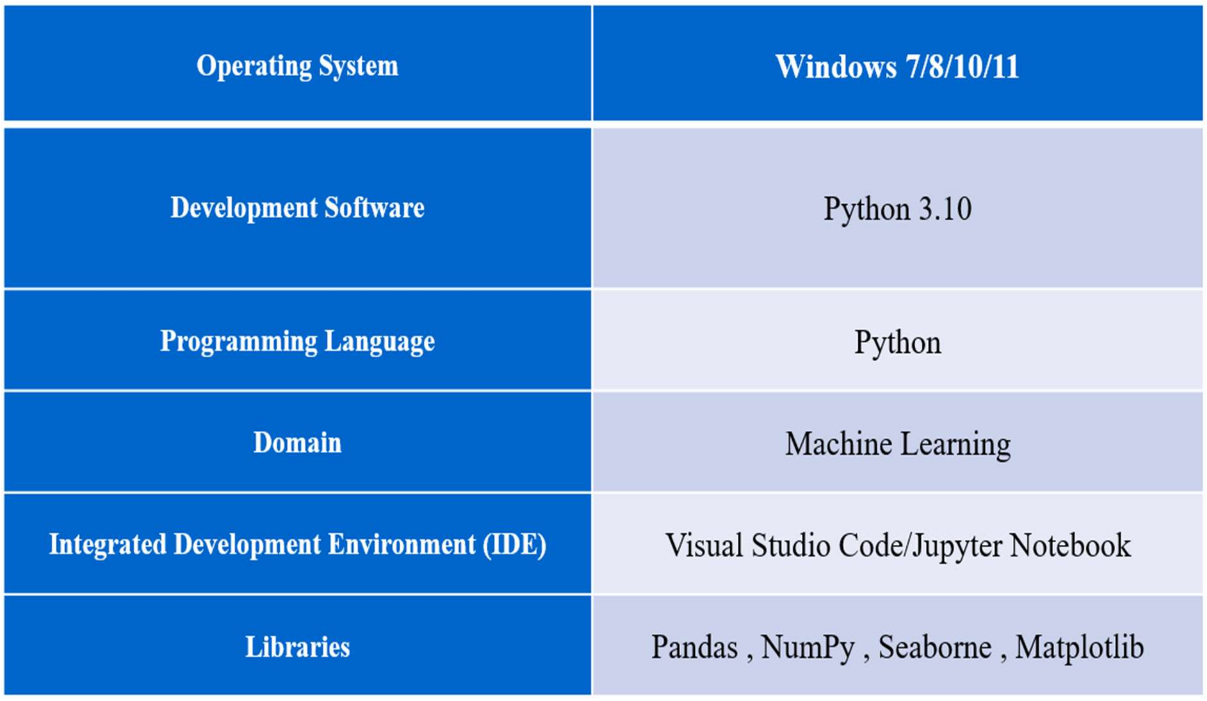
A requirement specification for a software system is a complete description of the behaviour of a system to be developed. It includes a set of use cases that describe allthe interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Non-functional requirements which impose constraints on the design or implementation such as performance engineering requirements, quality standards.

System requirement specification is a structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analysing the business needs of their clients and stakeholders to help identify the business problems and propose solutions.

4.2. Hardware Requirements:

	MY SYSTEM
System	A PC with Windows/Linux OS
Hard Di sk	512 GB
Ram	Minimum of 8gb RAM.
Process or	Processor with 2.40GHz 2.50 GHz speed

4.3. Software Requirements:



Operating System	Windows 7/8/10/11
Development Software	Python 3.10
Programming Language	Python
Domain	Machine Learning
Integrated Development Environment (IDE)	Visual Studio Code/Jupyter Notebook
Libraries	Pandas , NumPy , Seaborne , Matplotlib

4.4. Selected Software

4.4.1. Introduction to python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted − Python is processed at runtime by the interpreter.You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable andterse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you haveto scan,read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

4.4.2 Python libraries

For the computation and analysis we need certain python libraries which are used to perform analytics. Packages such as SKlearn, Numpy, pandas, Matplotlib, Flask framework, etc are needed.

NumPy

NumPy is a fundamental library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a wide array of mathematical functions to operate on these arrays. In the context of a movie recommendation system, NumPy is primarily used for handling numerical data and performing mathematical operations efficiently. It enables fast computation by leveraging its powerful array operations and broadcasting features. For instance, NumPy can be employed to manage and manipulate the feature vectors of movies, which are essential for clustering algorithms. It also helps in optimizing mathematical operations during the clustering process, such as calculating distances between data points.

Scikit-learn (sklearn)

Scikit-learn is a versatile library for machine learning in Python, providing a range of tools for data preprocessing, model building, and evaluation. In the movie recommendation system, Scikit-learn is used extensively for implementing the k-means clustering algorithm. This library simplifies the process of clustering by offering a robust and well-documented implementation of k-means, including methods for fitting the model, predicting clusters, and evaluating clustering performance. Additionally, Scikit-learn provides various utilities for preprocessing data, such as scaling numerical features and encoding categorical variables, which are crucial steps before applying clustering algorithms. Its built-in functions for model evaluation, like the silhouette score, help in assessing the quality and effectiveness of the clustering results.

Pandas

Pandas is a powerful library for data manipulation and analysis in Python,designed to work with structured data. It provides DataFrames, which are versatile data structures that allow for efficient manipulation and analysis of tabular data. In the movie recommendation system, Pandas is used to handle andpreprocess the data collected from various sources. It helps in cleaning the data by removing duplicates, filling missing values, and performing transformations. Pandas also facilitates the extraction of relevant features from the data, such as genres and ratings, which are necessary for clustering. Its data handling capabilities make it an indispensable tool for managing the data workflow, from initial collection to final preprocessing before applying machine learning algorithms.

Matplotlib

Matplotlib is a widely used library for creating static, animated, and interactive visualizations in Python. It is particularly useful for plotting graphs and charts toanalyze and interpret data. In the context of a movie recommendation system, Matplotlib is employed to visualize various aspects of the data and the results of the clustering process. For example, it can be used to create scatter plots to showhow movies are distributed across clusters, or to plot the distribution of ratings within clusters. Visualization is crucial for understanding the clustering outcomes, evaluating the effectiveness of the recommendation system, and presenting insights to stakeholders. Matplotlib's flexibility and wide range of plotting functions make it a valuable tool for data analysis and presentation inthe project.

4.4.3. JUPYTER NOTEBOOK

Jupyter Notebook is an open-source web application that enables users to create and share documents containing live code, equations, visualizations, and narrative text. It supports interactive computing by allowing users to write and execute code in a variety of programming languages, although Python is the most commonly used. The notebook format (.ipynb) combines code execution with rich text elements, making it an ideal tool for data analysis, exploratory research, and collaborative development.

Interactive Data Analysis and Exploration

In the context of developing a movie recommendation system, Jupyter Notebookexcels in facilitating interactive data analysis and exploration. Users can load datasets into the notebook using libraries such as Pandas, which provide powerful data manipulation capabilities. With Jupyter, you can immediately view and manipulate the data, perform exploratory data analysis, and execute code in an iterative manner. This interactive environment is particularly useful for pre-processing tasks such as cleaning, normalization, and feature extraction, allowing you to make adjustments and see the effects in real-time.

Visualization Capabilities

One of the significant advantages of Jupyter Notebook is its integration with visualization libraries like Matplotlib and Seaborn. These libraries allow you to create a wide range of plots and charts directly within the notebook. For a movierecommendation system, you can use these visualizations to explore the distribution of features, evaluate the results of clustering algorithms, and visualize the relationships between different data points. The ability to generate and view plots inline enhances the understanding of data patterns and the effectiveness of clustering outcomes.

Model Development and Experimentation

Jupyter Notebook is highly conducive to iterative model development and experimentation. When implementing the k-means clustering algorithm, you canwrite and test different versions of the clustering code, adjust parameters, and immediately observe the results. This iterative approach allows you to experiment with various configurations, such as the number of clusters, and evaluate their impact on clustering performance. The notebook's ability to run code in chunks and visualize results instantaneously supports a more dynamic and responsive development process.

Documentation and Reporting

A key feature of Jupyter Notebook is its ability to integrate narrative text with code. You can use Markdown cells to document your workflow, explain methodologies, and describe findings alongside your code and visualizations. This feature is invaluable for creating comprehensive reports that detail the development process of the movie recommendation system. It allows you to produce well-documented notebooks that not only showcase code and results but also provide context and explanations, making it easier to communicate insights and methodologies to stakeholders or team members.

Collaboration and Sharing

Jupyter Notebook facilitates collaboration and sharing of work. Notebooks can be version-controlled using tools like Git, which allows for tracking changes andcollaborating with others on the development of the recommendation system. Additionally, notebooks can be shared with colleagues or stakeholders in various formats, including HTML, PDF, and slides. This flexibility supports collaborative development and enables easy dissemination of findings and results

Integration with Other Tools

Jupyter Notebook supports integration with a range of other tools and libraries. For instance, it can work with interactive widgets through the ipywidgets library, allowing users to create interactive controls such as sliders and buttons. This feature can be used for interactive parameter tuning or to explore different aspects of the recommendation system in a dynamic way. Moreover, Jupyter supports various kernels for different programming languages, providing flexibility in using a diverse set of tools and libraries within the notebook environment.

Educational and Prototyping Benefits

Jupyter Notebook is widely used in educational settings due to its interactive nature and ability to combine code with explanatory text. For prototyping a movie recommendation system, the notebook serves as an excellent platform forrapid development and testing. The ability to document experiments, visualize results, and refine models interactively makes Jupyter an effective tool for both learning and prototyping complex systems.

6. SOFTWARE DESIGN

5.1 System Architecture:

The nodes involved are admin and user which stands as UI for the system. The deployment is performed as per the requirements of Hardware and software specified in the requirements phase.

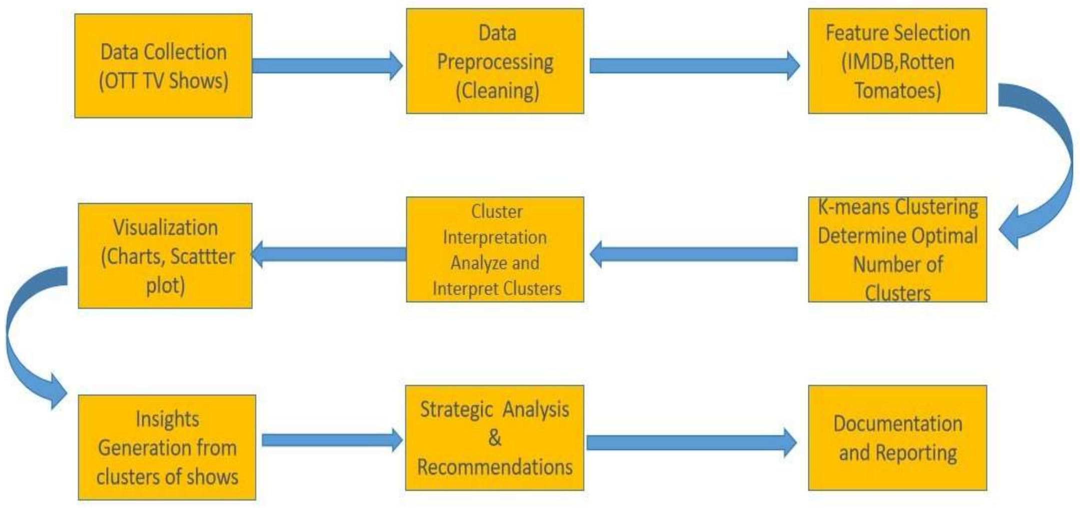

Data Collection (OTT TV Shows)
Data Preprocessing (Cleaning)
Feature Selection (IMDB,Rotten Tomatoes)
K-means Clustering Determine Optimal Number of Clusters
Cluster Interpretation Analyze and Interpret Clusters
Visualization (Charts, Scattter plot)
Insights Generation from clusters of shows
Strategic Analysis & Recommendations
Documentation and Reporting

Figure 5.1 Architecture

Data Collection

- Data is collected from various sources such as OTT platforms (Netflix, Hulu, etc.), IMDB and Rotten Tomatoes ratings.

Data Preprocessing

- This stage involves cleaning the data. This may include handling missing values,outliers, and inconsistencies in the data.

Data Visualization

Techniques like charts and scatter plots are used to visualize the data andidentify patterns. This can help in understanding user preferences and movie characteristics.

K-means Clustering

- This is where the k-means algorithm comes in. It groups similar movies together based on the selected features. The number of groups (k) is predefined.

Cluster Interpretation

- Once movies are grouped into clusters, the characteristics of each cluster are analyzed. This helps to identify what ties the movies in each cluster together – genre, director, theme, etc.

Insights Generation & Strategic Analysis

- Based on the cluster analysis, insights are generated about user preferences and movie characteristics. This can inform strategic decisions about content acquisition or movie recommendations.

Documentation & Reporting

- The findings are documented and reported. This may include reports on user preferences, movie clusters, and recommendations.

Recommendations

- Finally, the system uses the insights from the cluster analysis to recommend movies to users. Here’s a simplified example: If a user likes action movies by director A, the system might recommend other action movies by the same director or similar directors from the same cluster.

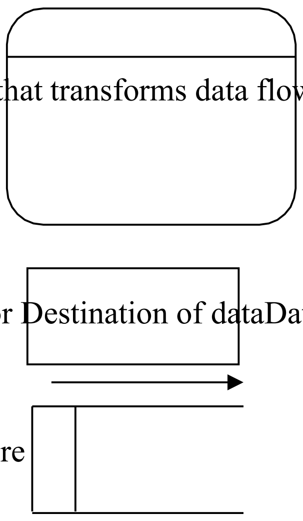
5.2 Data Flow Diagrams (DFD):

DFD SYMBOLS:

In the DFD, there are four symbols.

1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the informationflows
3. A circle or a bubble represents a process that transforms incoming data flow intooutgoing data flows.

4. An open rectangle is a data store, data at rest or a temporary repository of data.



Process that transforms data flow.

Source or Destination of dataData flow

Data Store

CONSTRUCTING A DFD:

Several rules of thumb are used in drawing DFD'S:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
3. When a process is exploded into lower level details, they are numberedThe names of data stores and destinations are written in capital letters. Process anddataflow names have the first letter of each work capitalized

   A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

   Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

SAILENT FEATURES OF DFD'S:

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.

2. The DFD does not indicate the time factor involved in any process whether thedataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

TYPES OF DATA FLOW DIAGRAMS:

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

CURRENT PHYSICAL:

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process thedata. Similarly data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

CURRENT LOGICAL:

The physical aspects at the system are removed as mush as possible so that the current system is reduced to its essence to the data and the processors that transforms them regardless of actual physical form.

