**ACKNOWLEDGEMENT**

First and foremost, I thank God Almighty for his divine grace and blessings in making all this possible. May he continue to lead me in the years to come. No words can express my humble gratitude to my beloved parents who have been guiding me in all walks of my journey.

I am also grateful to Prof. --------, Head of Computer Applications Department, Prof. -------, Project Coordinator for their valuable guidance and constant supervision as well as for providing necessary information regarding the project & also for their support.

I would like to express my special gratitude and thanks to my Project Guide Prof. ------, Assistant Professor, Department of Computer Applications for giving me such attention and time.

I profusely thank other Professors in the department and all other staffs of MBITS, for their guidance and inspirations throughout my course of study. My thanks and appreciations also goes my friends and people who have willingly helped me out with their abilities.

**ABSTRACT**

The exponential growth of online social media platforms has brought about new forms of interpersonal communication but has also given rise to harmful behaviors like cyberbullying. Cyberbullying, characterized by repetitive hostile and abusive language, can cause significant psychological and emotional harm to individuals, especially among adolescents and vulnerable groups. In response to this growing issue, this research proposes a robust cyberbullying detection system that leverages advanced machine learning techniques and natural language processing (NLP) to identify abusive content in real-time. By analyzing textual data from social media platforms, we aim to detect instances of bullying before they escalate, thereby contributing to safer online environments.

The system we developed employs several state-of-the-art classifiers, including support vector machines (SVM), random forests, and deep learning models like convolutional neural networks (CNN) and recurrent neural networks (RNN). These models are trained on large annotated datasets that contain various examples of cyberbullying behaviors, allowing the system to differentiate between harmful and benign interactions. To further enhance detection accuracy, we incorporate features such as sentiment analysis, user interaction patterns, and word embeddings like Word2Vec and GloVe, which provide richer semantic representations of the text. This multi-faceted approach enables the system to capture nuanced instances of cyberbullying that may otherwise go unnoticed by traditional keyword-based detection methods.

Our evaluation results show that the proposed system achieves high precision and recall in identifying cyberbullying across diverse datasets and platforms, making it a valuable tool for social media moderation and intervention efforts. By providing real-time alerts and flagging abusive content, this system has the potential to reduce the prevalence of cyberbullying and contribute to healthier online interactions. This work underscores the importance of combining technological innovation with ethical considerations to tackle one of the most pressing challenges

### LIST OF FIGURES

* 1. Snapshot of ratings.csv 7
  2. Snapshot of movies.csv 8
  3. Snapshot of users.csv 8
  4. Number of Movies, Users and Ratings 9
  5. Data type of Each Attribute 9
  6. Dropped attribute timestamp 10
  7. Merged movie title 10
  8. Handling missing values and checking duplicates 11
  9. Number of ratings per user 11
  10. Number of ratings per movie 12
  11. Occupation distribution of users 15
  12. Ratings distribution 16
  13. Pie chart for movie genre count 16
  14. Boxplot for ratings 17
  15. Example figure of kNN 19
  16. Activity Diagram 21
  17. Project Pipeline 22
  18. User feature dataframe 31
  19. Model knn1 31
  20. Model knn2 32
  21. Code snippet for user based recommendation 32
  22. Recommendation for user 23… 33
  23. Code snippet for item based recommendation 33
  24. Recommendation for the movie ‘Speed’ 34
  25. User movie input 35
  26. Item based recommendation 35
  27. User ID input 35
  28. User based recommendation 36
  29. Index page of movie recommender 38
  30. Input page of movie recommender 38
  31. User id input 39
  32. Similar users in recommendation for user 23 39
  33. Recommendation for user 12 40
  34. Movie title input 40
  35. Recommendation for the movie’12 Angry Men’ 41
  36. Validation for user id 41
  37. Validation for movie title 42
  38. Git history of 3 sprints 43
  39. Git history of sprint 3 43

**CONTENTS**

##### Introduction 1

##### Supporting Literature 2

* 1. Literature Review 2
     1. Summary Table 5
  2. Findings and Proposals 6

##### System Analysis 7

* 1. Analysis of Dataset 7
     1. About the Dataset 7
     2. Explore the dataset 9
  2. Data Pre-processing 10
     1. Data Cleaning 10
     2. Analysis of Feature Variables 13
     3. Analysis of Class Variables 15
  3. Data Visualization 15
  4. Analysis of Algorithm 18
     1. Activity Diagram 21
  5. Project Plan 22
     1. Project Pipeline 22
  6. Feasibility Analysis 24
     1. Technical Feasibility 24
     2. Economic Feasibility 25
     3. Operational Feasibility 26
  7. System Environment 27
     1. Software Environment 27
     2. Hardware Environment 29

##### System Design 30

* 1. Model Building 30
     1. [Model Planning 30](#_TOC_250011)
     2. [Training 34](#_TOC_250010)
     3. [Testing 34](#_TOC_250009)

