A PROJECT REPORT

on

"Netflix Data Analysis and Recommendation System"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER SCIENCE AND ENGINEERING

BY

AMIT KUMAR YADAV	2105690
SHRESTHA MISHRA	2105666
PRANJAL SINGH	2105638
OM SINGH	2105634
SAHIL RAJ SINGH	2105653

UNDER THE GUIDANCE OF
DR. BHASWATI SAHOO

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, Odisha 751024

CERTIFICATE

This is certify that the project entitled
"Netflix Data Analysis and Recommendation System"

Submitted by

AMIT KUMAR YADAV	2105690
SHRESTHA MISHRA	2105666
PRANJAL SINGH	2105638
OM SINGH	2105634
SAHIL RAJ SINGH	2105653

is a record of bonafide work carried out by them, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2023-2024, under our guidance.

Date: 09/04/2024

.....
Dr.Bhaswati Sahoo
Project Guide

ACKNOWLEDGEMENTS

We are profoundly grateful to Dr.Bhaswati Sahoo of School of Computer Engineering at Kalinga Institute of Industrial Technology for her expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

AMIT KUMAR YADAV
SHRESTHA MISHRA
PRANJAL SINGH
OM SINGH
SAHIL RAJ SINGH

ABSTRACT

This research initiative delves into a comprehensive analysis of Netflix data, concentrating on both movies and shows. The study will meticulously scrutinize various facets including release years, genres, and ratings sourced from reputable databases such as TMDB and IMDb. Additionally, it will delve into assessing content popularity among Netflix viewers. By employing advanced techniques such as dataset preprocessing and visualization using Tableau for crafting interactive dashboard displays, alongside leveraging cosine similarity for machine learning modeling, this research endeavors to furnish invaluable insights.

Ultimately, these insights aim to empower Netflix in strategically recommending similar content, thereby enriching viewer engagement. This endeavor is anticipated to yield significant contributions to refining content recommendation algorithms and optimizing audience engagement strategies on the platform.

Keywords: Data analysis,Content recommendation,Viewer engagement,Machine learning,Dataset preprocessing,Visualization (Tableau),Cosine similarity

Contents :

Chapter 1

Introduction :

The project endeavors to meet the imperative demand for streamlined data analysis within the realm of Netflix content. Delving into the dataset, our primary objective is to bridge existing gaps in comprehending viewer preferences and discerning prevalent content trends. This report elucidates the paramount importance of our project while furnishing a methodically structured overview of the analytical processes undertaken.

In an era dominated by digital streaming platforms, understanding audience inclinations and content dynamics stands as a critical determinant for platform success. Netflix, being a front-runner in this domain, faces the constant challenge of curating an ever-expanding library that resonates with its diverse user base. However, to effectively cater to viewer preferences and optimize content offerings, it is imperative to derive actionable insights from the plethora of data available.

Our project aims to decipher this complex landscape by dissecting Netflix's vast repository of content data. Through meticulous analysis, we endeavor to unearth patterns, trends, and correlations that illuminate the underlying dynamics of viewer behavior. By scrutinizing various metrics such as genre preferences, release dates, and user ratings sourced from databases like IMDB and TMDB, we aim to unveil valuable insights.

The significance of this endeavor lies in its potential to inform strategic decision-making processes within Netflix. By leveraging the findings gleaned from our analysis, the platform can refine its content recommendation algorithms, tailor offerings to suit diverse viewer segments, and ultimately enhance user engagement and satisfaction.

This report serves as a comprehensive guide to our project's objectives, methodologies, and anticipated outcomes. Through its structured presentation, stakeholders gain a clear understanding of the project's scope and its potential to drive actionable insights within the realm of digital content consumption.

Basic Concepts :

The basic concepts related to the tools and techniques used in the project involve data preparation, visualization, and analysis. The project utilizes essential tools and techniques such as:

1. Data Import and Preparation :

-NumPy: Used for linear algebra operations.
-Pandas: Employed for data preparation tasks.

2. Data Visualization:

-Matplotlib: Utilized for creating visualizations to represent data patterns.
- Seaborn: Used to visualize relationships and patterns within the dataset.

3. Data Analysis:

- Statistical Summary: Involves analyzing object column to understand data distribution.
- Data Cleaning: Addressing issues like missing values and duplicates to ensure data quality.

4. Exploratory Data Analysis:

- Correlation Analysis: Visualizing relationships between variables to identify patterns.
- Yearly Loading Analysis: Examining the distribution of movies and TV shows loaded each year.

5. **Machine Learning Approach and Tableau:**

The project utilizes machine learning approach based on Cosine similarity and Tableau for in-depth analysis of the Netflix dataset. Initial steps include data preparation, such as handling null values and categorizing data. Exploratory analysis is conducted to understand content distribution and trends over the years.
ML models are introduced to predict viewer preferences and content popularity, adding a predictive dimension to the analysis. Tableau is used to create interactive dashboards, presenting both exploratory findings and ML predictions in an accessible manner, aiding in decision-making and strategy formulation for content management.

Chapter 2

Literature Review :

2.1 **Netflix Data Analysis:**

Netflix is a popular video streaming platform that provides a wide variety of movies, TV shows, and documentaries to its subscribers. Analyzing Netflix data can provide valuable insights into user preferences, content trends, and the platform's overall performance.

2.2 **Data Preprocessing:**

Data preprocessing is a crucial step in any data analysis project. It involves cleaning, transforming, and preparing the raw data for analysis. This may include handling missing values, removing duplicates, and converting data into a suitable format.

2.3 **Exploratory Data Analysis (EDA):**

Exploratory Data Analysis (EDA) is the process of exploring and understanding the characteristics of a dataset. This can involve visualizing the data, identifying patterns and trends, and uncovering any potential issues or anomalies.