NEW LOGICAL:

This is exactly like a current logical model if the user were completely happy with he user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

NEW PHYSICAL:

The new physical represents only the physical implementation of the newsystem.

RULES GOVERNING THE DFD'S:

PROCESS:

1)No process can have only outputs.

2)No process can have only inputs. If an object has only inputs than it must be a sink.

3)A process has a verb phrase label.

DATA STORE:

1)Data cannot move directly from one data store to another data store, a process mustmove data.

2)Data cannot move directly from an outside source to a data store, a process, whichreceives, must move data from the source and place the data into data store

3)A data store has a noun phrase label.

SOURCE OR SINK:

The origin and / or destination of data.

1)Data cannot move direly from a source to sink it must be moved by a process

2)A source and /or sink has a noun phrase land

DATA FLOW:

1)A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.

2)A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.

3)A data flow cannot go directly back to the same process it leads. There must be atleast one other process that handles the data flow produce some other data flow returns the original data into the beginning process.

4)A Data flow to a data store means update (delete or change).

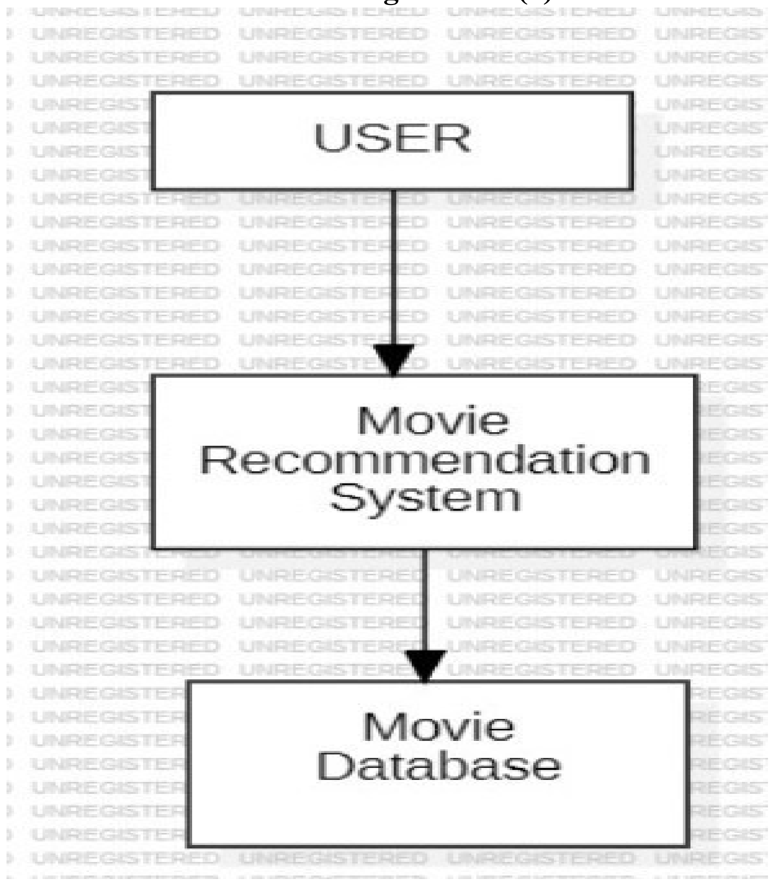
5)A data Flow from a data store means retrieve or use.

A data flow has a noun phrase label more than one data flow noun phrase can appear ona single arrow as long as all of the flows on the same arrow move together as one package.

DFD Diagrams:

Context Level Diagram (O level)

Figure 5.2.1(a): Level 0 DFD



Level 1 DFD:

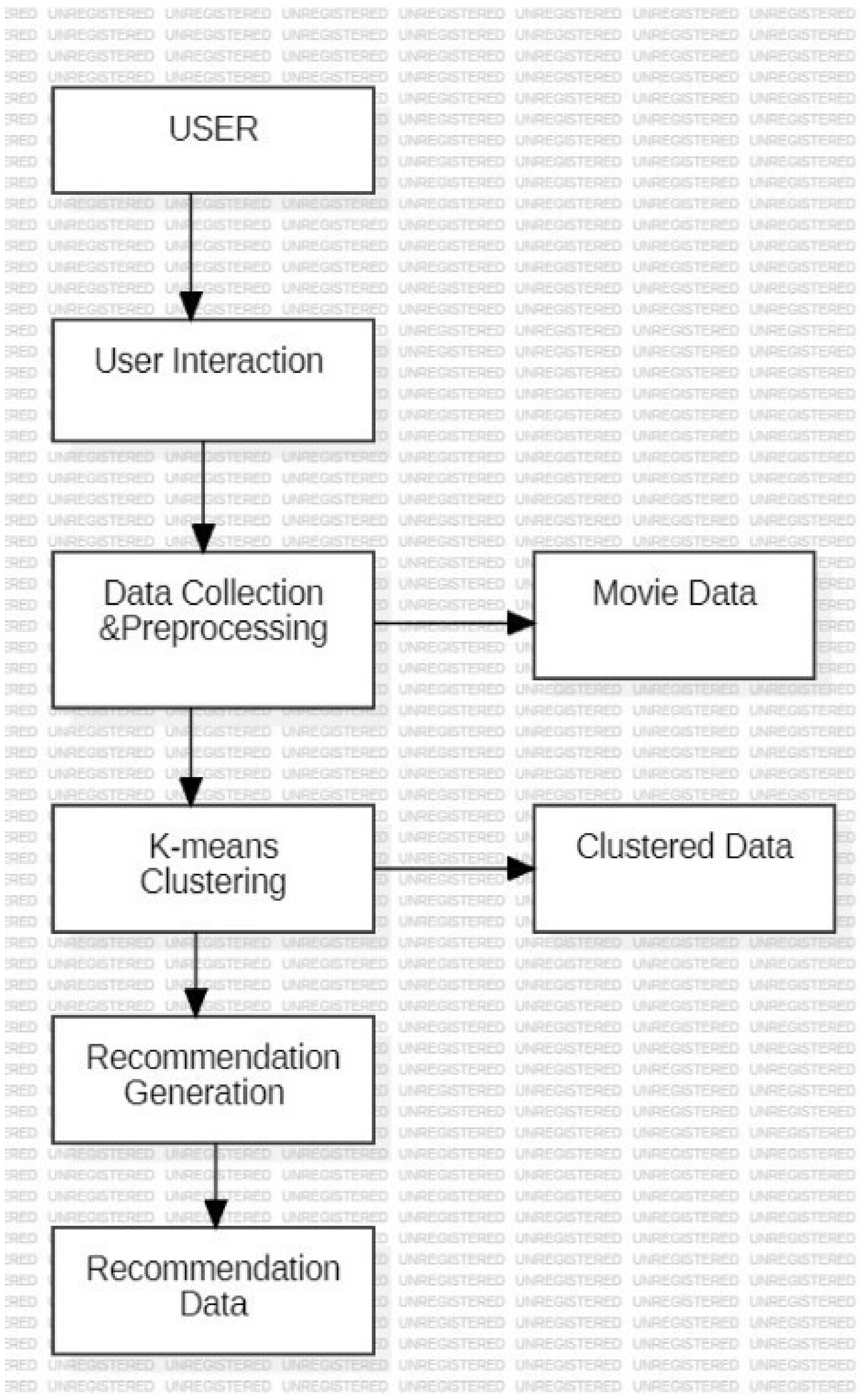


Figure 5.2.2(b): Level 1 DFD

5.3. UML Diagrams:

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

UML was created by Object Management Group (OMG) and UML 1.0 specificationdraft was proposed to the OMG in January 1997.

OMG is continuously putting effort to make a truly industry standard.

- UML stands for **U**nified **M**odeling **L**anguage.
- UML is a pictorial language used to make software blue prints.

UML Modeling Types:

It is very important to distinguish between the UML model. Different diagrams are used for different type of UML modeling. There are three important type of UML modelings:

5.3.1 Structural Things:

Structural things are classified into two types those are as follows:

1. Class Diagram
2. UseCase Diagrams

Class diagram:

Class diagrams are the most common diagrams used in UML. Class diagram consists of classes, interfaces, associations and collaboration. Class diagrams basically represent the object oriented view of a system which is static in nature. Active class is used in a class diagram to represent the concurrency of the system.

Class diagram represents the object orientation of a system. So it is generally used for development purpose. This is the most widely used diagram at the time of system construction.

The purpose of the class diagram is to model the static view of an application. The class diagrams are the only diagrams which can be directly mapped with object oriented languages and thus widely used at the time of construction

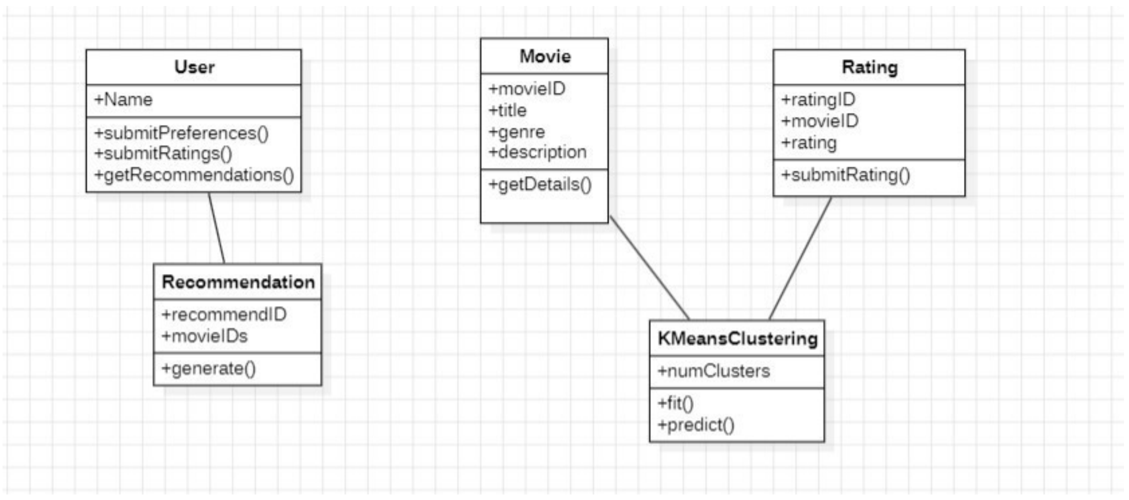


Figure 5.3.1 : Class Diagram.

Use Case Diagram:

Use case diagrams are considered for high level requirement analysis of asystem. So when the requirements of a system are analyzed the functionalities are captured in use cases.So we can say that uses cases are nothing but the system functionalities written in an organized manner. Now the second things which arerelevant to the use cases are the actors. Actors can be defined as something thatinteracts with the system.

The actors can be human user, some internal applications or may be some external applications. So in a brief when we are planning to draw an use case diagram we shouldhave the following items identified.

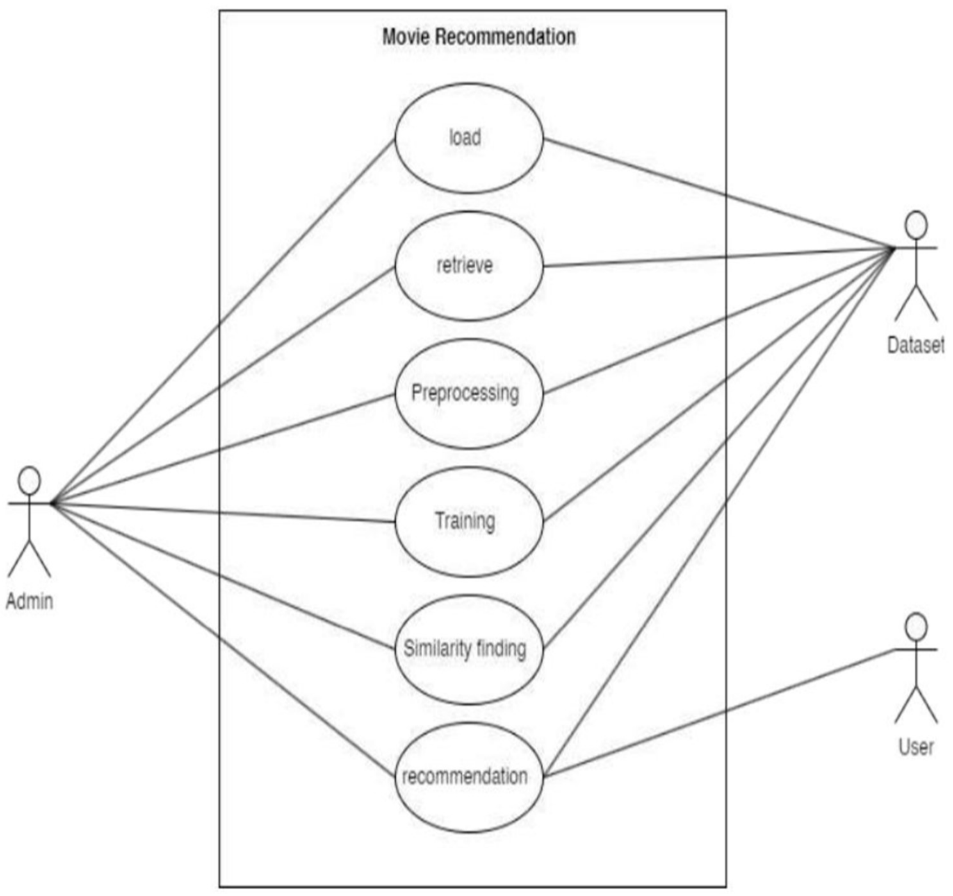
- Functionalities to be represented as an use case
- Actors
- Relationships among the use cases and actors.

Use case diagrams are drawn to capture the functional requirements of a system. So after identifying the above items we have to follow the following guidelines to draw an efficient use case diagram.

- The name of a use case is very important. So the name should be chosen in sucha way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships. Because the main purpose of thediagram is to identify requirements.
- Use note when ever required to clarify some important points…

Use Case:

Figure 5.3.2: Use Case Diagram



5.3.2 Behavioral Things

Behavioural things are considered as verbs of a model.These are the ‘dynamic' parts which describes how the model carry out its functionality with respect to time andspace. Behavioral things are classified into two types:

From the term Interaction, it is clear that the diagram is used to describe some type of interactions among the different elements in the model. This interaction is a part of dynamic behavior of the system.

Purpose of Interaction Diagrams

The purpose of interaction diagrams is to visualize the interactive behavior of the system. Visualizing the interaction is a difficult task. Hence, the solution is to use different types of models to capture the different aspects of the interaction.