1. [Results and Discussion 35](#_TOC_250008)
2. [Model Deployment 37](#_TOC_250007)
3. [Git History 43](#_TOC_250006)
4. [Conclusion 44](#_TOC_250005)
5. [Future Work 45](#_TOC_250004)
6. [Appendix 47](#_TOC_250003)
   1. [Minimum Software Requirements 47](#_TOC_250002)
   2. [Minimum Hardware Requirements 49](#_TOC_250001)
7. [References 50](#_TOC_250000)
8. **INTRODUCTION**

In today's digital landscape, the vast amount of user-generated content on platforms like social media and forums has brought increased attention to the critical role of cyberbullying detection systems. These systems serve as intelligent filters, employing advanced algorithms to detect and flag potentially harmful or abusive comments. This functionality is especially relevant in online spaces where diverse voices and perspectives converge, underscoring the need for proactive and adaptive content moderation.

In the face of information overload and the challenge of managing vast amounts of user interactions, cyberbullying detection systems act as essential tools for maintaining a safe and welcoming environment. By identifying content that could be harmful, these systems contribute to a healthier and more supportive digital community, where users can engage without fear of encountering abusive or offensive language.

The primary goal of cyberbullying detection is to enhance user safety and satisfaction. By analyzing comment histories and language patterns, these systems aim to identify patterns of abuse, harassment, or negativity, allowing for a safer and more respectful user experience. In this context, detecting abusive comments based on linguistic features and user behavior supports more meaningful and positive interactions between users.

In this project, we focus on developing a **cyberbullying comments detector** using **logistic regression** as a classification technique. By leveraging this supervised machine learning method, we aim to predict whether a given comment is abusive or non-abusive, enriching the toolkit for online content moderation and fostering a safer digital environment for all users. Through this exploration, we contribute to the ongoing evolution of AI-driven moderation tools, specifically within the domain of cyberbullying detection, with the ultimate objective of creating a more respectful and inclusive digital space.

1. **SUPPORTING LITERATURE**
   1. **Literature Review**
2. Paper 1: Cyberbullying Analysis in Intercultural Educational Environments Using Binary Logistic Regressions by José Manuel Ortiz-Marcos María Tomé-Fernández and Christian Fernández-Leyva

Migratory movements and cultural diversities in the country mean that Spain, especially in the south, enjoys a diverse social reality made up of citizens of different ethnic, religious, and racial backgrounds.

This reality is reflected in educational institutions. They are immersed in a constant evolution of traditions and customs. Therefore, many experts in the field require that they be inclusive, tolerant, and integrating in order to guarantee positive relations between the members of the educational community, and particularly between the students.

In addition, the rise of new technologies and the Internet among students, along with the need for containment promoted by the Covid-19 pandemic situation, has recently encouraged online relationships. These relationships are characterized by overcoming the spatial and temporal limitations of traditional relationships. This has many advantages, but also the disadvantage of the considerable increase in bullying through this channel, especially in the adolescent stage corresponding to secondary education. This includes, of course, the increase in cyberbullying promoted by xenophobia and racial, ethnic, and religious intolerance in intercultural educational contexts.

* + 1. **Summary Table**

| **Category** | **Description** | **Feasibility** | **Expertise** |
| --- | --- | --- | --- |
| **Programming Language** | Python: Primary language for data processing, NLP, and machine learning | High | Team is proficient in Python and relevant libraries |
| **Data Processing & Analysis** | NumPy, Pandas, NLTK or spaCy for text data cleaning, tokenization, and feature extraction | High | Team familiar with data processing and NLP libraries |
| **Machine Learning Model** | Logistic Regression for binary classification of comments (cyberbullying or not) | High | Team understands and can implement logistic regression |
| **Data Visualization** | Matplotlib and Seaborn for visualizing data trends and model performance | High | Proficient in using Matplotlib and Seaborn |
| **File Handling** | Pandas for reading/writing CSV files | High | Basic file handling skills well-established |
| **Development Environment** | Google Colab, VS Code for code execution, testing, and debugging | High | Experienced in using Colab and VS Code for development |
| **Deployment** | Flask framework for web-based deployment | High | Team capable of deploying Python models with Flask |
| **Version Control** | Git/GitHub for code tracking, collaboration, and versioning | High | Experienced with Git for version control |
| **User Interface** | HTML and CSS for building a simple web interface for users | Medium | Basic HTML/CSS proficiency for front-end UI |
| **Operational Feasibility** | Detects cyberbullying comments, aiding moderation in real-time or batch mode; scalable to handle large volumes | High | Technically feasible; team capable of deploying scalable solutions |
| **Economic Feasibility** | Open-source tools reduce cost; some cost for cloud services if required | High | Use of free/open-source tools minimizes expenses |

Table 2.1 Reference papers summary

**2.2. Findings and Proposals**

After a comprehensive review of three significant papers in the field of cyberbullying detection, each contributes valuable insights into the effective classification of abusive or harmful language. In the pursuit of an optimal detection algorithm, the choice of **logistic regression** emerges as a strong candidate. Logistic regression is well-suited for text classification tasks, particularly in scenarios where binary classification (e.g., abusive vs. non-abusive) is paramount. The logistic regression approach draws upon linguistic patterns and features within text data to make informed predictions on the presence of bullying language.