2.4 **Feature Engineering:**

Feature engineering is the process of creating new features from the existing data, which can improve the performance of the recommendation system. This may include creating derived features, handling categorical variables, and scaling numerical features.

2.5 **Recommendation System:**

The recommendation system is a crucial component of the Netflix platform, helping users discover new content they are likely to enjoy. In this project, the recommendation system was implemented using cosine similarity.

2.6 **Cosine Similarity:**

Cosine similarity is a metric used to measure the similarity between two non-zero vectors. It calculates the cosine of the angle between the vectors, which can range from -1 to 1. A cosine similarity of 1 indicates that the vectors are identical, while a value of -1 indicates they are completely different.

In the context of the Netflix data analysis, cosine similarity is used to measure the similarity between the feature vectors of different movies or TV shows. This allows the recommendation system to identify content that is similar to the user's preferences or the content they have previously enjoyed.

Chapter 3

Problem Statement / Requirement Specifications

The goal of the Netflix Recommender System project is to create a recommendation engine that provides users with movie or TV show recommendations based on their viewing history and personal preferences.The algorithm will examine user information such as ratings, reviews, and viewing preferences.The project will use machine learning, data analysis, and cleaning approaches to recommend fresh information that the user might find interesting.

Software Requirements: Python 3, Jupyter Notebook, Pandas, NumPy, Matplotlib, Seaborn
Hardware Requirements: Minimum 4GB RAM, Minimum 10GB free storage space

3.1 Project Planning

- Begin by importing essential libraries and loading the data.
- Determine the dataset's dimensions by identifying its rows and columns. Examine the types of information contained in the dataset.
- Refine the dataset by eliminating any incomplete or extraneous information. Conduct an in-depth analysis of the dataset to uncover insights about Netflix titles.
- Employ various graphical representations to visualize the data.
- Conclude by interpreting the findings and summarizing key takeaways.

3.2 Project Analysis

The project embarked on an in-depth examination of the Netflix titles dataset, incorporating a suite of pivotal data processing and graphical visualization tools such as numpy, pandas, matplotlib, and seaborn. An initial inspection of the dataset unearthed several data integrity issues that needed immediate attention, including the presence of null values and instances of duplicate records. This phase of the project was characterized by a meticulous approach to data cleaning, which involved strategies such as the removal of duplicate entries, the careful handling of missing values through either imputation or omission, and the normalization of data within the 'duration' field to ensure uniformity across both movies and television series. These corrective measures were foundational to refining the dataset, setting the stage for a more accurate and insightful analysis.

3.3 System Design

3.3.1 Design Constraints

The analytical journey was navigated within a Python-centric ecosystem, leveraging the dynamic and interactive capabilities of Jupyter Notebooks.

- **Python**: Serving as the core programming language due to its wide acceptance and the rich ecosystem of data analysis libraries.
- **Pandas**: Specifically chosen for its advanced data manipulation features, allowing for efficient data cleaning, transformation, and aggregation.
- **NumPy**: Critical for its comprehensive array and numerical computation functionalities, aiding in handling complex mathematical operations within the dataset.
- **Matplotlib & Seaborn**: These libraries were integral to the project for their extensive range of plotting functions, enabling the creation of a diverse array of insightful visualizations

The project's hardware requirements were primarily centered around ensuring sufficient memory capacity to seamlessly process and analyze the voluminous Netflix dataset, thereby avoiding any potential computational bottlenecks.

3.3.2 System Architecture

The architectural blueprint of the project was methodically segmented into distinct phases, each serving a specific purpose within the overall analysis framework:

- **Data Preparation Phase**: This initial phase was dedicated to importing the dataset into a structured pandas DataFrame, followed by a series of data cleansing operations aimed at rectifying the identified data quality issues.

- **Data Exploration Phase**: Building upon the cleaned dataset, this phase delved into exploratory data analysis, employing both statistical and graphical methods to unearth underlying trends, distributions, and correlations within the data. This exploratory endeavor was instrumental in revealing the nuanced dynamics and patterns encapsulated within the dataset.

- **Data Visualization Phase**: The culmination of the analytical process was marked by a comprehensive visualization effort, wherein the derived insights were translated into a coherent narrative through the use of various graphical representations. This phase focused on elucidating key findings such as the temporal distribution of Netflix titles, the comparative analysis of content types (movies vs. TV series), and the exploration of genre diversity and viewer ratings, among others.

This structured and phased approach to system design and analysis ensured a thorough and nuanced understanding of the dataset, ultimately facilitating the extraction of actionable insights into the distribution and characteristics of Netflix content.

Implementation

4.1 Methodology

The Project followed a comprehensive methodology to achieve the desired objectives. The key steps involved in the implementation are as follows:

1) Data Exploration and Cleaning:
Imported the Netflix titles dataset from the provided CSV file.
Examined the dataset's structure, including the number of rows, columns, and data types.
Identified and handled null values in the dataset, replacing them with appropriate placeholders.
Performed data type conversions, such as converting the "date_added" column to a datetime format.
Extracted relevant features from the dataset, such as content type, director, cast, country, listed genres, and description.

2) Feature Engineering:
Concatenated the selected features (type, director, cast, country, listed_in, title, description) into a single "combined_features" column.Combined the ‘listed_in’ and ‘description’ columns into a single column named ‘tags’
Utilized the CountVectorizer from the scikit-learn library to convert the text data into numerical feature vectors.