Sequence and collaboration diagrams are used to capture the dynamic nature but froma different angle.

The purpose of interaction diagram is −

- To capture the dynamic behaviour of a system.
- To describe the message flow in the system.
- To describe the structural organization of the objects.
- To describe the interaction among objects.

How to Draw an Interaction Diagram?

As we have already discussed, the purpose of interaction diagrams is to capture the dynamic aspect of a system. So to capture the dynamic aspect, we need to understand what a dynamic aspect is and how it is visualized. Dynamic aspect can be defined as the snapshot of the running system at a particular moment

We have two types of interaction diagrams in UML. One is the sequence diagram and the other is the collaboration diagram. The sequence diagram captures thetime sequence of the message flow from one object to another and the collaboration diagram describes the organization of objects in a system taking part in the message flow.

Following things are to be identified clearly before drawing the interaction diagram

- Objects taking part in the interaction.
- Message flows among the objects.
- The sequence in which the messages are flowing.
- Object organization.

Following are two interaction diagrams Movie Recommendation System.The first diagram is a sequence diagram and the second is a collaboration diagram

The Sequence Diagram

The sequence diagram has four objects (User, Recommendation Engine, KMeansClustering and Database).

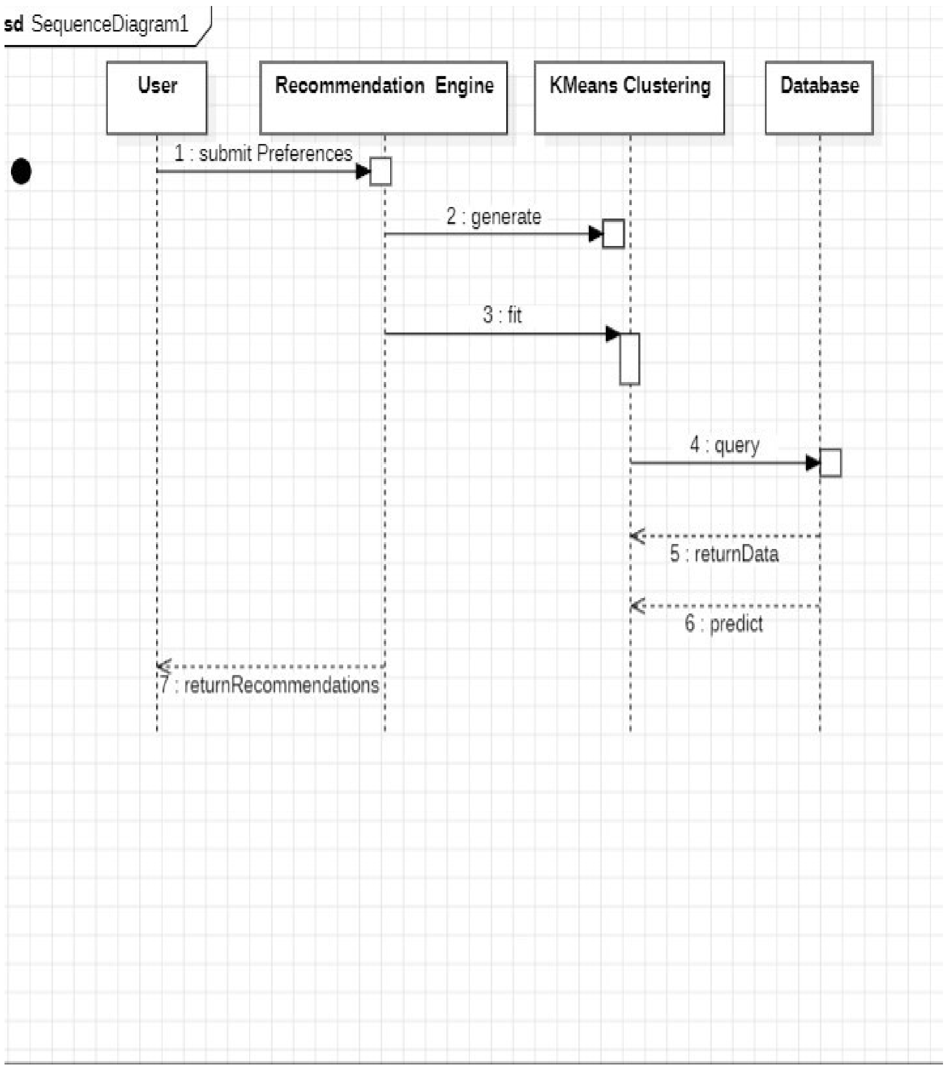


Figure 5.3.2(a): Sequence Diagram

Where to Use Interaction Diagrams?

We have already discussed that interaction diagrams are used to describe the dynamic nature of a system. Now, we will look into the practical scenarios wherethese diagrams are used. To understand the practical application, we need to understand the basic nature of sequence and collaboration diagram.

The main purpose of both the diagrams are similar as they are used to capture the dynamic behavior of a system. However, the specific purpose is more important to clarify and understand.

Sequence diagrams are used to capture the order of messages flowing from one object to another. Collaboration diagrams are used to describe the structural organization of the objects taking part in the interaction. A single diagram is not sufficient to describe the dynamic aspect of an entire system, so a set of diagrams are used to capture it as a whole.

Interaction diagrams are used when we want to understand the message flow and the structural organization. Message flow means the sequence of control flowfrom one object to another. Structural organization means the visual organization of the elements in a system.

Interaction diagrams can be used −

- To model the flow of control by time sequence.
- To model the flow of control by structural organizations.
- For forward engineering.
- For reverse engineering.

5.3.3 Relationships In The Uml:

In UML modeling, a relationship is a connection between two or more UML model elements that adds semantic information to a model.

In the product, you can use several UML relationships to define the structure between model elements. Examples of relationships include associations, dependencies, generalizations, realizations, and transitions.

Relationship	**Description**
Abstraction	An abstraction relationship is a dependency betweenmodel elements that represent the same concept at different levels of abstraction or from different viewpoints. You can add abstraction relationships to a model in several diagrams, including use-case, class, and component diagrams.
Aggregation	An aggregation relationship depicts a classifier as a part of, or as subordinate to, another classifier.
Association	An association relationship is a structural relationship between two model elements that shows that objects of one classifier (actor, use case, class, interface, node, or component) connect and can navigate to objects of another classifier. Even in bidirectional relationships, anassociation connects two classifiers, the primary (supplier) and secondary (client),
Binding	A binding relationship is a dependency relationship that assigns values to template parameters and generates anew model element from the template.
Communication path	A communication path is a type of association betweennodes in a deployment diagram that shows how the nodes exchange messages and signals.
Composition	A composition relationship represents a whole–part relationship and is a type of aggregation. A composition relationship specifies that the lifetime of the part classifier is dependent on the lifetime of the whole classifier.
Control flow	A control flow is a type of activity edge that models the movement of control from one activity node to another.

Dependency	A dependency relationship indicates that changes to onemodel element (the supplier or independent model element) can cause changes in another model element (the client or dependent model element). The supplier model element is independent because a change in the client does not affect it. The client model element depends on the supplier because a change to the supplier affects the client.
Deploy	A deploy relationship shows the specific componentthat an instance of a single node uses. In a UML model, a deploy relationship typically appears in deployment diagrams.
Directed association	A directed association relationship is an association that is navigable in only one direction and in which the control flows from one classifier to another (forexample, from an actor to a use case). Only one of the association ends specifies navigability.
Extend	An extend relationship between use cases indicates that one use case, the extended use case, can extend another use case, the base use case. An extend relationship has the option of using the extended use case.
Generalization	A generalization relationship indicates that a specialized(child) model element is based on a general (parent) model element. Although the parent model element can have one or more children, and any child model elementcan have one or more parents, typically a single parent has multiple children. In UML 2.0, several classes can constitute a generalization set of another class. Generalization relationships appear in class, component,and use-case diagrams.
Interface realization	An interface realization relationship is a specialized type of implementation relationship between a classifier and a provided interface. The interface realization relationship specifies that the realizing classifier must conform to the contract that the provided interface specifies.

Include	An include relationship between use cases specifies thatan including (or base) use case requires the behavior from another use case (the included use case). In an include relationship, a use case must use the included use case.
Manifestation	A manifestation relationship shows which model elements, such as components or classes, are manifestedin an artifact. The artifact manifests, or includes, a specific implementation for, the features of one or several physical software components.
Note attachment	A note attachment relationship connects a note or text box to a connector or shape. A note attachment indicates that the note or text box contains information that is relevant to the attached connector or shape.
Object flow	An object flow is a type of activity edge that models theflow of objects and data from one activity node to another.
Realization	A realization relationship exists between two model elements when one of them must realize, or implement, the behavior that the other specifies. The model element that specifies the behavior is the supplier, and the model element that implements the behavior is the client. In UML 2.0, this relationship is normally used to specify those elements that realize or implement the behavior of a component.
Usage	A usage relationship is a dependency relationship in which one model element requires the presence of another model element (or set of model elements) for itsfull implementation or operation. The model element that requires the presence of another model element is the client, and the model element whose presence is required is the supplier. Although a usage relationship indicates an ongoing requirement, it also indicates that the connection between the two model elements is not always meaningful or present.

Table 5.3.3(a) Relationships In the Uml

5.3.4 Diagrams In Uml:

Structural UML diagrams

- Class diagram
- Package diagram
- Object diagram

Types of UML Diagrams

The current UML standards call for 13 different types of diagrams: class, activity, object, use case, sequence, package, state, component, communication, composite structure, interaction overview, timing, and deployment.

These diagrams are organized into two distinct groups: structural diagrams and behavioral or interaction diagrams

.

Behavioral UML diagrams

- Activity diagram
- Sequence diagram
- Use case diagram

Class Diagram:

Class diagrams are the backbone of almost every object-oriented method,including UML. They describe the static structure of a system.

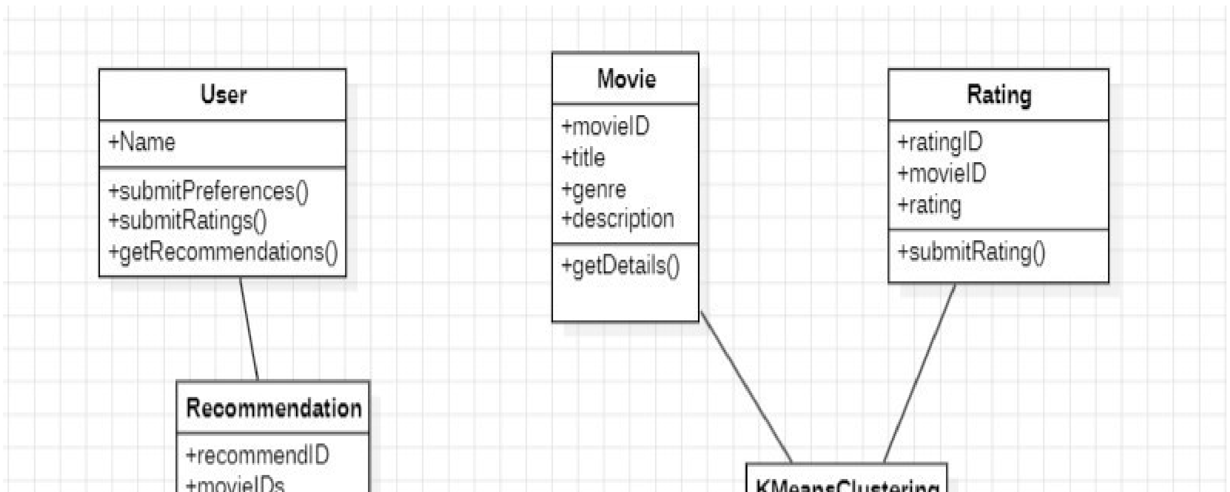


Figure 5.3.4 (a): Class Diagram

Use case Diagram:

Use case diagrams model the functionality of a system using actors and use cases.

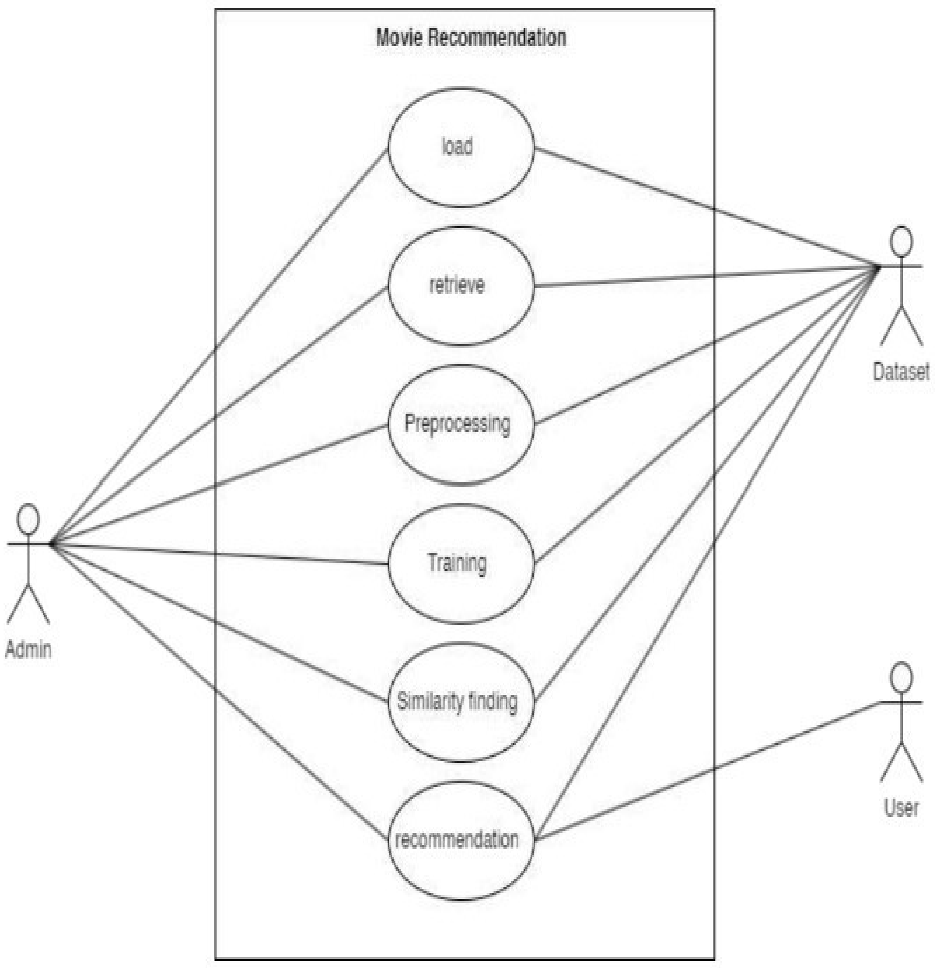


Figure5.3.4(b): Use case Diagram

Activity Diagram:

Activity diagrams illustrate the dynamic nature of a system by modeling the flow of control from activity to activity. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and internal operation.