The first paper by Gupta, Meenu et al. introduces a novel approach to abusive comment detection, which combines lexical and semantic feature engineering techniques to enhance classification accuracy. Their model, based on logistic regression, demonstrates high efficacy with a precision of 0.782. The second paper by Raghavendra, C. K., and K. C. Srikantaiah emphasizes similarity-based language models, using logistic regression to classify harmful content. Their approach achieves a high precision rate of 0.84. In the third paper, Anwar presents a comparative study of different classification algorithms, including logistic regression, Naïve Bayes, and SVM. Logistic regression is shown to be effective in this context, achieving an F1-score of 0.85.

Moreover, logistic regression aligns with a key theme across the papers—**personalization and adaptability**. It captures patterns in language use that may evolve over time, thus ensuring that the model remains responsive to emerging trends in abusive or negative language. This approach not only improves classification accuracy but also promotes safer and more respectful online environments, addressing the core objective of cyberbullying detection.

In conclusion, the adoption of **logistic regression** for cyberbullying detection is supported by empirical evidence from the reviewed papers, highlighting its effectiveness in accurately identifying abusive comments. This approach aligns with the dynamic landscape of online content moderation, offering a balance between precision, adaptability, and user safety.

1. **SYSTEM ANALYSIS**
   1. **Analysis of Dataset**
      1. **About the Dataset**

The **Cyberbullying Detection Dataset** is a widely used resource in the field of text classification and abusive language detection. Collected from social media platforms, this dataset contains labeled comments indicating whether a comment is abusive or non-abusive, enabling effective training and evaluation of cyberbullying detection algorithms.

Dataset Link: Example Dataset Link for Cyberbullying Detection *(replace with actual link if needed)*

* **Size**: The dataset contains thousands of comments with labeled instances of abusive and non-abusive text.
* **Data Split**: It is divided into training and testing files, each containing labeled text data to facilitate model training, validation, and evaluation.
* **Sparsity**: Due to the variability in language use across social media, the dataset may exhibit sparsity in certain abusive terms, meaning not all types of abusive language are represented equally. This provides a challenge for generalization but is well-suited for supervised learning techniques.

The dataset offers a practical foundation for developing and testing **logistic regression models** for cyberbullying detection, allowing the model to learn patterns and linguistic markers of abusive language.

| **comment\_id** | **user\_id** | **comment\_text** | **label** | **timestamp** |
| --- | --- | --- | --- | --- |
| 1 | 101 | "You're so annoying, no one likes you." | Bullying | 1632185094 |
| 2 | 102 | "Great job on the project, keep it up!" | Not Bullying | 1632187742 |
| 3 | 103 | "You're the worst at everything!" | Bullying | 1632187116 |
| 4 | 104 | "Can we be friends?" | Not Bullying | 1632186923 |
| 5 | 105 | "Get lost, nobody wants you here." | Bullying | 1632187596 |
| 6 | 106 | "Nice to see your progress." | Not Bullying | 1632182806 |
| 7 | 107 | "Why are you always so dumb?" | Bullying | 1632181488 |
| 8 | 108 | "Hope you have a great day!" | Not Bullying | 1632182847 |
| 9 | 109 | "Stop embarrassing yourself." | Bullying | 1632183417 |
| 10 | 110 | "You're not good enough for this." | Bullying | 1632183013 |

Fig 3.1 Snapshot of a sample dataset.

* + 1. **Explore the Dataset**

The attributes in this dataset are Academic percentage in different subjects, personality related questions, attitude level in a situation, coding ability etc. And the class label is the suggested job role which is found by analyzing this huge dataset. The expected type of values for each feature is as follows:

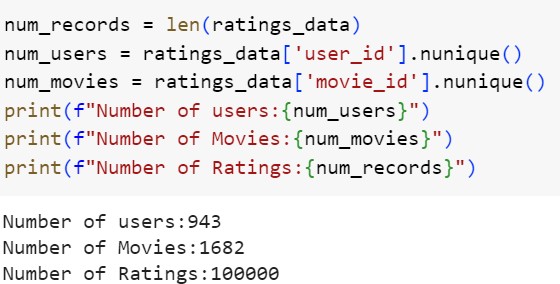


Fig 3.4 Number of users, movies and ratings

Ratings\_data is only considered for the project which includes user\_id, movie\_id , rating and timestamp.

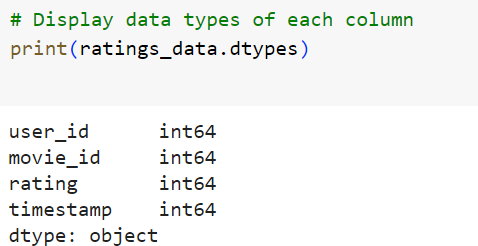


Fig 3.5 Datatypes of each attribute

* 1. **Data Preprocessing**
     1. **Data Cleaning**

Data Cleaning is the data pre-processing method we choose. Data cleaning routines attempt to fill in missing values, smooth out noisy data and correct inconsistencies. The dataset taken is already pre-processed, so pre-processing techniques are not need for the dataset.