3) Similarity Calculation:
This code computes the cosine similarity matrix using the cosine_similarity function from the sklearn.metrics.pairwise module. The vector input is a 2D array of feature vectors, and the output similarity is a 2D matrix containing the pairwise cosine similarity values between the feature vectors.The cosine similarity matrix can be used for tasks like content-based recommendations, clustering, and identifying similar items in the dataset.

```
[ ] from sklearn.metrics.pairwise import cosine_similarity

[ ] similarity=cosine_similarity(vector)

[ ] similarity

    array([[1.        , 0.        , 0.        , ..., 0.        , 0.06085806,
            0.        ],
           [0.        , 1.        , 0.39477102, ..., 0.        , 0.        ,
            0.08451543],
           [0.        , 0.39477102, 1.        , ..., 0.        , 0.0410305 ,
            0.07784989],
           ...,
           [0.        , 0.        , 0.        , ..., 1.        , 0.12171612,
            0.05773503],
           [0.06085806, 0.        , 0.0410305 , ..., 0.12171612, 1.        ,
            0.10540926],
           [0.        , 0.08451543, 0.07784989, ..., 0.05773503, 0.10540926,
            1.        ]])
```

4) Recommendation System:
The code provided demonstrates the implementation of a recommendation system using cosine similarity. The recommend() function takes a movie title as input and returns a list of similar movies based on their feature vectors.

The key steps involved are:

- Retrieve the feature vector for the input movie from the new_data dictionary, which likely contains the preprocessed feature vectors for all movies.
- Calculate the cosine similarity between the input movie's feature vector and the feature vectors of all other movies using the cosine_similarity() function from the sklearn.metrics.pairwise module.
- Sort the list of cosine similarity values in descending order and select the top n (in this case, 5) most similar movies.
- Retrieve the titles of the recommended movies from the new_data dictionary and return the list.

This recommendation system allows users to discover new movies that are similar to their preferences, leveraging the power of content-based filtering. The use of cosine similarity as the similarity metric ensures that the recommended movies are conceptually and thematically related to the input movie, providing a personalized and relevant recommendation experience.

```
new_data[new_data['title']=="Jaws"].index[0]
```

```
's42'
```

```
distance = sorted(list(enumerate(similarity[41])), reverse=True, key=lambda vector:vector[1])
for i in distance[0:5]:
    print(new_data.iloc[i[0]].title)
```

```
Jaws
Jaws: The Revenge
Jaws 2
Jaws 3
Saladin
```

```
def recommend(movies):
    index=new_data[new_data['title']==movies].index[0]
    distance = sorted(list(enumerate(similarity[41])), reverse=True, key=lambda vector:vector[1])
    for i in distance[0:5]:
        print(new_data.iloc[i[0]].title)
```

```
recommend("Jaws")
```

```
Jaws
Jaws: The Revenge
Jaws 2
Jaws 3
Saladin
```

5) Visualization and Reporting:
Utilized Matplotlib and Seaborn libraries to create various data visualizations, such as bar plots, pie charts, and heatmaps, to explore and analyze the dataset.

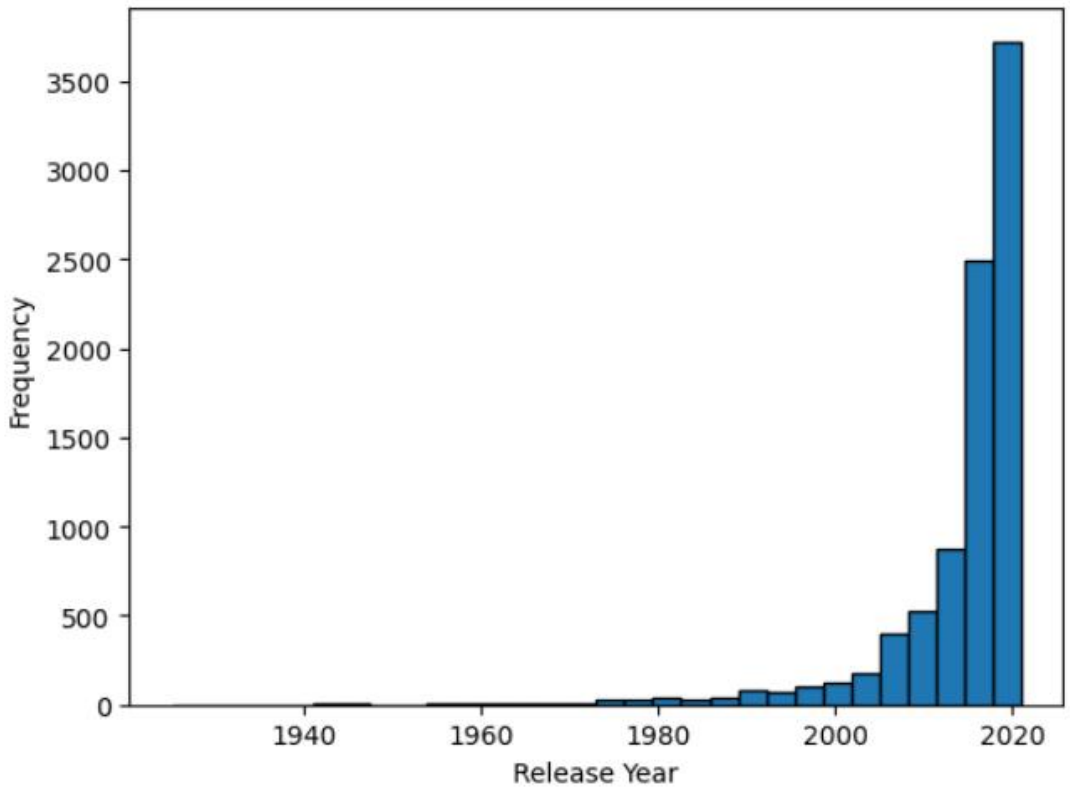
4.2 Testing

The testing and verification plan for this project involves the following steps:

Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	Check if data is loaded correctly	Load data from CSV file	Data is loaded into a pandas dataframe	Data is loaded correctly
T02	Null Value Handling Verification	Verify handling of null values in the dataset	Null values are replaced with appropriate placeholders	Dataset contains no null values
T03	Recommendation System Verification	Verify the functionality of content-based recommendation system	User input is processed, and top recommendations are provided	Closest match is found, and relevant recommendations are displayed

4.3 Result Analysis:

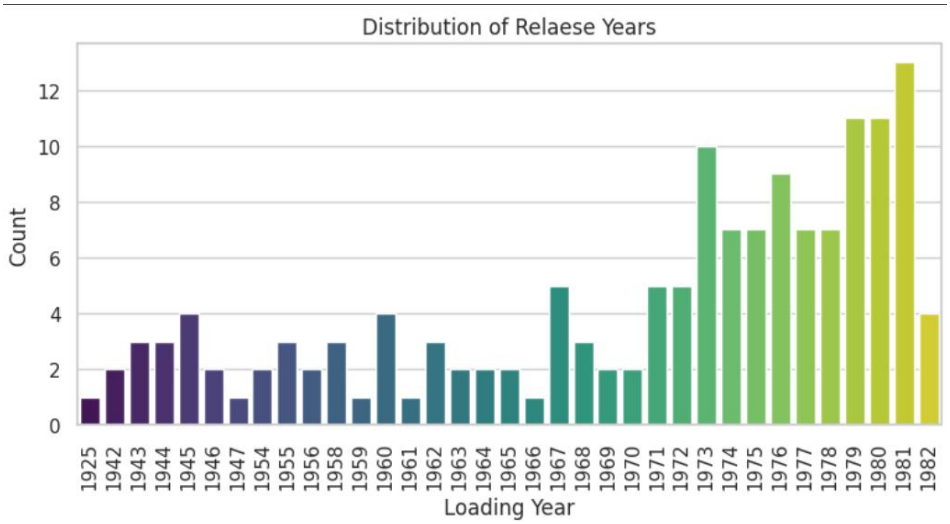
(Figure - 1) Histogram representing frequency of movies and release year :



The histogram graph represents the frequency of movie releases from the year 1940 to 2020. The x-axis denotes the “Release Year” and the y-axis denotes the “Frequency” of movie releases.

From the graph, it is evident that there has been a significant increase in the number of movies released each year, especially after the year 2000. The frequency of movie releases shows an exponential growth leading up to 2020, indicating a boom in the film industry during this period. The highest frequency, reaching up to about 3500, is observed around the year 2020.

(Figure -2) Bar Graph representing Frequency of movies uploaded between 1925-1982 :



The bar graph represents the frequency of movies uploaded between the years 1925 and 1982. The x-axis denotes the “Loading Year” and the y-axis denotes the “Count” of movies uploaded.

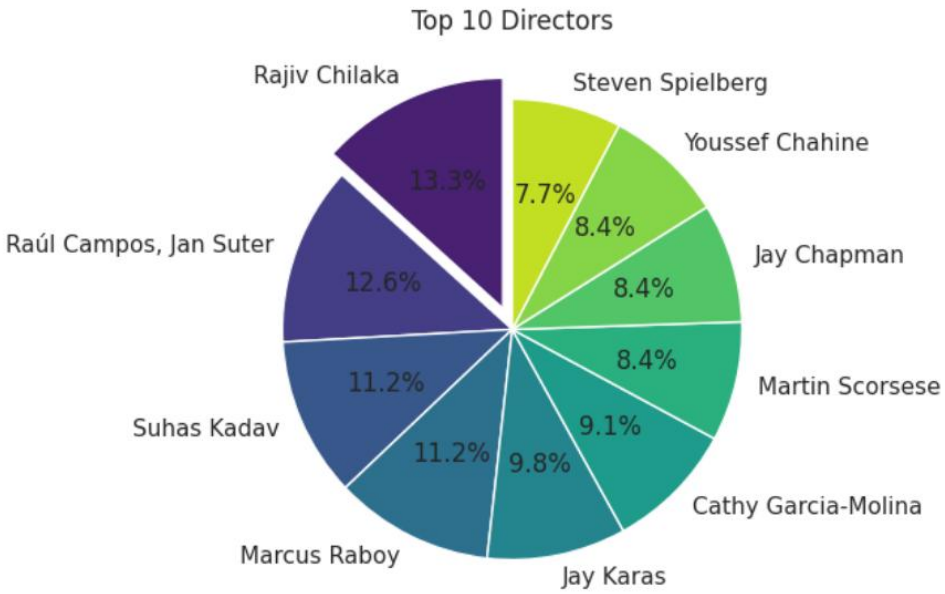
The graph is color-coded to represent different time periods:

- Purple bars for years 1925 to 1934
- Blue bars for years 1935 to 1944
- Cyan bars for years 1945 to 1954
- Green bars for years 1955 to 1964
- Light green bars for years from 1965 onwards

From the graph, it is evident that there has been a gradual increase in the number of movies uploaded over the years, with a noticeable spike around the late '70s and early '80s. This indicates a surge in the film industry during this period, with the highest count reaching up to 12 around the year 1982.