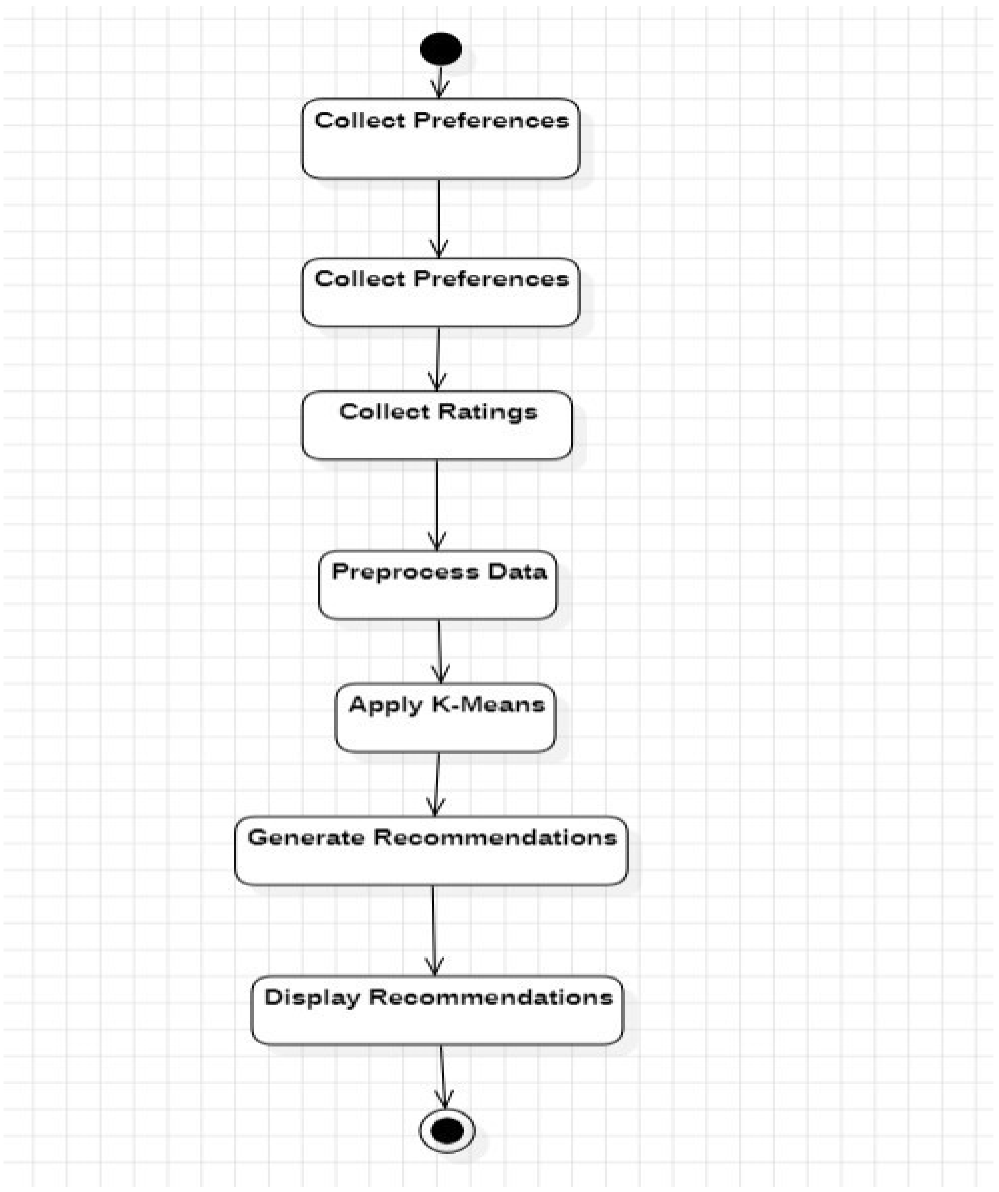


Figure 5.3.4 (c): Activity Diagram

Sequence diagrams:

Sequence diagrams describe interactions among classes in terms of an exchangeof messages over time.

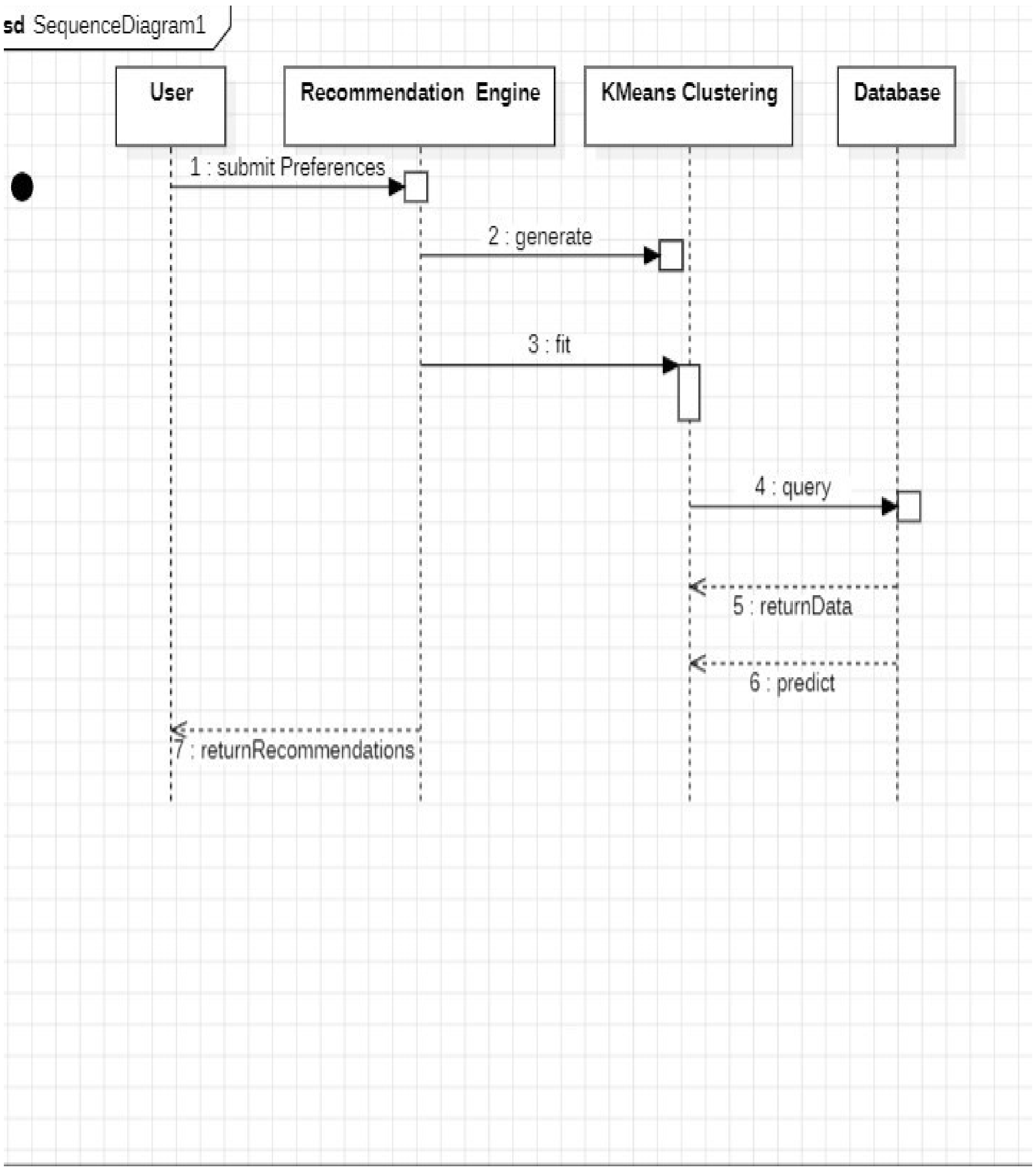


Figure 5.3.4 (d): Sequence Diagram

Component Diagram:

Component diagrams describe the organization of physical software components, including source code, run-time (binary) code, and executables.

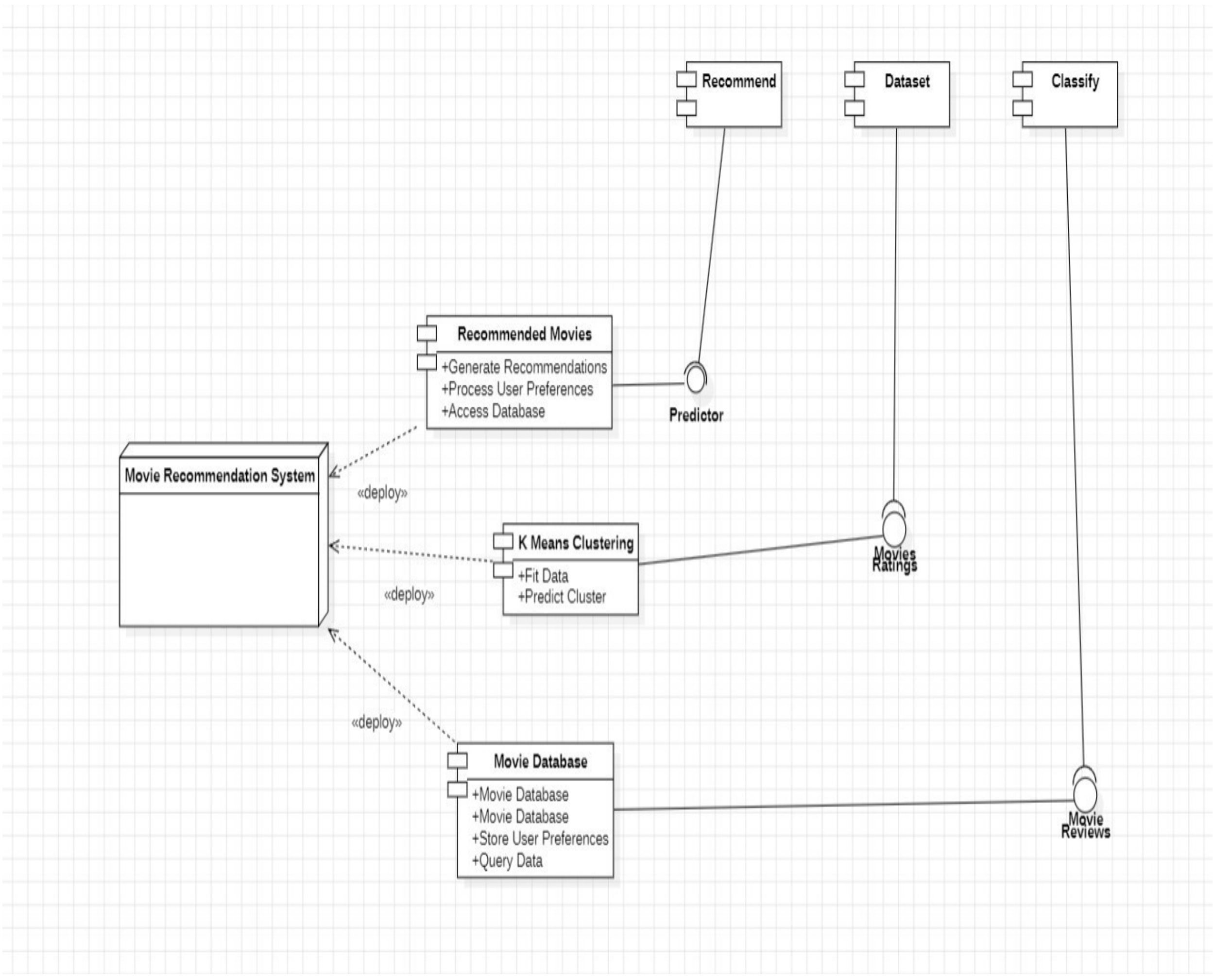
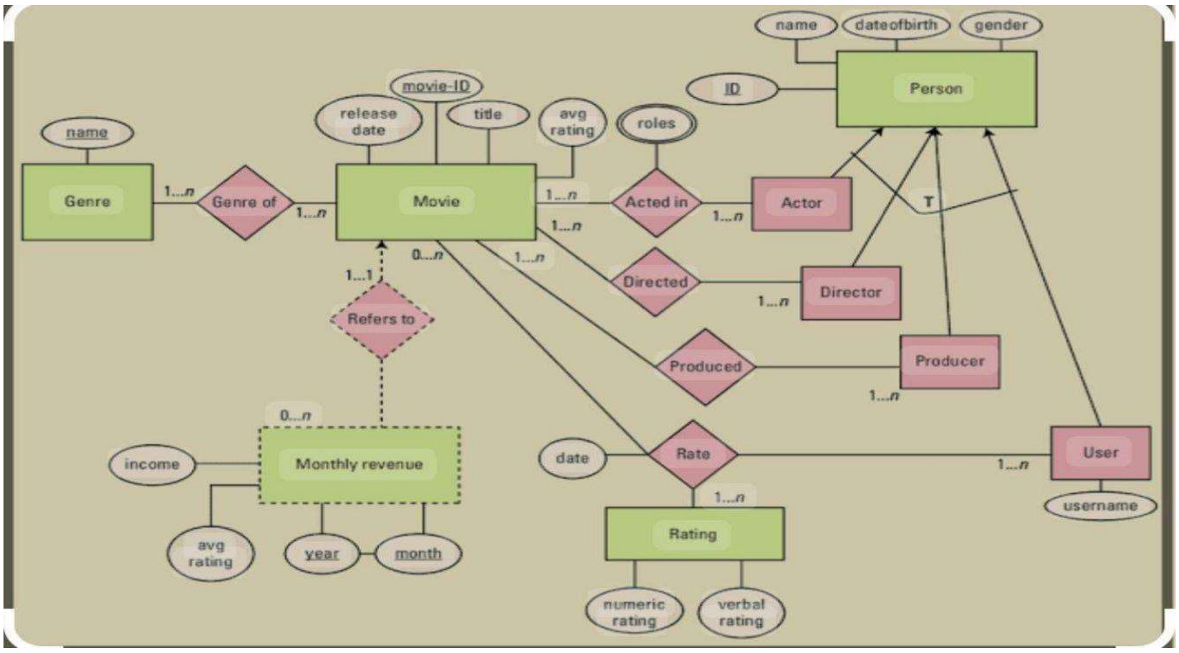


Figure 5.3.4 (e): Component Diagram

5.4 ER Diagram



Entities

- **User:** This entity represents a user of the movie recommendation system.

o Attributes:

▪ Demographics (optional): This could include things like age,gender, etc.