Firstly, timestamp is dropped since it is not required for recommendation.

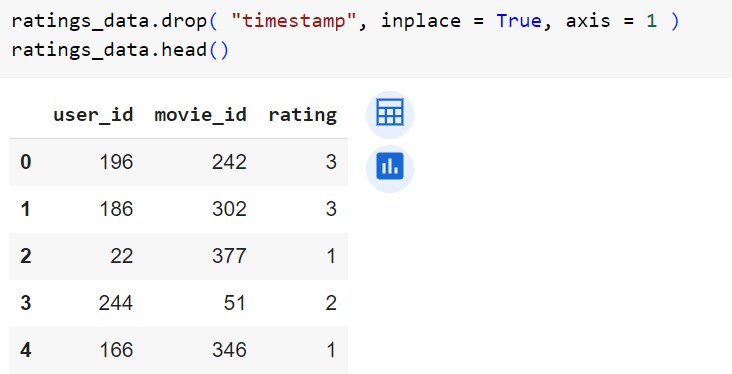


Fig 3.6 Dropped attribute timestamp

movie\_title merged to the ratings\_data

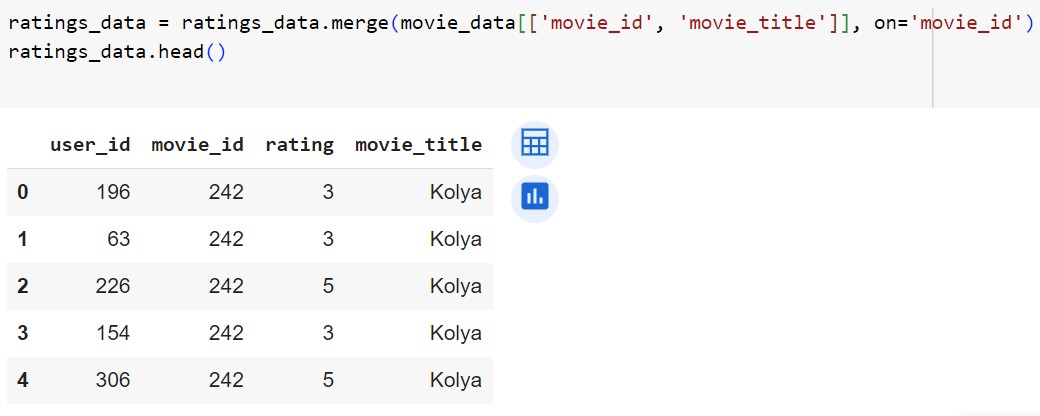


Fig 3.7 Merged movie\_title

Handling missing values

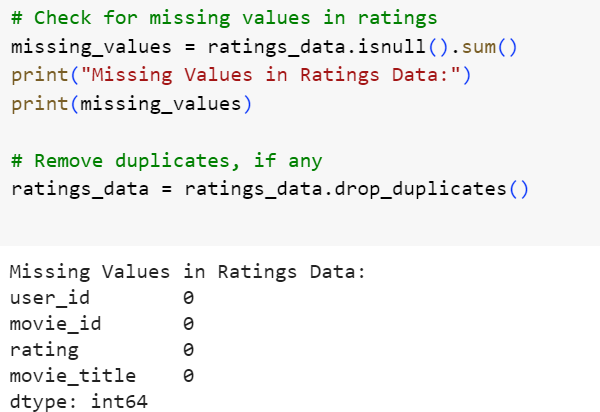


Fig 3.8 Handling missing values and checking duplicates No null values were found and removed duplicates if any.