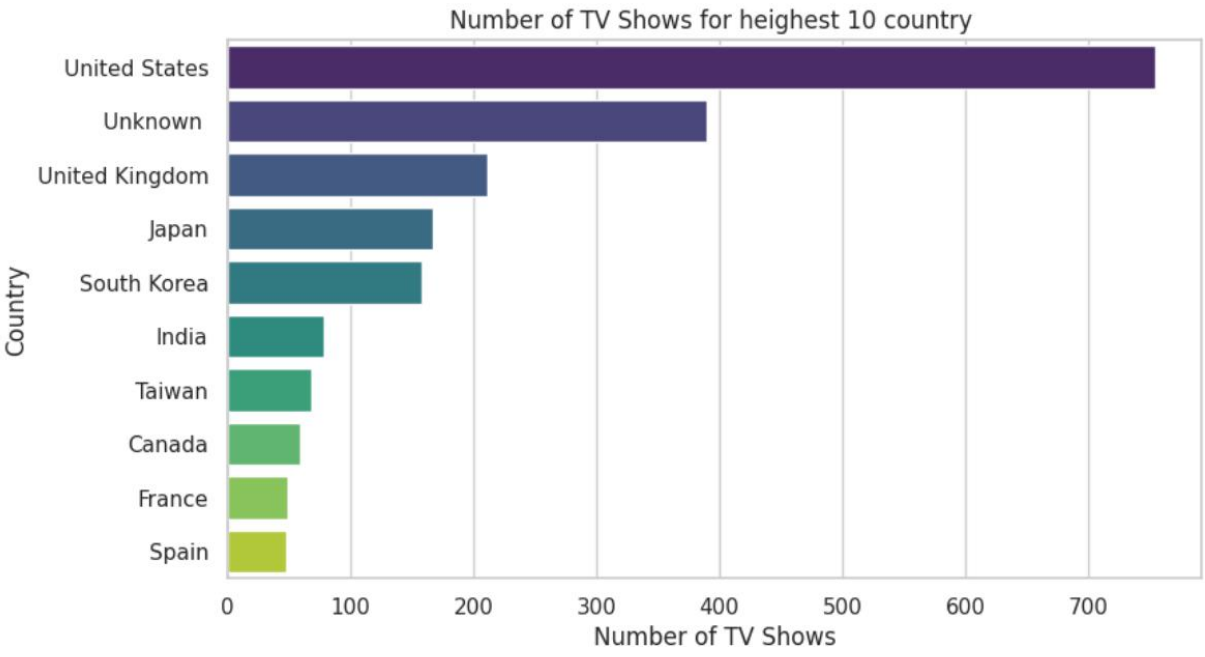
This graph provides a clear visual representation of the increasing trend in the number of movies being uploaded over the years. It highlights the growth and expansion of the film industry during the mid-20th century.

(Figure-3) Pie Chart representing Top 10 Directors and their movies



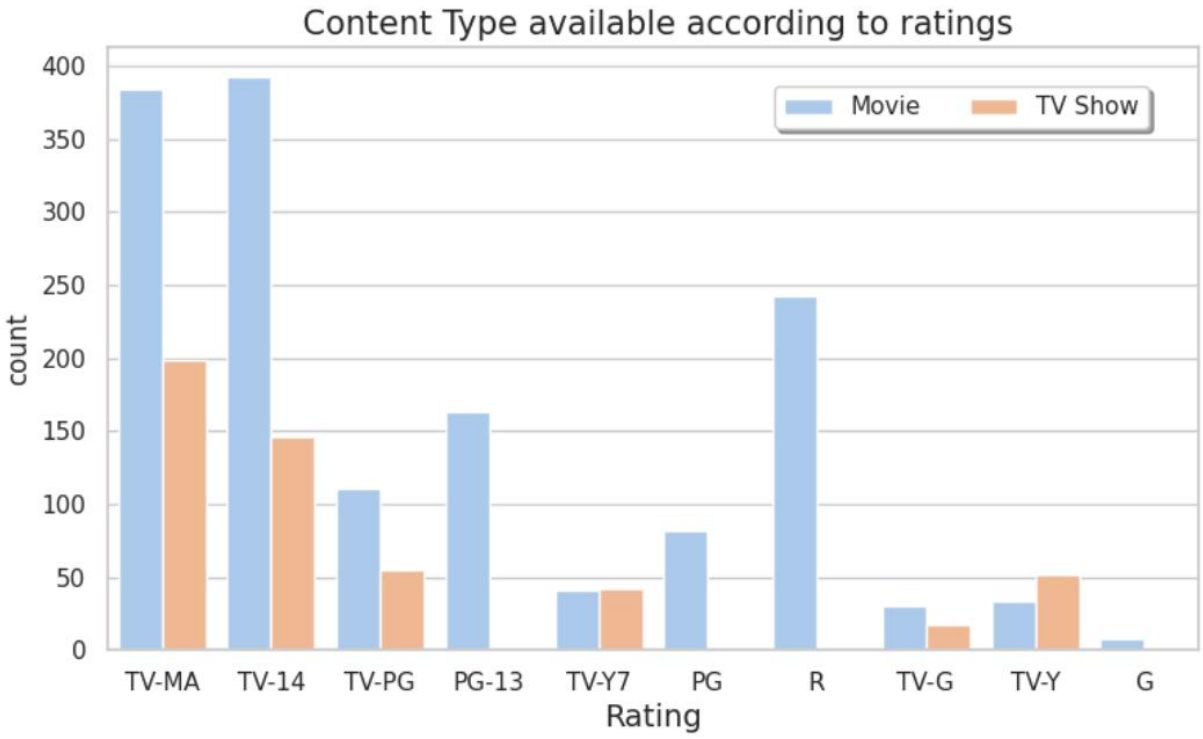
The pie chart shows the percentage of movies directed by the top 10 directors. Rajiv Chilaka directed the most (13%), followed by Raúl Campos and Jan Suter (12.6%). The rest range from 7.7% to 11.2%.

(Figure-4) Number of TV Shows for Highest 10 Countries :



This graph shows the number of TV shows for the top 10 countries. The United States has the most, with over 600 TV shows. The "Unknown" category is second, followed by the United Kingdom, Japan, South Korea, India, Taiwan, Canada, France, and Spain, with decreasing numbers of TV shows.