- **Movie:** This entity represents a movie in the system.

o Attributes:

▪ MovieID (unique identifier for the movie, likely a number)

▪ Title (text)

▪ Genre(s) (text)

▪ Release Year (optional) (number)

- **Rating:** This entity captures the rating a user gives to a movie.

o Attributes:

▪ MovieID (foreign key referencing the Movie table)

▪ Rating (numeric value representing the user's rating of the movie)

- **Cluster:** This entity represents a group of movies identified by the K-Means clustering algorithm.

  - Attributes:
    - ClusterID (unique identifier for the cluster, likely a number)
    - Description (optional) (text): This attribute provides a human-readable description of the movies within the cluster.

Relationships

- **Rates:** This is a many-to-many relationship between User and Movie. A user can rate many movies, and a movie can be rated by many users. The relationship is captured by the Rating entity, which has foreign keys referencingboth User and Movie.
- **Belongs_To:** This is a many-to-one relationship between User and Cluster. A user belongs to one cluster based on their movie ratings. This is represented bya foreign key in the User table referencing the ClusterID in the Cluster table.
- **Has_Movies:** This is a many-to-many relationship between Cluster and Movie. A cluster contains many movies that share similar characteristics based on user ratings. This relationship is implicit in the K-Means clustering process, where movies are assigned to clusters based on how similar they are according to the ratings data

Additional Considerations

- The ER diagram might include an additional entity for movie features (e.g., director, actors) if these are considered during K-Means clustering.
- The "Description" attribute in the Cluster entity is optional and can be used to provide a human-readable explanation of the types of movies the cluster represents.

In essence, the ER diagram outlines how the movie recommender system leverages K- Means clustering to group movies together based on user ratings.

6. CODING AND IMPLEMEMTATION

6.1. Sample code:

notebook.ipynb

1. Import and observe dataset

```
pip install pandas numpy scikit-learn matplotlib
```

```
!pip install nltk
import numpy as np
import pandas as pd
import nltk
        # Set seed for reproducibility
np.random.seed(5)
        # Read in IMDb and Wikipedia movie data (both in same file)
movies_df = pd.read_csv("movies.csv")
        print("Number of movies loaded: %s " % (len(movies_df)))
        # Display the data
movies_df
```

2. Combine Wikipedia and IMDb plot summaries

The dataset we imported currently contains two columns titled wiki_plot and imdb_plot. They are the plot found for the movies on Wikipedia and IMDb, respectively. The text in the two columns is similar, however, they are often written in different tones and thus provide context on a movie in a different manner of linguistic expression. Further, sometimes the text in one column may mention a feature of the plot that is not present in the other column. For example, consider the following plot extracts from *The Godfather*:

Wikipedia: "On the day of his only daughter's wedding, Vito Corleone"

IMDb: "In late summer 1945, guests are gathered for the wedding reception of Don Vito Corleone's daughter Connie"

While the Wikipedia plot only mentions it is the day of the daughter's wedding, the IMDb plot also mentions the year of the scene and the name of the daughter.

Let's combine both the columns to avoid the overheads in computation associated with extra columns to process.

Code:

```
    movies_df["plot"] = movies_df["wiki_plot"].astype(str) + "\n" + \
                movies_df["imdb_plot"].astype(str)

# Inspect the new DataFrame
movies_df.head()
```

3. Tokenization

Tokenization is the process by which we break down articles into individual sentences or words, as needed. Besides the tokenization method provided by NLTK, we might have to perform additional filtration to remove tokens which are entirely numeric values or punctuation.

While a program may fail to build context from "While waiting at a bus stop in 1981" (*Forrest Gump*), because this string would not match in any dictionary, it is possible to build context from the words "while", "waiting" or "bus" because they are present in the English dictionary. Let us perform tokenization on a small extract from *The Godfather*.

Code:

```
import nltk
nltk.download('punkt')
sent_tokenized = [sent for sent in nltk.sent_tokenize("""
                Today (May 19, 2016) is his only daughter's wedding.
                Vito Corleone is the Godfather.
                """)]
      # Word Tokenize first sentence from sent_tokenized, save as words_tokenized
words_tokenized = [word for word in nltk.word_tokenize(sent_tokenized[0])]
      # Remove tokens that do not contain any letters from words_tokenized
import re
      filtered = [word for word in words_tokenized if re.search('[a-zA-Z]', word)]
      # Display filtered words to observe words after tokenization
filtered
```

4. Stemming

Stemming is the process by which we bring down a word from its different forms to the root word. This helps us establish meaning to different forms of the same words without having to deal with each form separately. For example, the words 'fishing', 'fished', and 'fisher' all get stemmed to the word 'fish'.

Consider the following sentences:

"Young William Wallace witnesses the treachery of Longshanks" ~ *Gladiator*

"escapes to the city walls only to witness Cicero's death" ~ *Braveheart*

Instead of building separate dictionary entries for both witnesses and witness, which mean the same thing outside of quantity, stemming them reduces them to 'wit'

Code:

```
from nltk.stem.snowball import SnowballStemmer
      # Create an English language SnowballStemmer object
stemmer = SnowballStemmer("english")
      # Print filtered to observe words without stemming
print("Without stemming: ", filtered)
      # Stem the words from filtered and store in stemmed_words
```

```
stemmed_words = [stemmer.stem(t) for t in filtered]
        # Print the stemmed_words to observe words after stemming
print("After stemming:   ", stemmed_words)
```

5. Club together Tokenize & Stem

We are now able to tokenize and stem sentences. But we may have to use the two functions repeatedly one after the other to handle a large amount of data, hence we can think of wrapping them in a function and passing the text to be tokenized and stemmed as the function argument. Then we can pass the new wrapping function, which shall perform both tokenizing and stemming instead of just tokenizing, as the tokenizer argument while creating the TF-IDF vector of the text.

What difference does it make though? Consider the sentence from the plot of *The Godfather*: "Today (May 19, 2016) is his only daughter's wedding." If we do a 'tokenize-only' for this sentence, we have the following result:

'today', 'may', 'is', 'his', 'only', 'daughter', "'s", 'wedding'

But when we do a 'tokenize-and-stem' operation we get:

'today', 'may', 'is', 'his', 'onli', 'daughter', "'s", 'wed'

All the words are in their root form, which will lead to a better establishment of meaning as some of the non-root forms may not be present in the NLTK training corpus.

```
# Define a function to perform both stemming and tokenization
def tokenize_and_stem(text):
 # Tokenize by sentence, then by word
   tokens = [word for sent in nltk.sent_tokenize(text) for word in nltk.word_tokenize(sent)]
  # Filter out raw tokens to remove noise
filtered_tokens = [token for token in tokens if re.search('[a-zA-Z]', token)]
 # Stem the filtered_tokens
stems = [stemmer.stem(t) for t in filtered_tokens]
return stems
words_stemmed = tokenize_and_stem("Today (May 19, 2016) is his only daughter's wedding.")
        print(words_stemmed)
```

6. Create TfidfVectorizer

Computers do not *understand* text. These are machines only capable of understanding numbers and performing numerical computation. Hence, we must convert our textual plot summaries to numbers for the computer to be able to extract meaning from them. One simple method of doing this would be to count all the occurrences of each word in the entire vocabulary and return the counts in a vector. Enter CountVectorizer.

Consider the word 'the'. It appears quite frequently in almost all movie plots and will have a high count in each case. But obviously, it isn't the theme of all the movies! Term Frequency-Inverse Document Frequency (TF-IDF) is one method which overcomes the shortcomings of CountVectorizer. The Term Frequency of a word is the measure of how often it appears in a document, while the Inverse Document Frequency is the parameter which reduces the importance of a word if it frequently appears in several documents.

For example, when we apply the TF-IDF on the first 3 sentences from the plot of *The Wizard of Oz*, we are told that the most important word there is 'Toto', the pet dog of the lead character. This is because the movie begins with 'Toto' biting someone due to which the journey of Oz begins!

In simplest terms, TF-IDF recognizes words which are unique and important to any given document.

Code:

```
from sklearn.feature_extraction.text import TfidfVectorizer
        # Instantiate TfidfVectorizer object with stopwords and tokenizer
# parameters for efficient processing of text
tfidf_vectorizer = TfidfVectorizer(max_df=0.8, max_features=200000,
                          min_df=0.2, stop_words='english',
                          use_idf=True, tokenizer=tokenize_and_stem,
                          ngram_range=(1,3))
```

7. Fit transform TfidfVectorizer

Once we create a TF-IDF Vectorizer, we must fit the text to it and then transform the text to produce the corresponding numeric form of the data which the computer will be able to understand and derive meaning from. To do this, we use the fit_transform() method of the TfidfVectorizer object.

If we observe the TfidfVectorizer object we created, we come across a parameter stopwords. 'stopwords' are those words in a given text which do not contribute considerably towards the meaning of the sentence and are generally grammatical filler words. For example, in the sentence 'Dorothy Gale lives with her dog Toto on the farm of her Aunt Em and Uncle Henry', we could drop the words 'her' and 'the', and still have a similar overall meaning to the sentence. Thus, 'her' and 'the' are stopwords and can be conveniently dropped from the sentence.

On setting the stopwords to 'english', we direct the vectorizer to drop all stopwords from a pre-defined list of English language stopwords present in the nltk module. Another parameter, ngram_range, defines the length of the ngrams to be formed while vectorizing the text.

Code:

```
tfidf_matrix = tfidf_vectorizer.fit_transform([x for x in movies_df["plot"]])
        print(tfidf_matrix.shape)
```

8. Import KMeans and create clusters

A good basis of clustering in our dataset could be the genre of the movies. Say we could have a cluster '0' which holds movies of the 'Drama' genre. We would expect movies like *Chinatown* or *Psycho* to belong to this cluster. Similarly, the cluster '1' in this project holds movies which belong to the 'Adventure' genre (*Lawrence of Arabia* and the *Raiders of the Lost Ark*, for example).

K-means is an algorithm which helps us to implement clustering in Python. The name derives from its method of implementation: the given sample is divided into ***K*** clusters where each cluster is denoted by the ***mean*** of all the items lying in that cluster.

Code:

```
from sklearn.decomposition import PCA
import numpy as np
        # Reduce dimensionality to 2D using PCA
pca = PCA(n_components=2)
reduced_matrix = pca.fit_transform(tfidf_matrix.toarray())
        # Create a DataFrame for the reduced data
reduced_df = pd.DataFrame(reduced_matrix, columns=['PC1', 'PC2'])
reduced_df['cluster'] = clusters

# Plot the 2D clusters
plt.figure(figsize=(10, 8))
sns.scatterplot(x='PC1', y='PC2', hue='cluster', data=reduced_df, palette='viridis', s=100)
plt.title('2D Plot of Clusters (PCA)')
plt.show()
# Import necessary libraries
from sklearn.cluster import KMeans
import pandas as pd
        # Assuming tfidf_matrix and movies_df are already defined
        # Create a KMeans object with 5 clusters and fit the model
km = KMeans(n_clusters=5)
km.fit(tfidf_matrix)
        # Get the cluster labels for each movie
```

```
clusters = km.labels_.tolist()
        # Add the cluster information to the DataFrame
movies_df["cluster"] = clusters
        # Display the number of films per cluster
cluster_counts = movies_df['cluster'].value_counts()
print(cluster_counts)
        # Plot the counts for better visualization
import matplotlib.pyplot as plt
cluster_counts.plot(kind='bar')
plt.title('Number of Films per Cluster')
plt.xlabel('Cluster')
plt.ylabel('Number of Films')
plt.show()
# Function to print movies in each cluster
def print_movies_in_clusters(df, cluster_label_column='cluster', title_column='title'):
    for cluster in sorted(df[cluster_label_column].unique()):
        print(f"\nCluster {cluster}:")
        movies_in_cluster = df[df[cluster_label_column] == cluster][title_column].tolist()
        for movie in movies_in_cluster:
            print(movie)
        # Print the names of the movies in each cluster
print_movies_in_clusters(movies_df)
        # Function to find similar movies in the same cluster
def find_similar_movies(movie_title, df, cluster_label_column='cluster', title_column='title'):
    # Check if the movie exists in the DataFrame
    if movie_title not in df[title_column].values:
        return f"Movie '{movie_title}' not found in the dataset."
     # Get the cluster label of the given movie
    cluster_label = df[df[title_column] == movie_title][cluster_label_column].values[0]
     # Get all movies in the same cluster
    similar_movies = df[df[cluster_label_column] == cluster_label][title_column].tolist()
     # Remove the given movie from the list
    if movie_title in similar_movies:
        similar_movies.remove(movie_title)
     return similar_movies
# Example usage: Prompt user for a movie title and print similar movies
movie_title = input("Enter a movie title: ")  # User input for movie title
```

```
similar_movies = find_similar_movies(movie_title, movies_df)
        if isinstance(similar_movies, list):
   print(f"\nMovies similar to '{movie_title}':")
   for movie in similar_movies:
       print(movie)
else:
   print(similar_movies)
```

7.SYSTEM TESTING

7.1. Testing Strategies:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and userexpectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

1. Unit Testing

Unit testing is a fundamental step in software development aimed at ensuring that individual components or functions operate correctly in isolation before they are integrated into a larger system. For a movie clustering project, this involves testing key components like PCA (Principal Component Analysis), KMeans clustering, DataFrames, plotting functions, and helper functions.

Dimensionality Reduction (PCA) Testing

The primary objective of testing PCA is to ensure it correctly reduces the dimensionality of the dataset to the desired number of dimensions, typically two for visualization purposes. After applying PCA, the transformed data should have a shape reflecting this reduction. For example, if you started with a dataset containing 200 samples, the PCA output should have a shape of (200, 2), indicating that each of the 200 samples is now represented in a 2-dimensional space. This shape validation confirms that PCA has executed as expected.

In addition to verifying the shape, you should check that the output data consists of numerical values. This is crucial because PCA's primary function is to produce numerical representations of the data in a reduced dimensional space. The data type of the output should be appropriate for further numerical analysis, typically float. Validating the data type ensures that subsequent processing steps, such as clustering or visualization, can be performed without issues. A useful approach to further validate PCA's effectiveness is to plot the reduced dimensions. This visualization allows you to see if the data maintains its structure or clusters in a meaningful way after dimensionality reduction.

KMeans Clustering Testing

Testing the KMeans clustering algorithm involves verifying that it performs clustering correctly according to the specified parameters. When initializing KMeans with a set number of clusters, for example, five, you need to ensure that the algorithm produces exactly that number of clusters. This involves checking that the number of unique clusters in the output matches the expected count. Each data point should be assigned to one of these clusters.