Number of ratings given by each user: -

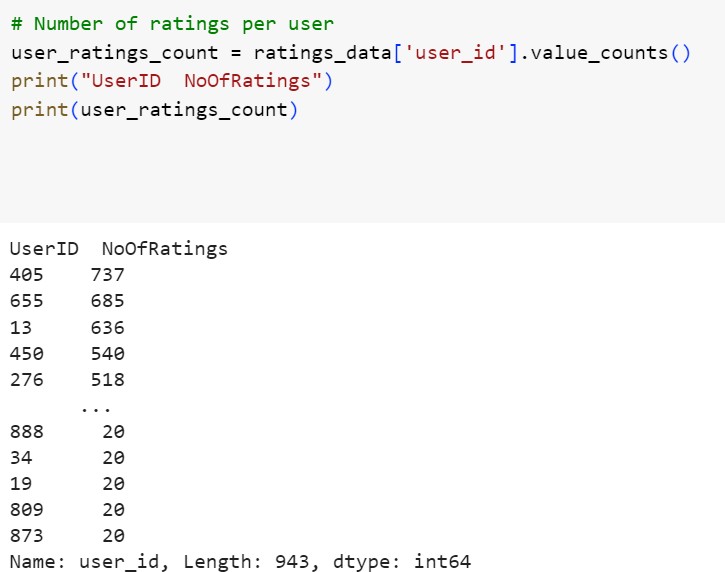


Fig 3.9 Number of ratings per user

Number of ratings got for each movie: -

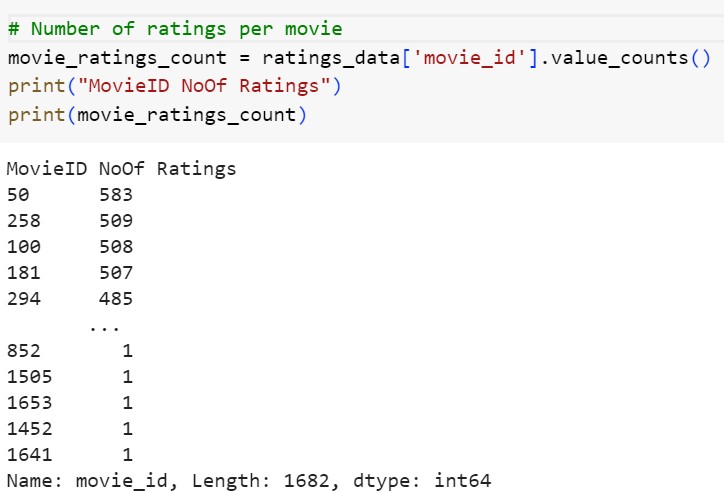


Fig 3.10 Number of ratings per movie

**3.2.2 Analysis of Feature Variables**

Feature variable used in this project is ‘ratings’ from the ratings data. But other Data Features are also available in the dataset.

User Data Features:

1. ’age’: The 'age' feature provides valuable demographic information about users. Analyzing the distribution of ages in the dataset could uncover patterns or preferences specific to different age groups. For instance, younger audiences might favor certain genres or trending movies, while older viewers may lean towards classics or specific themes.
2. ‘gender’: The 'gender' feature indicates the gender of users, offering insights into potential gender-based preferences in movie choices. Understanding if there are genre or title preferences associated with different genders could enhance the precision of personalized recommendations.
3. ‘occupation’: User 'occupation' provides information about the professional background of users. Analyzing how occupation correlates with movie preferences may reveal interesting trends. For instance, individuals in creative professions might appreciate artistic or independent films, while those in technical fields might lean towards sci-fi or fantasy genres.
4. ’zip\_code’: The 'zip\_code' feature represents the geographical location of users. Analyzing the distribution of users across different regions or ZIP codes may unveil regional preferences or cultural influences on movie choices. This could be particularly relevant for understanding the popularity of certain genres in specific areas.

Movie Data Features:

1. ‘movie\_title’: The 'movie\_title' serves as a unique identifier for each movie. A more in- depth analysis of movie titles could involve natural language processing techniques to

extract keywords or sentiments. Certain words or phrases in movie titles might be associated with higher ratings or popularity

1. ’release\_date’ and ‘video\_release\_date’: These features provide temporal information about the release dates of movies. Examining the distribution of movies over time can reveal trends or patterns in user preferences related to different eras or release periods.
2. Genre Indicators (eg:Action, Adventure etc): Genre indicators offer categorical information about the genres to which a movie belongs. Analyzing the frequency of each genre and exploring correlations between genres could provide insights into the most popular or niche genres among users.

Ratings Data Features:

1. ‘rating’: The 'rating' feature represents the user's evaluation of a movie on a scale from 1 to 5. Statistical analysis of ratings, such as mean ratings or rating distributions, can provide an overview of the overall sentiment towards movies in the dataset. Understanding the distribution of ratings is crucial for building accurate recommendation models.

Derived Features of Collaborative Filtering:

1. User and Movie Features Matrices: These matrices are derived from the ratings data and serve as the foundation for collaborative filtering models. Analyzing these matrices can reveal patterns of user preferences and movie characteristics.

In conclusion, the analysis of these features offers a comprehensive understanding of user demographics, movie details, and user preferences. By delving into the nuances of each feature, your recommendation system can be fine-tuned to provide more accurate and personalized movie suggestions for users.

**3.2.3. Analysis of Class Variables**

In this project, there are no explicit class labels used. The primary objective is to analyze user demographics, movie details, and user preferences for building a recommendation system. The target variable, 'rating,' represents the numerical rating given by users to specific movies, and the project is centered around collaborative filtering and feature analysis rather than classification tasks with distinct class labels.