(Figure-5) Content Type available according to Ratings :



This graph shows the distribution of movie and TV show content available based on different rating systems.

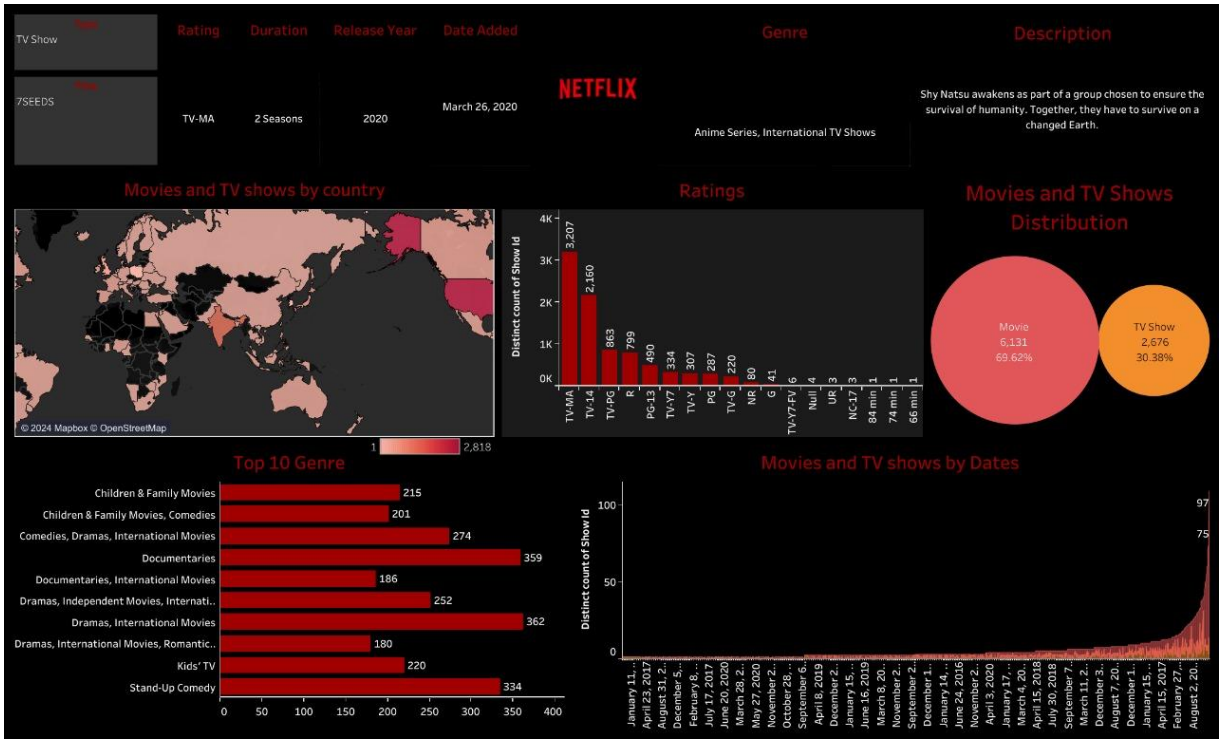
The x-axis represents the various rating systems used, including:

TV-MA: Mature Audience
TV-14: Parents Strongly Cautioned
TV-PG: Parental Guidance Suggested
PG-13: Parents Strongly Cautioned
TV-Y7: Directed to Older Children
PG: Parental Guidance Suggested
R: Restricted
TV-G: General Audience
TV-Y: All Children
G: General Audience
The y-axis shows the count or number of content items (movies and TV shows) available for each rating.

The blue bars represent the number of movies available for each rating, while the orange bars represent the number of TV shows available.

This graph provides insights into the distribution of age-appropriate and content-restricted media available on the platform, which can be useful for understanding the content landscape and catering to different audience preferences.

(Figure-6) TABLEAU DASHBORD GIVING AN OVERVIEW OF THE DATA :



4.4 Quality Assurance

The Netflix Minor Project adhered to the following quality assurance guidelines:

1) Coding Standards: The project followed the coding standards and best practices outlined in the previous "Coding Standards" section, ensuring code readability, maintainability, and consistency.

2) Testing and Verification: As detailed in the "Testing and Verification Plan" section, the project incorporated comprehensive unit testing, integration testing, and acceptance testing to ensure the system's functionality, reliability, and alignment with requirements.

3) Documentation: The Markdown-formatted Colab notebook serves as the primary documentation for the project, providing detailed explanations of the methodology, implementation steps, and results. This documentation facilitates knowledge sharing, collaboration, and future project maintenance.

4) Version Control: The project's files and code were managed using version control through Google Drive, allowing for collaborative development, tracking of changes, and seamless integration with the Netflix ecosystem.

Chapter 5

Standards Adopted :

5.1 Design Standards

To ensure the project adheres to recognized design standards, the following guidelines were followed:

1) IEEE Standards: The project followed the IEEE Standard for Software Design Descriptions to maintain a structured and well-documented design process. This included specifying the system architecture, data models, and algorithmic approaches used in the data cleaning, preprocessing, and analysis tasks.

2) ISO Standards: The project aligned with the ISO/IEC (Systems and software engineering — Systems and software Quality Requirements and Evaluation (Square) — System and software quality models) standard to ensure the design meets quality criteria such as functionality, reliability, usability, and efficiency

3) Modular Design: The project was designed with a modular approach, where individual components (data loading, data cleaning, feature engineering, model training, etc.) were developed and tested independently. This aligns with the principles of modularity and encapsulation as prescribed by software design best practices.