Another important aspect of testing KMeans involves checking the validity of cluster labels. These labels should fall within a range of integers from 0 to n_clusters - 1. For instance, with five clusters, valid labels should be between 0 and 4. Ensuring that no labels fall outside this range helps confirm that the algorithm is functioning correctly.

Furthermore, it's essential to verify that the number of cluster labels matches the number of data points in the dataset. Each data point should have a corresponding cluster label, ensuring that every point is assigned to a cluster.

DataFrame and Plotting Functions Testing

Testing the DataFrame and plotting functions involves ensuring that DataFrames are correctly populated with the results of PCA and KMeans clustering and that the plots accurately visualize the data.

After performing PCA and clustering, you should verify that the resulting DataFrame contains all necessary columns, such as 'PC1', 'PC2', and 'cluster'. These columns should correctly reflect the reduced dimensions and clustering results. Ensuring the presence of these columns confirms that the DataFrame is appropriately updated.

For plotting functions, you need to confirm that they generate plots without errors and that these plots accurately represent the data. For example, when plotting the number of films per cluster, the plot should display the correct number of bars representing each cluster and accurately label these bars. Testing these functions involves running the plotting code and visually inspecting the output to ensure it matches expectations.

Helper Functions Testing

Helper functions such as those for printing movies in clusters and finding similar movies also need to be tested. For the function that prints movies in clusters, you should verify that it correctly outputs the movies for each cluster based on the clustering results. This involves checking that the list of movies printed for each cluster matches the expected movies that belong to that cluster.

For the function that finds similar movies, you need to ensure it accurately identifies and returns movies from the same cluster as a specified movie, excluding the input movie itself. Testing should include cases where the specified movie exists and where it does not, ensuring that the function handles both scenarios appropriately and returns correct results.

2. Integration Testing

Integration testing is crucial for ensuring that various components of the system work together as intended. This involves testing the entire pipeline from data input to clustering, PCA transformation, and visualization.

End-to-End Workflow Testing

The end-to-end workflow test focuses on validating that the complete process works seamlessly. Start by executing the entire pipeline, which includes data input, PCA transformation, KMeans clustering, and visualization. You need to confirm that the data flows correctly through each stage of this pipeline. For example, ensure that the data transformed by PCA is accurately reflected in the DataFrame, that KMeans clustering is applied as expected, and that the results are visualized correctly in plots.

To conduct this test, simulate a typical workflow with sample data and ensure that each step PCA, clustering, and plotting—executes without errors. Verify that the results produced at each stage align with expectations and that there are no issues with data handling or processing.

3. Exploratory Testing

Exploratory testing aims to uncover unexpected issues or behaviors by exploring different scenarios and edge cases not covered by formal tests.

Varied Inputs Testing

Testing with varied inputs involves checking how the system handles different scenarios, including edge cases. Start by testing the system with empty datasets to see how it responds. An empty dataset should be handled gracefully, with appropriate messages or errors indicating the absence of data.

Similarly, test how the system deals with incomplete or corrupted data. For example, if some data points are missing features, verify that the system either processes this data appropriately or raises informative errors. This ensures that the system is robust and can handle unexpected or suboptimal input conditions.

User Input Testing

User input testing involves checking how the system handles various types of user inputs. For example, test the system's response to movie titles that do not exist in the dataset. The system should display appropriate error messages indicating that the movie is not found.

Additionally, test how the system manages unusually long or short movie titles. Ensure that these inputs do not break the system and that the system processes them correctly. This helps ensure that the user interface and input handling are resilient to different types of user input.

Performance Testing

Performance testing evaluates how the system performs with large datasets. Measure the execution time and resource usage as you increase the size of the dataset. For instance, test the system with datasets ranging from hundreds to thousands of movies and observe how the clustering and plotting functions handle the increased load.

Assessing scalability and efficiency is crucial to ensure that the system remains responsive and performs well even with large volumes of data. This involves monitoring performance metrics and ensuring that the system can handle large datasets without significant slowdowns or failures.

4. Regression Testing

Regression testing ensures that new changes to the code do not introduce new bugs or disrupt existing functionality. This involves re-running existing tests after code changes to verify that the changes have not affected the overall system.

Continuous Testing

Implementing an automated test suite is essential for continuous testing. This suite should re-run all tests automatically after each code change to detect any regressions early. Integrating automated testing into your version control system and continuous integration/continuous deployment (CI/CD) pipelines helps streamline this process and ensures that code changes are validated promptly.

Test Coverage

To ensure comprehensive test coverage, use code coverage analysis tools to measure how much of the codebase is exercised by your tests. This analysis helps identify any untested code paths and ensures that all critical parts of the application are covered by tests. Comprehensive test coverage is crucial for maintaining the robustness of the system as it evolves.

5. White Box Testing

White box testing involves examining the internal logic and structure of the codebase to ensure that algorithms and error handling mechanisms work as intended.

Algorithm Verification

To verify algorithms like PCA and KMeans, test all possible execution paths and branches within the code. For PCA, ensure that the steps involved in dimensionality reduction are correctly implemented and that each step performs as expected. For KMeans, verify the initialization of centroids and the update process during iterations to ensure the algorithm functions correctly.

Error Handling and Validation

Check that the code handles errors and invalid inputs gracefully. This involves verifying that exceptions are properly caught and managed. Test how the system deals with various types of invalid or edge-case inputs, such as missing values or corrupted data, to ensure that it operates reliably under all conditions.

6. Black Box Testing

Black box testing focuses on validating the application’s functionality based on user requirements without knowledge of the internal code. This type of testing is crucial for ensuring that the system meets user expectations and performs as intended.

Functional Testing

Functional testing involves ensuring that the system meets its functional requirements and operates as expected. Validate that PCA correctly reduces data to two dimensions and that KMeans clustering produces accurate results. Ensure that all functional aspects of the application align with user expectations and requirements.

Output Verification

Confirm that outputs are accurate and meet user expectations. This involves verifying that DataFrames correctly reflect clustering results and that plots accurately visualize the data. For example, check that plots of clustered data display correctly and that DataFrames contain the expected results from PCA and clustering.

Boundary and Edge Case Testing

Test the system's behavior with boundary values and edge cases to ensure it operates correctly under all conditions. For example, verify how the system handles extreme values or unusual input scenarios, such as very large or very small numbers, to ensure it remains stable and performs as expected.

User Interaction Testing

Verify that interactive features, such as search functionality, work correctly. Ensure that the system returns accurate results for user queries and handles errors gracefully when invalid input is provided. This includes checking that the search functionality correctly identifies movies and provides appropriate feedback for non-existent or incorrect queries.

The comprehensive testing strategies for the Best streaming shows segmentation analysis project ensure its reliability, accuracy, and performance by combining unit, integration, exploratory, regression, white box, and black box testing. This multifaceted approach validates individual components, their interactions, and the system's overall functionality, while also uncovering edge cases and performance issues. By rigorously assessing both internal logic and user-facing features, these strategies help deliver a robust, stable, and user-friendly application that meets functional requirements and handles various scenarios effectively.

8.EXPERIMENTAL RESULTS

8.1. Implementation Description:

The project represents a sophisticated approach to data analysis, aiming to uncover underlying patterns and groupings within a dataset. The implementation of this project involves several meticulously planned stages, from data preparation and processing to clustering, visualization, and user interaction. This detailed description provides an in-depth look at each aspect of the project.

Data Preparation

The project begins with data preparation, a foundational step critical to ensuring the quality and effectiveness of the clustering process. The first task is data collection, where a dataset relevant to the analysis is gathered. For a movie clustering project, this typically involves collecting data from movie databases or APIs, such as IMDb, TMDb, or OMDb. These sources provide a wealth of information, including movie titles, descriptions, genres, and ratings.

Once the data is collected, the next step is data cleaning and preprocessing. This involves examining the dataset for missing or incomplete values. For instance, if certain movie descriptions or ratings are missing, it is essential to address these gaps. Depending on the extent of missing data, you might choose to impute the missing values with averages or medians, or you might remove records with significant gaps.

Textual data, such as movie descriptions, requires special processing to be transformed into a format suitable for clustering algorithms. This process includes tokenization, which breaks down text into individual words or tokens. Libraries like NLTK or SpaCy can facilitate this process. Next, stop words removal is performed to eliminate common words that do not add significant meaning to the text analysis. Subsequently, stemming or lemmatization reduces words to their root form, standardizing variations of the same word.

After preprocessing, the text data is transformed into numerical features using TF-IDF vectorization. This method calculates the term frequency-inverse document frequency for each word, providing a weighted representation of words that highlights their importance in the context of the entire dataset.

Feature normalization is another critical step, ensuring that all features contribute equally to the clustering process. Features with different scales can disproportionately affect the clustering results. To address this, normalization techniques such as standardization (scaling features to have a mean of zero and a standard deviation of one) or Min-Max scaling (scaling features to a range between 0 and 1) are applied.

Dimensionality Reduction

With the data prepared, the next phase is dimensionality reduction. This step simplifies the dataset by reducing the number of dimensions while retaining as much variance as possible. Principal Component Analysis (PCA) is employed for this purpose. PCA identifies principal components that capture the maximum variance in the data, transforming the feature matrix into a lower-dimensional space.

In practice, PCA is fit to the feature matrix obtained from the TF-IDF vectorization, and the data is transformed into a two-dimensional space for easier visualization. The first two principal components are selected to provide a simplified yet effective representation of the data. By examining the explained variance ratio, one can gauge how well the reduced dimensions represent the original data. This step is crucial for making the clustering results more interpretable.

Clustering

The core of the project is the K-means clustering algorithm, which groups data points into KKK clusters based on their similarity. The goal is to partition the data such that the within-cluster variance is minimized, and the separation between clusters is maximized.

The implementation begins by initializing the K-means algorithm with a predetermined number of clusters, KKK. This number can be chosen based on domain knowledge or through methods such as the Elbow Method, which involves plotting the within-cluster sum of squares (WCSS) against different values of KKK to identify the optimal number of clusters.

Next, the K-means algorithm is fitted to the 2D data obtained from PCA. The algorithm iterates to assign each data point to the nearest centroid and update the centroids based on the assigned points. This iterative process continues until the centroids stabilize or the assignments no longer change. The result is a set of cluster labels, where each data point is assigned to one of the KKK clusters.

To assess the quality of the clustering, evaluation metrics such as the Silhouette Score or Inertia can be used. The Silhouette Score measures how similar each data point is to its own cluster compared to other clusters, while Inertia represents the sum of squared distances of samples to their nearest cluster center.

Visualization

Visualization plays a crucial role in interpreting and presenting the clustering results. Data visualization involves creating plots that effectively communicate the results of the clustering process.

One key visualization is the PCA scatter plot, where the 2D PCA-reduced data is plotted, with each point colored according to its cluster label. This scatter plot allows for a clear visual representation of how data points are distributed across different clusters.

Another important visualization is the cluster distribution plot, which is typically a bar chart showing the number of data points in each cluster. This helps in understanding the relative sizes of the clusters and identifying any potential imbalances.

Libraries such as Matplotlib and Seaborn are used to create these visualizations. Matplotlib provides the basic plotting capabilities, while Seaborn enhances the visual aesthetics and allows for more advanced statistical graphics.

Cluster analysis is an additional aspect of visualization where the characteristics of each cluster are examined. By analyzing the contents of each cluster, one can identify patterns or themes. For example, if clustering movies, it might be observed that one cluster predominantly contains action films, while another contains romantic comedies. Reviewing the movies within each cluster helps in understanding the nature of the clusters and validating the results.

User Interaction

Adding user interaction features enhances the practical utility of the clustering results. One such feature is the ability to find similar movies. This functionality allows users to input a movie title and retrieve a list of other movies within the same cluster. This is achieved by developing a function that takes the movie title as input, retrieves its cluster label, and then lists other movies in the same cluster. This feature provides users with recommendations based on clustering results.

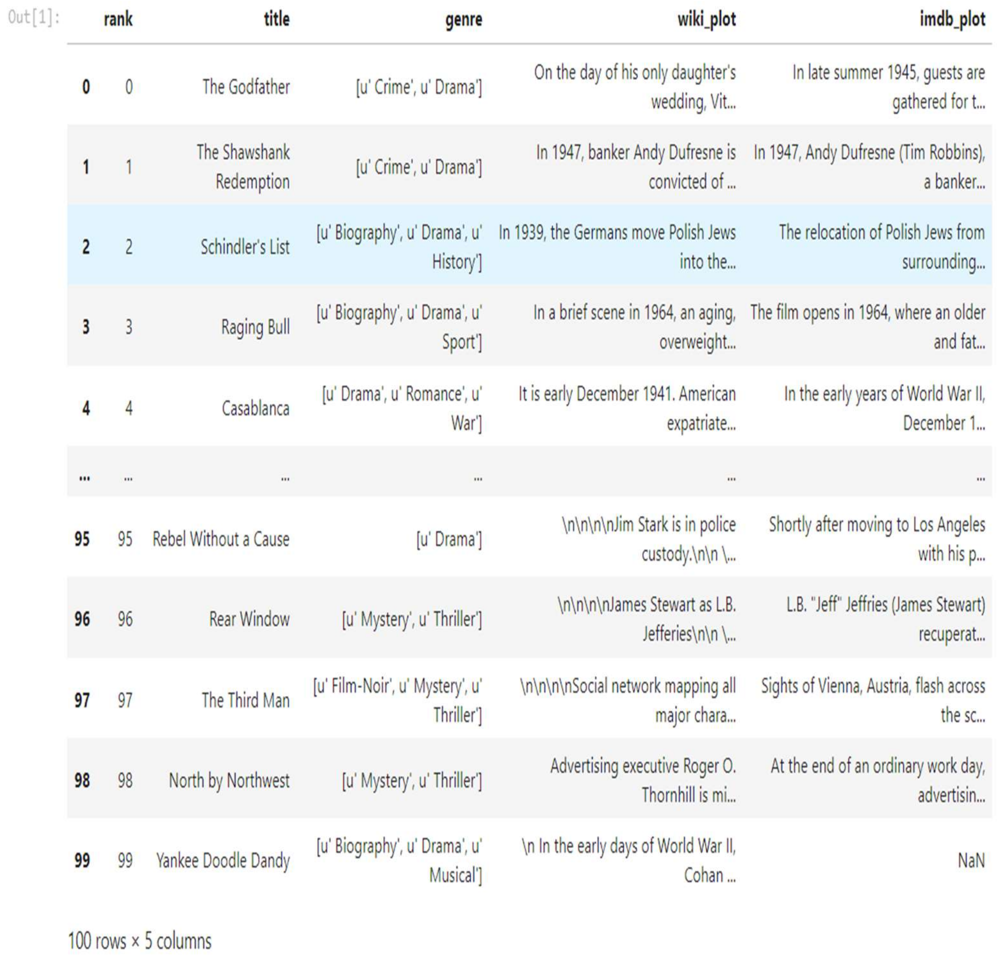
Input validation is crucial to ensure that user inputs are handled correctly. The system should validate user inputs, such as movie titles, to prevent errors. If the input movie title is not found, the system should provide informative messages or suggestions to guide users.