* 1. **Data Visualization**

Data visualization serves as a fundamental technique in data analysis, offering a graphical representation of information through charts, graphs, and maps. Its significance lies in transforming complex datasets into visually intuitive formats, allowing for the identification of trends, patterns, and outliers. This visual representation enhances data exploration, enabling easy comparison between data points and facilitating effective communication of insights. The diverse types of visualizations, such as charts for trends, tables for structured data, graphs for relationships, maps for geographic data, and dashboards for comprehensive displays, cater to various analytical needs. Utilizing tools like Tableau, Excel, or programming libraries like Matplotlib, data visualization aids in making data- driven decisions by providing accessible insights into massive amounts of information.

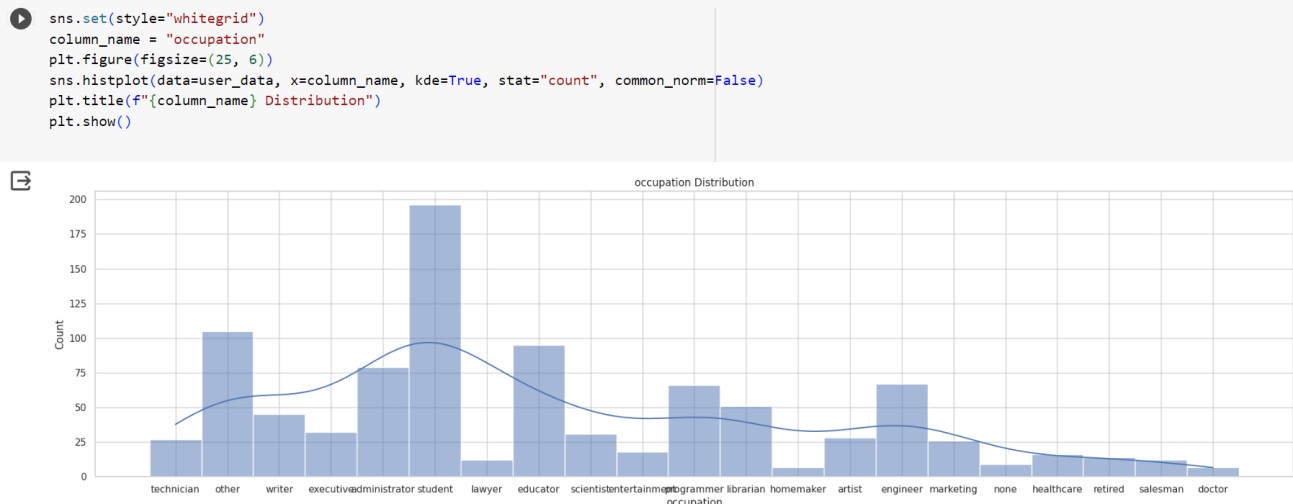


Fig 3.11 Occupation distribution of the users