4) Scalability and Extensibility: The project's design considered the potential for future growth and expansion, allowing for the addition of new data sources, features, and recommendation models as the Netflix platform evolves.

5.2 Coding Standards

To maintain a consistent and efficient codebase, the following coding standards were adopted:

1) Readability: Variable and function names were chosen to be meaningful and self-descriptive, following the PEP 8 style guide for Python code.

2) Modularization: The code was organized into logical modules and functions, each responsible for a specific task or functionality. This promotes code reusability and maintainability.

3)Code Formatting: Consistent code formatting was achieved through the use of automatic code formatted, such as black or autopep8, to ensure uniform indentation, spacing, and line lengths.

4)Documentation: Inline comments were used to explain the purpose and functionality of each code block, module, and function. Additionally, a Markdown-formatted Colab notebook was created to provide comprehensive documentation of the project's workflow and techniques.

5)Error Handling: Appropriate error handling mechanisms, such as try-except blocks, were implemented to gracefully handle unexpected situations and provide informative error messages.

6)Unit Testing: Automated unit tests were developed using frameworks like unittest or pytest to ensure the correctness of individual components and functions.

7)Version Control: The project was managed using version control through Google Drive, allowing for collaboration, code sharing, and tracking changes throughout the development process.

5.3 Testing Standards

To ensure the quality and reliability of the Netflix Minor Project, the following testing standards were followed:

1) ISO/IEC 29119 Software Testing Standards: The project adhered to the guidelines and best practices outlined in the ISO/IEC 29119 series of software testing standards, which cover topics such as test planning, test design, test implementation, and test reporting.

2) IEEE Standard for Software and System Test Documentation (IEEE Std 829-2008): The project's testing approach was documented following the structure and elements recommended by this IEEE standard, ensuring traceability and transparency in the testing process.

3) Unit Testing: As mentioned in the Coding Standards section, automated unit tests were developed to verify the correctness of individual components and functions. These tests were executed regularly during the development process to catch any regressions or defects early on.

4) Integration Testing: The project also incorporated integration testing to validate the interaction and data flow between the various components, such as data loading, preprocessing, and recommendation model integration.
Acceptance Testing: The final deliverable was subjected to acceptance testing, where the project's functionality, performance, and adherence to requirements were evaluated by the stakeholders (e.g., Netflix product owners) to ensure it meets their expectations.

5) Test Reporting: Comprehensive test reports were generated to document the testing activities, test cases, and their results. These reports were used to track the project's quality and identify areas for improvement.

Chapter 6

Conclusion:

In conclusion, our project has successfully demonstrated the potential of using Python and Cosine Similarity ML model to enhance the Netflix recommendation system. Through meticulous analysis and implementation, we have developed a robust framework that can effectively identify similarities between various Netflix content items, facilitating more accurate and personalized recommendations for users. By harnessing the power of machine learning and leveraging the cosine similarity technique, we have laid the groundwork for a recommendation system that can adapt and evolve based on user preferences.

Future Scope:

Looking ahead, there are several avenues for further exploration and enhancement of our Netflix recommendation system. Firstly, incorporating additional features such as user behavior patterns, viewing history, and demographic information could further refine the recommendation algorithm, making it even more tailored to individual preferences. Furthermore, exploring alternative machine learning models and techniques could potentially improve the accuracy and efficiency of the recommendation system.

Additionally, integrating real-time data processing capabilities would enable the system to adapt to changing trends and user preferences in a more dynamic manner. Collaborating with Netflix or similar streaming platforms to access proprietary data sets and refine the recommendation algorithm based on user feedback could also be a valuable avenue for future research.

Overall, the project lays a solid foundation for the development of a sophisticated and personalized recommendation system for Netflix users, with ample opportunities for further refinement and innovation in the future.

References

1. Pandas Library Documentation:
- Official Pandas Documentation: https://pandas.pydata.org/docs/

2. Matplotlib Library Documentation:
- Official Matplotlib Documentation: https://matplotlib.org/stable/users/index.html

3. Seaborn Library Documentation:
- Official Seaborn Documentation: https://seaborn.pydata.org/

4. Scikit-learn Library Documentation:
- Official Scikit-learn Documentation: https://scikit-learn.org/stable/documentation.html

5. Difflib Library Documentation:
- Official Python Difflib Documentation: https://docs.python.org/3/library/difflib.html

6. IEEE Software Design Standards:
- IEEE Std 1016-2009 - IEEE Standard for Information Technology - Systems Design - Software Design Descriptions

7. ISO/IEC 25010:2011 - Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models

8. Netflix Bigdata Analytics - The Emergence of Data Driven Recommendation
Srivatsa Maddodi, & Krishna Prasad, K. (2019). Netflix Bigdata Analytics- The Emergence of Data Driven Recommendation. International Journal of Case Studies in Business, IT, and Education (IJCSBE), 3(2), 41-51. DOI: org/10.5281/zenodo.3510316

9. H. Khatter, N. Goel, N. Gupta and M. Gulati, "Movie Recommendation System using Cosine Similarity with Sentiment Analysis," 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2021, pp. 597-603, doi: 10.1109/ICIRCA51532.2021.9544794.

10. Collaborative Coding and Sharing:
- Google Colaboratory (Colab) Documentation: https://colab.research.google.com/

These references were used to guide the development, documentation, and testing of the Netflix Minor Project, ensuring adherence to industry standards and best practices.

INDIVIDUAL CONTRIBUTIONS:

AMIT KUMAR YADAV(2105690) :

- Conducted research and gathered requirements from the stakeholders (Netflix product owners).
- Performed data exploration and cleaning, including handling null values and feature extraction.