Additional Considerations

Performance optimization is an essential aspect of implementing the K-means clustering project,especially when dealing with large datasets. Ensuring that the system is efficient and scalable involves optimizing the clustering and dimensionality reduction algorithms. This may include using optimized libraries or parallel processing techniques to speed up computations. Testing the system with progressively larger datasets helps evaluate its performance and scalability.

Error handling and robustness are also critical for maintaining the reliability of the system. Implementing robust error handling mechanisms ensures that exceptions are managed effectively, whether they arise during data processing, clustering, or user interactions. Data integrity checks throughout the process are necessary to prevent issues that could affect the overall functionality of the system.

8.2. Output:



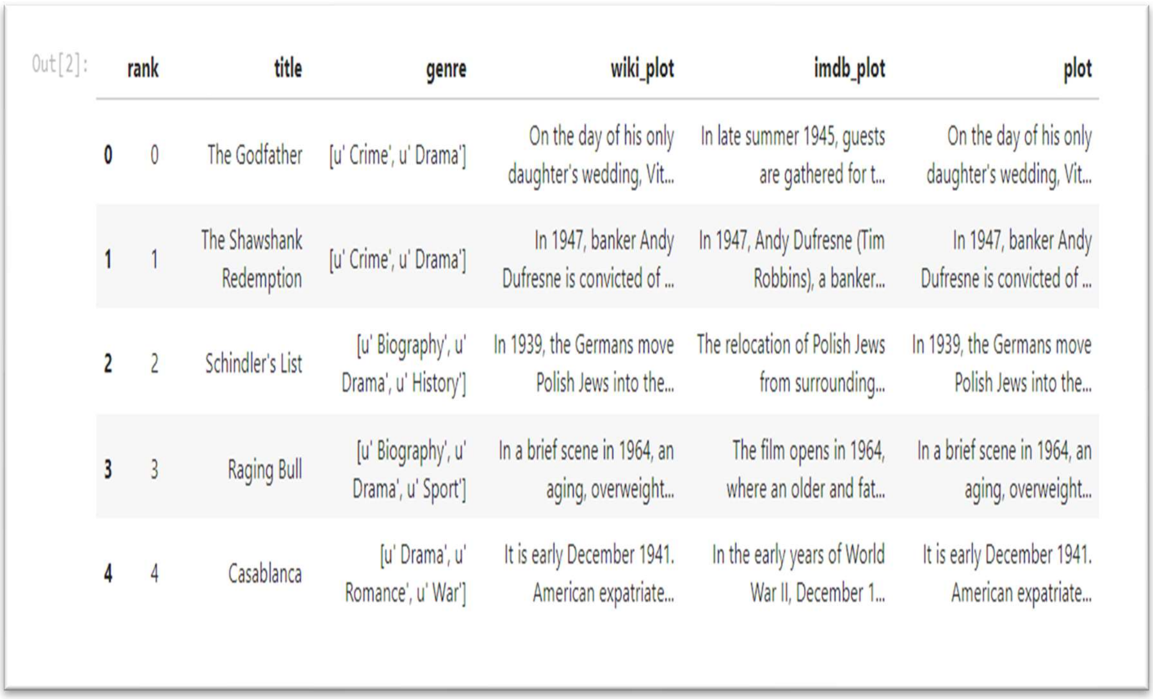
Out[1]:

	rank	title	genre	wiki_plot	imdb_plot
0	0	The Godfather	[u' Crime', u' Drama']	On the day of his only daughter's wedding, Vit...	In late summer 1945, guests are gathered for t...
1	1	The Shawshank Redemption	[u' Crime', u' Drama']	In 1947, banker Andy Dufresne is convicted of ...	In 1947, Andy Dufresne (Tim Robbins), a banker...
2	2	Schindler's List	[u' Biography', u' Drama', u' History']	In 1939, the Germans move Polish Jews into the...	The relocation of Polish Jews from surrounding...
3	3	Raging Bull	[u' Biography', u' Drama', u' Sport']	In a brief scene in 1964, an aging, overweight...	The film opens in 1964, where an older and fat...
4	4	Casablanca	[u' Drama', u' Romance', u' War']	It is early December 1941. American expatriate...	In the early years of World War II, December 1...
...	...	...	...	...	...
95	95	Rebel Without a Cause	[u' Drama']	\n\n\n\nJim Stark is in police custody.\n\n \...	Shortly after moving to Los Angeles with his p...
96	96	Rear Window	[u' Mystery', u' Thriller']	\n\n\n\nJames Stewart as L.B. Jefferies\n\n \...	L.B. "Jeff" Jeffries (James Stewart) recuperat...
97	97	The Third Man	[u' Film-Noir', u' Mystery', u' Thriller']	\n\n\n\nSocial network mapping all major chara...	Sights of Vienna, Austria, flash across the sc...
98	98	North by Northwest	[u' Mystery', u' Thriller']	Advertising executive Roger O. Thornhill is mi...	At the end of an ordinary work day, advertisin...
99	99	Yankee Doodle Dandy	[u' Biography', u' Drama', u' Musical']	\n In the early days of World War II, Cohan ...	NaN

100 rows × 5 columns

Figure 8.2.1: Import and observe dataset

:



```
Out[2]:
```

	rank	title	genre	wiki_plot	imdb_plot	plot
0	0	The Godfather	[u' Crime', u' Drama']	On the day of his only daughter's wedding, Vit...	In late summer 1945, guests are gathered for t...	On the day of his only daughter's wedding, Vit...
1	1	The Shawshank Redemption	[u' Crime', u' Drama']	In 1947, banker Andy Dufresne is convicted of ...	In 1947, Andy Dufresne (Tim Robbins), a banker...	In 1947, banker Andy Dufresne is convicted of ...
2	2	Schindler's List	[u' Biography', u' Drama', u' History']	In 1939, the Germans move Polish Jews into the...	The relocation of Polish Jews from surrounding...	In 1939, the Germans move Polish Jews into the...
3	3	Raging Bull	[u' Biography', u' Drama', u' Sport']	In a brief scene in 1964, an aging, overweight...	The film opens in 1964, where an older and fat...	In a brief scene in 1964, an aging, overweight...
4	4	Casablanca	[u' Drama', u' Romance', u' War']	It is early December 1941. American expatriate...	In the early years of World War II, December 1...	It is early December 1941. American expatriate...

Figure 8.2.2: Combine Wikipedia and IMDb plot summaries

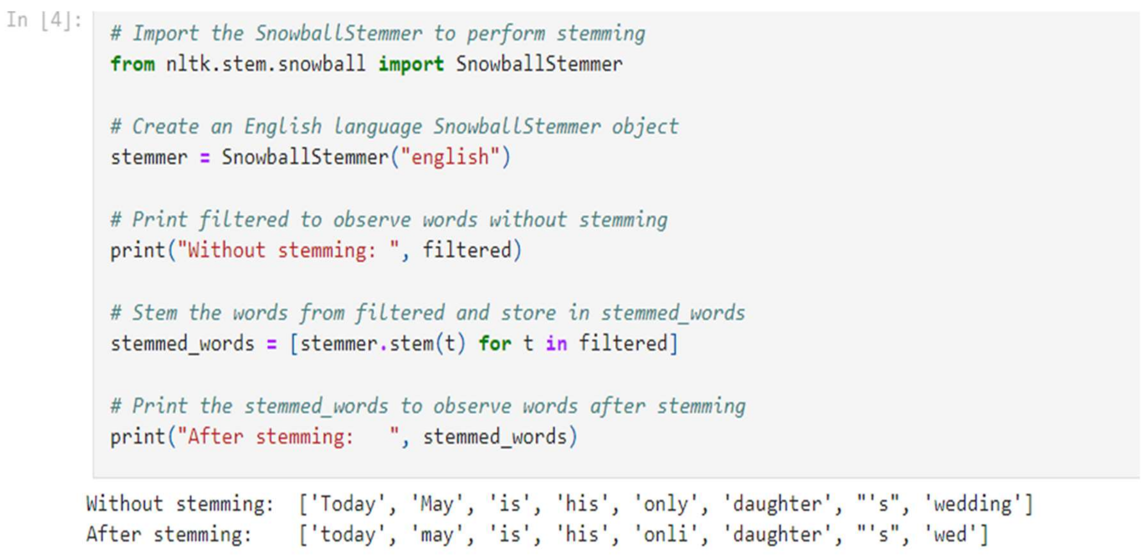
Figure 8.2.2: provides the dataset we imported currently contains two columns titled wiki_plot and imdb_plot. They are the plot found for the movies on Wikipedia and IMDb, respectively. The text in the two columns is similar, however, they are often written in different tones and thus provide context on a movie in a different manner of linguistic expression.

```
Out[3]:  ['Today', 'May', 'is', 'his', 'only', 'daughter', "'s", 'wedding']
```

Figure 8.2.3: Tokenization

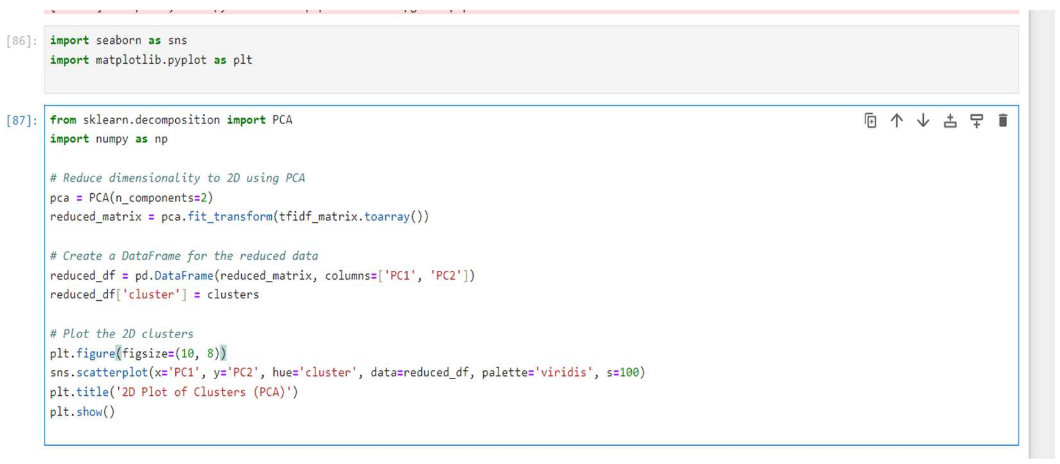
Tokenization is the process by which we break down articles into individual sentences or words, as needed. Besides the tokenization method provided by NLTK, we might have to perform additional filtration to remove tokens which are entirely numeric values or punctuation.

Tokenization on a small extract from *The Godfather* is provided above.


In [4]:

```python
# Import the SnowballStemmer to perform stemming
from nltk.stem.snowball import SnowballStemmer

# Create an English language SnowballStemmer object
stemmer = SnowballStemmer("english")

# Print filtered to observe words without stemming
print("Without stemming: ", filtered)

# Stem the words from filtered and store in stemmed_words
stemmed_words = [stemmer.stem(t) for t in filtered]

# Print the stemmed_words to observe words after stemming
print("After stemming:   ", stemmed_words)
```

```
Without stemming:  ['Today', 'May', 'is', 'his', 'only', 'daughter', "'s", 'wedding']
After stemming:    ['today', 'may', 'is', 'his', 'onli', 'daughter', "'s", 'wed']
```


[86]:

```python
import seaborn as sns
import matplotlib.pyplot as plt
```

[87]:

```python
from sklearn.decomposition import PCA
import numpy as np

# Reduce dimensionality to 2D using PCA
pca = PCA(n_components=2)
reduced_matrix = pca.fit_transform(tfidf_matrix.toarray())

# Create a DataFrame for the reduced data
reduced_df = pd.DataFrame(reduced_matrix, columns=['PC1', 'PC2'])
reduced_df['cluster'] = clusters

# Plot the 2D clusters
plt.figure(figsize=(10, 8))
sns.scatterplot(x='PC1', y='PC2', hue='cluster', data=reduced_df, palette='viridis', s=100)
plt.title('2D Plot of Clusters (PCA)')
plt.show()
```