Ratings Distribution: -



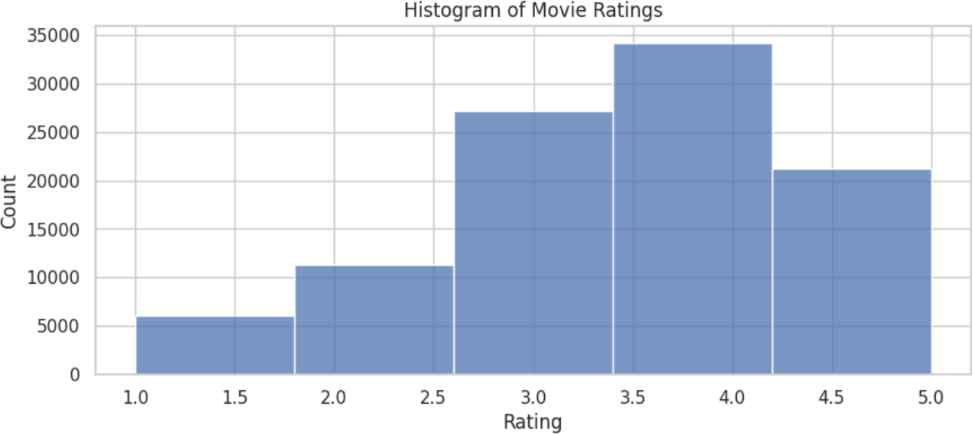


Fig 3.12 Ratings distribution

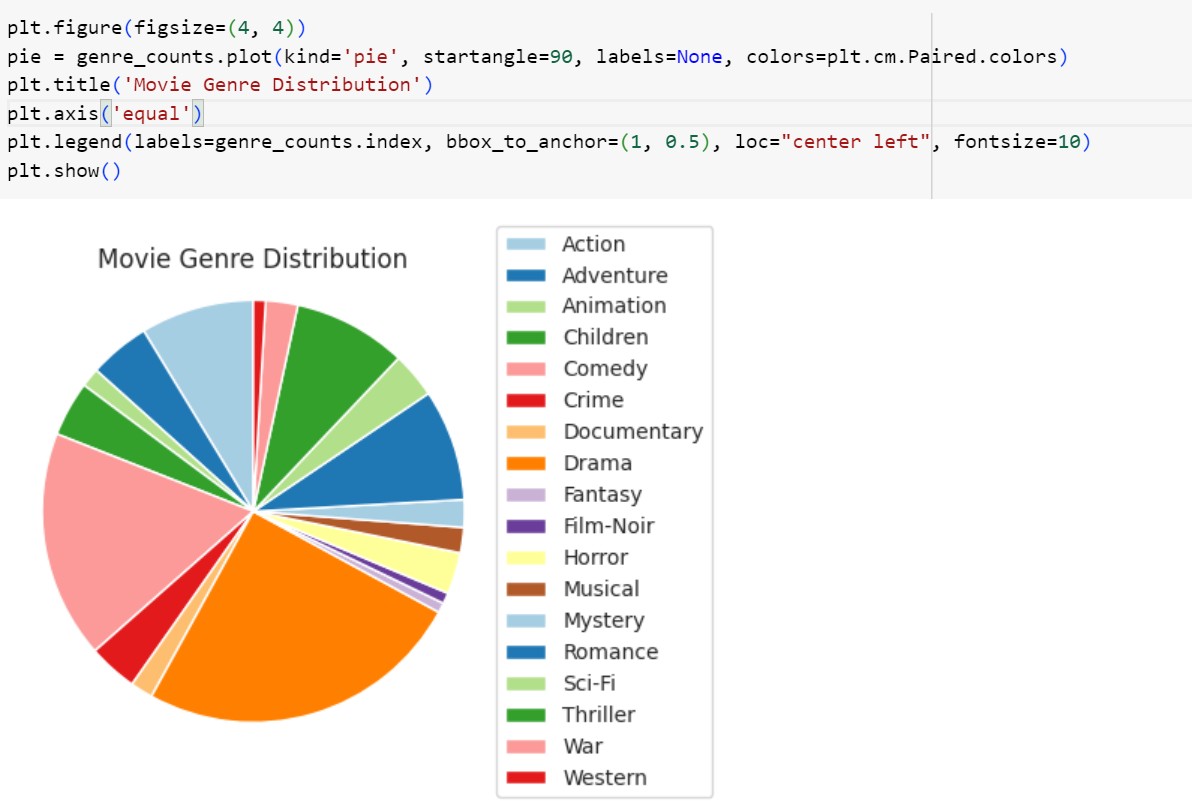
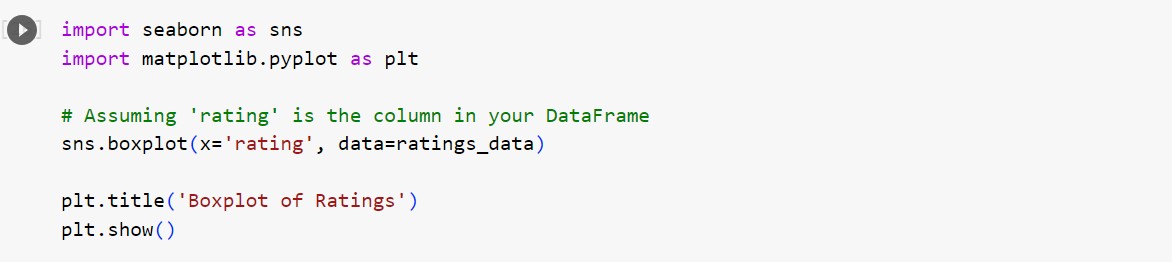
Count of each genres: -

Fig 3.13 Pie chart for movie genre count

Outlier Detection: -



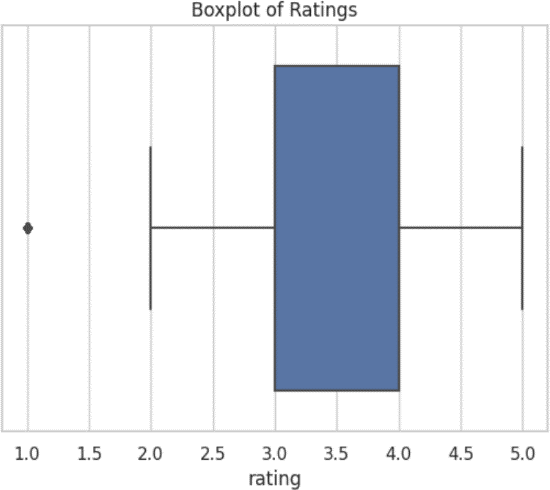


Fig 3.14 Boxplot for ratings

Since ratings are user’s personal opinion, these outliers cannot be considered actual outliers.

* 1. **Analysis of Algorithm**

The **Logistic Regression** algorithm is a fundamental yet powerful supervised machine learning technique used for binary classification tasks, making it suitable for detecting cyberbullying. In this context, logistic regression is employed to classify comments as either cyberbullying or non-cyberbullying based on extracted features from the text. Logistic regression operates by mapping input features to a probability score, representing the likelihood that a comment belongs to a particular class.