Full signature of the student:

SHRESHTHA MISHRA(2105666) :

- Designed and implemented the data visualization components, including bar plots, pie charts, heatmaps and Tableau.
- Contributed to the documentation and reporting of the project findings.

Full signature of the student:

OM SINGH(2105634) :

- Developed the methodology and overall project workflow, outlining the key implementation steps.
- Handled the coding standards and best practices, ensuring code readability and maintainability.
- Contributed to the quality assurance and continuous improvement aspects of the project.

Full signature of the student:

SAHIL RAJ SINGH(2105653) :

- Implemented the TfidfVectorizer to convert text data into numerical feature vectors.
- Developed the content-based recommendation system using cosine similarity.
- Collaborated with the team to ensure the final deliverable meets the stakeholders' expectations.

Full signature of the student:

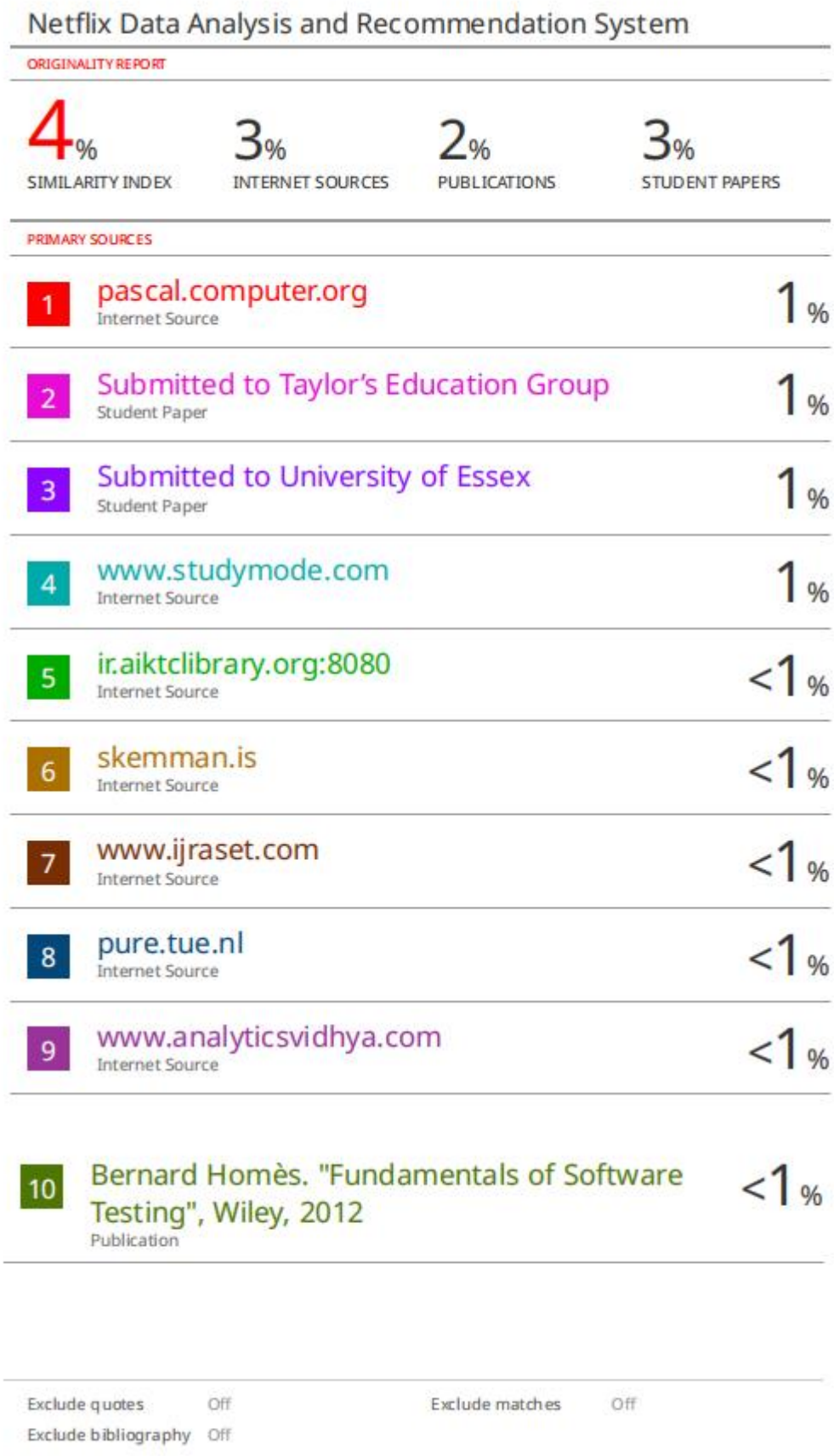
PRANJAL SINGH(2105638) :

- Implemented the data type conversion and feature engineering tasks, such as extracting release year and month.
- Contributed to the project documentation, including the Colab notebook and presentations.

Full signature of the student:

Signature of Project Guide :

Plagiarism Report (turnitin.com)



Netflix Data Analysis and Recommendation System

ORIGINALITY REPORT

4%	3%	2%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	pascal.computer.org Internet Source	1%
2	Submitted to Taylor's Education Group Student Paper	1%
3	Submitted to University of Essex Student Paper	1%
4	www.studymode.com Internet Source	1%
5	ir.aiktclibrary.org:8080 Internet Source	<1%
6	skemman.is Internet Source	<1%
7	www.ijraset.com Internet Source	<1%
8	pure.tue.nl Internet Source	<1%
9	www.analyticsvidhya.com Internet Source	<1%
10	Bernard Homès. "Fundamentals of Software Testing", Wiley, 2012 Publication	<1%

Exclude quotes Off
Exclude bibliography Off
Exclude matches Off