Figure 8.2.4: 2D Plot of Clusters

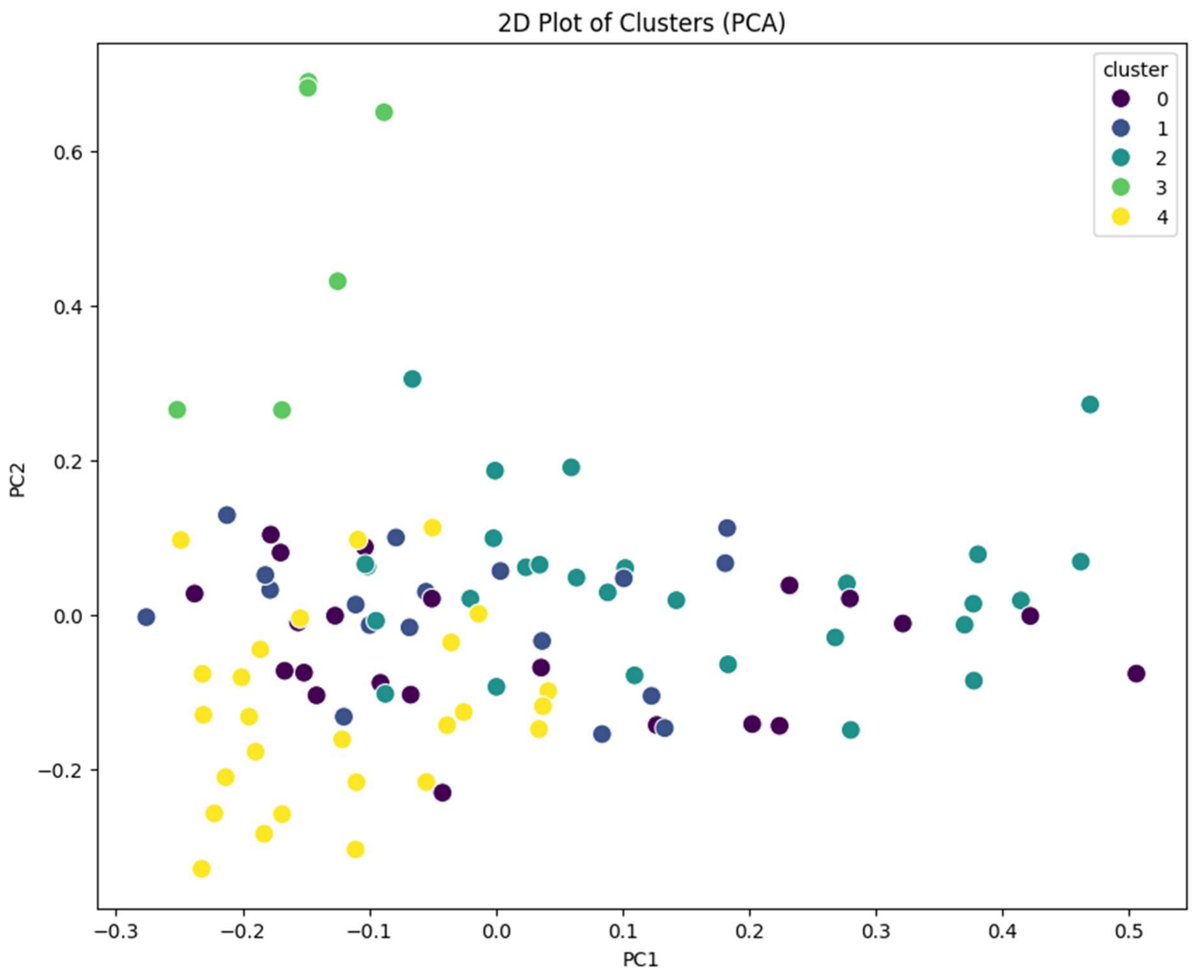
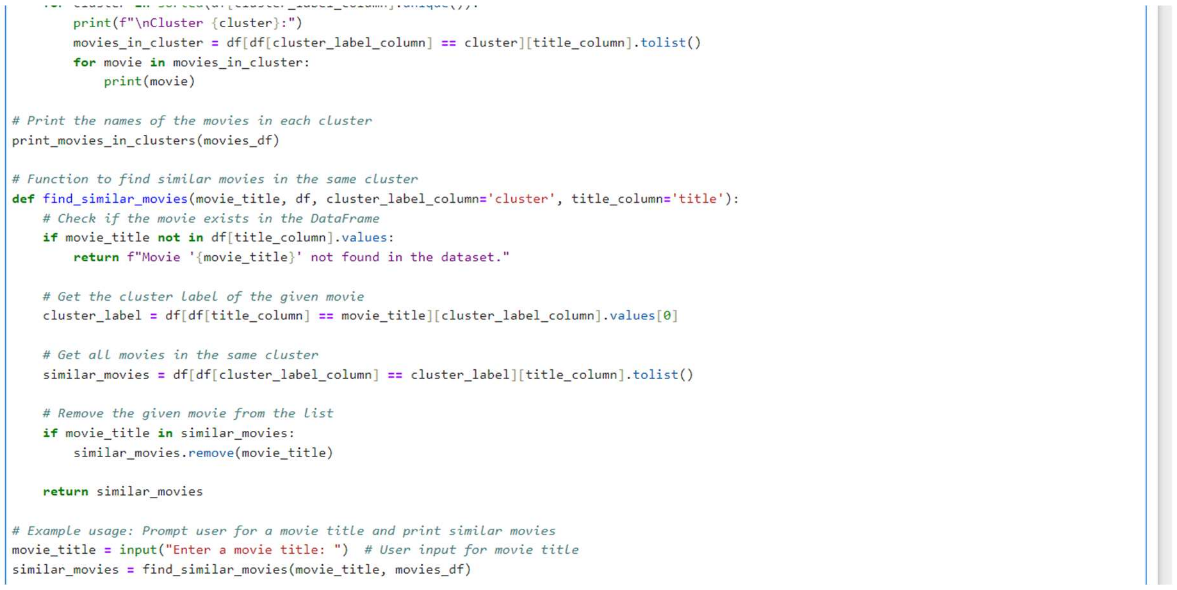
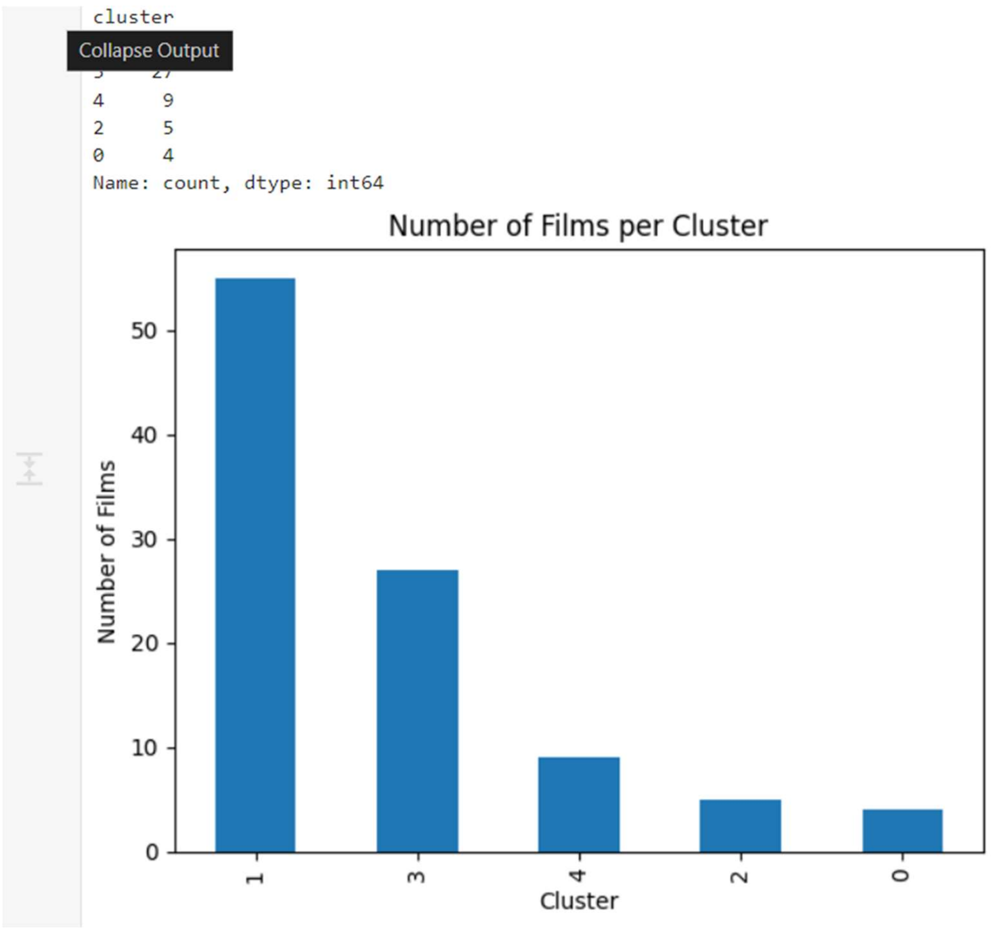


Figure 8.2.5: Plotted Graph

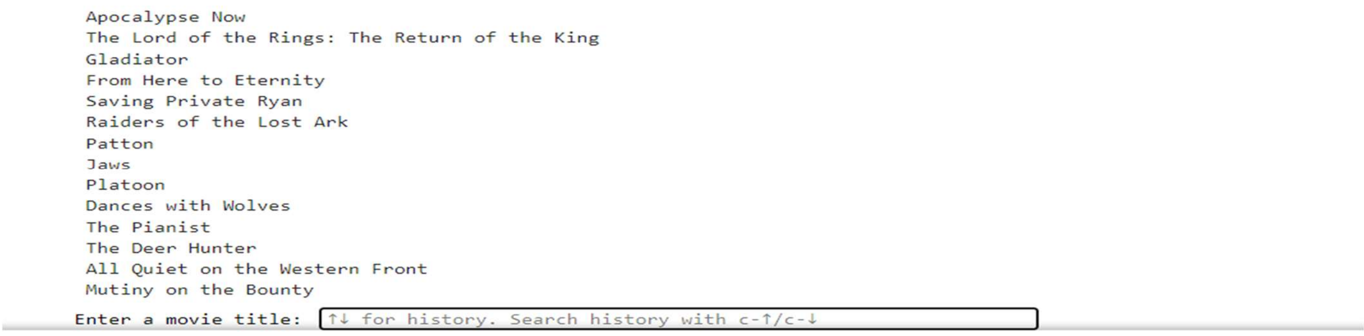


```
        print(f"\nCluster {cluster}:")
        movies_in_cluster = df[df[cluster_label_column] == cluster][title_column].tolist()
        for movie in movies_in_cluster:
            print(movie)

# Print the names of the movies in each cluster
print_movies_in_clusters(movies_df)

# Function to find similar movies in the same cluster
def find_similar_movies(movie_title, df, cluster_label_column='cluster', title_column='title'):
    # Check if the movie exists in the DataFrame
    if movie_title not in df[title_column].values:
        return f"Movie '{movie_title}' not found in the dataset."

    # Get the cluster label of the given movie
    cluster_label = df[df[title_column] == movie_title][cluster_label_column].values[0]

    # Get all movies in the same cluster
    similar_movies = df[df[cluster_label_column] == cluster_label][title_column].tolist()

    # Remove the given movie from the list
    if movie_title in similar_movies:
        similar_movies.remove(movie_title)

    return similar_movies

# Example usage: Prompt user for a movie title and print similar movies
movie_title = input("Enter a movie title: ")  # User input for movie title
similar_movies = find_similar_movies(movie_title, movies_df)
```

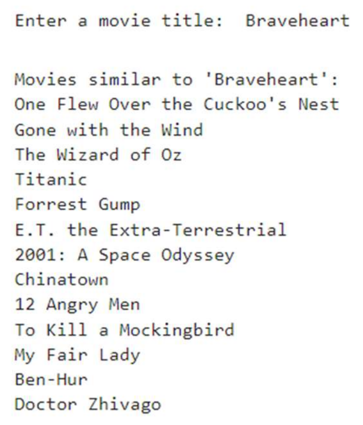


```
cluster
Collapse Output
3     27
4      9
2      5
0      4
Name: count, dtype: int64
```

Figure 8.2.6:Films per Cluster analysis



```
Apocalypse Now
The Lord of the Rings: The Return of the King
Gladiator
From Here to Eternity
Saving Private Ryan
Raiders of the Lost Ark
Patton
Jaws
Platoon
Dances with Wolves
The Pianist
The Deer Hunter
All Quiet on the Western Front
Mutiny on the Bounty
Enter a movie title: ↑↓ for history. Search history with c-↑/c-↓
```



```
Enter a movie title:  Braveheart

Movies similar to 'Braveheart':
One Flew Over the Cuckoo's Nest
Gone with the Wind
The Wizard of Oz
Titanic
Forrest Gump
E.T. the Extra-Terrestrial
2001: A Space Odyssey
Chinatown
12 Angry Men
To Kill a Mockingbird
My Fair Lady
Ben-Hur
Doctor Zhivago
```

Figure 8.2.7 Final Output

6. CONCLUSION

In summary, the project's purpose is to leverage K-means clustering to extract actionable insights from the OTT TV shows dataset, contributing to enhanced content delivery, improved user experiences, and strategic decision-making within the dynamic landscape of over-the-top streaming platforms. The project's findings are expected to contribute to an optimized content ecosystem, enhanced user satisfaction, and strategic decision-making within the dynamic and competitive OTT industry.This mini project successfully built a movie recommender system using K-Means clustering. We groupedmovies based on user ratings, assigned users to similar movie clusters, and recommended movies within their cluster, personalizing recommendations based on past preferences. While K-Means clustering effectively grouped movies and users, limitations include pre-defining cluster numbers and neglecting factors beyond ratings. Future work could involve incorporating additional movie features or user demographics, exploring different recommendation algorithms, and implementing evaluation metrics to further refine the system's accuracy and personalization.

7. FUTURE ENHANCEMENT

Future enhancements for a movie recommendation system using K-means clustering include integrating hybrid recommendation techniques like collaborative and content- based filtering, adopting advanced clustering algorithms such as Gaussian Mixture Models and hierarchical clustering, and improving data preprocessing through normalization and dimensionality reduction. Additionally, real-time recommendations can be achieved with streaming data processing and online learning algorithms. Enhancing user interaction through feedback loops and interactive recommendations, incorporating contextual and location-based information, and leveraging sentiment analysis from reviews and social media are key areas for improvement. Scalability can be optimized with distributed computing and cloud integration, while visualization tools and explainable recommendations increase transparency. Ensuring data privacy and security, providing multi-language and multi-region support, and integrating with external platforms like streaming services and social media will further enrich the system and user experience.

8. BIBLIOGRAPHY AND REFERENCES

[1]Sundaravel, E. & N., Elangovan. (2020). Emergence and future of Over-the-top (OTT) video services in India: analytical research. International Journal of Business Management and Social Research. 8. 489-499. 10.18801/ijbmsr.080220.50.

[2.]Jhala, Bhavyarajsinh Patadiya, Vivek 2021/09/13 120 125A STUDY ON CONSUMER BEHAVIOUR TOWARDS OTT PLATFORMS IN INDIA DURING COVID ERA

[3].Dwi Gustin Nurdialit, Data Scientist-Published in Analytics Vindhya 2/03/2012- Netflix Movies and TV Shows — Exploratory Data Analysis (EDA) and Visualization Using Python.

[4].Book by Achar Bhargav, VybhavChoi, SeongwooHaddad, David 2022/03/15 Titled Data Analysis on Netflix datasets

[5].Bristi, W.R., Zaman, Z. and Sultana, N., 2019, July. Predicting imdb rating of movies by machine learning techniques. In 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-5). IEEE.

[6].Fitzgerald, T., 2020. How Many Is Too Many? There Are Now More Than 500 TVShows.Available at https://www.forbes.com/sites/tonifitzgerald/2020/01/10/how-many-is-toomany-there-are-now-more-than-500-tv-shows/?sh=76c3bcbb550a.
Alghamdi, R. and Alfalqi, K., 2015. A survey of topic modeling in text mining. Int. J.Adv. Comput. Sci. Appl.(IJACSA), 6(1).

[7].Akula, R., Wieselthier, Z., Martin, L. and Garibay, I., 2019, April. Forecasting the Success of Television Series using Machine Learning. In 2019 Southeast Con (pp.1-8). IEEE.