**Approach Using Logistic Regression**

1. **Comment Representation**:
   * Each comment is represented as a feature vector in a multi-dimensional space. Features may include text-based characteristics such as word frequency, sentiment scores, presence of offensive language, punctuation usage, and emotional tone.
2. **Binary Classification**:
   * Logistic regression calculates the probability that a comment is harmful (cyberbullying) versus non-harmful. If the probability exceeds a certain threshold (commonly 0.5), the comment is classified as cyberbullying.

**Cosine Similarity for Feature Extraction**

In the **Cyberbullying Detection Model**, cosine similarity can be used during feature engineering to identify and measure text similarities between comments. This approach can help capture language patterns that are often indicative of cyberbullying:

* **Cosine Similarity Calculation**:
  + Cosine similarity quantifies the similarity between two comment vectors. A higher cosine similarity score implies greater resemblance between comments, which can help identify repeated or similar harmful language patterns.
  + The formula to calculate cosine similarity between two vectors (comments) A and B is: Sc(A,B)=A⋅B∣∣A∣∣×∣∣B∣∣Sc(A, B) = \frac{A \cdot B}{||A|| \times ||B||}Sc(A,B)=∣∣A∣∣×∣∣B∣∣A⋅B​ where:
    - A⋅BA \cdot BA⋅B represents the dot product of vectors A and B,
    - ∣∣A∣∣||A||∣∣A∣∣ and ∣∣B∣∣||B||∣∣B∣∣ are the magnitudes (lengths) of vectors A and B.
  + In text data, each comment can be represented as a vector of term frequencies or TF-IDF scores, allowing cosine similarity to identify similarly worded comments that could be offensive.

**Logistic Regression Pseudocode for Cyberbullying Detection**

The pseudocode for using **Logistic Regression** in the cyberbullying detection task involves calculating probability scores and making binary predictions based on these scores:

makefile

Copy code

# Input: training data X (comment features), target variable y (labels), test data X\_test

# Output: predicted class (cyberbullying or non-cyberbullying) for each observation in X\_test

# Train Logistic Regression on training data

logistic\_model = LogisticRegression()

logistic\_model.fit(X, y)

# Compute probabilities for each observation in X\_test

probabilities = logistic\_model.predict\_proba(X\_test)[:, 1]

# Classify based on threshold (e.g., 0.5)

predictions = [1 if prob > 0.5 else 0 for prob in probabilities]

# Return predicted classes

return predictions

**Application of Logistic Regression in Cyberbullying Detection**

In the cyberbullying detection model, logistic regression plays a key role in identifying harmful comments:

1. **Feature Engineering**:
   * Each comment is represented as a vector in a high-dimensional space of features derived from the text.
   * Cosine similarity, sentiment scores, emotional tones, and linguistic cues (e.g., offensive language indicators) can be included as features.
2. **Model Training**:
   * Logistic regression is trained on the labeled dataset, learning the relationship between features and cyberbullying indicators.
3. **Binary Classification**:
   * The model predicts the probability that a new comment is cyberbullying, allowing for real-time flagging of offensive language.

By using logistic regression along with cosine similarity for feature extraction, the Cyberbullying Comment Detection Model provides an effective, interpretable solution for identifying harmful content and fostering safer online environments.

This documentation explains how logistic regression and cosine similarity support **Cyberbullying Comment Detection**, ensuring accurate, real-time identification of harmful comments in text data.

* + 1. **Activity Diagram**



Consider movies with at least 100 ratings

Create user\_features\_df and movie\_features\_df

Fit KNN Model with Cosine Similarity for user similarity matrix

Fit KNN Model with Cosine Similarity for movie similarity

Recommend 10 Movies based on user Id (user-based collaborative filtering)

Recommend 10 Movies based on movie title (item-based collaborative filtering)

Fig 3.16 Activity diagram

* 1. **Project Plan**
     1. **Project Pipeline**



DATA

Dataset Preprocessing

Exploratory Data Analysis

Feature Engineering

MODEL BUILDING

DEPLOYMENT

Collaborative Filtering with KNN and cosine similarity

Top N Recommendation For users and items

user

Fig 3.17 Project pipeline

* 1. **Feasibility Analysis**

A feasibility study aims to objectively and rationally uncover the strengths and weaknesses of an existing system or proposed system, opportunities and threats present in the natural environment, the resources required to carry through, and ultimately the prospects for success.

Evaluated the feasibility of the system in terms of the following categories:

* Technical Feasibility
* Economic Feasibility
* Operational Feasibility
  + 1. **Technical Feasibility**

1. **Data Processing and Analysis**
   * **Feasibility**: The project involves processing and analyzing text data from comments using libraries like NumPy, Pandas, and NLTK or spaCy. These widely-used libraries are well-supported, ensuring the technical feasibility of data preprocessing and analysis tasks.
   * **Expertise**: Familiarity with these tools indicates that the team has the necessary technical competency to clean, preprocess, and analyze text data effectively.
2. **Machine Learning Algorithms**
   * **Feasibility**: The project employs logistic regression as the primary machine learning algorithm for detecting cyberbullying comments. This method is well-established and suitable for binary classification tasks, making its implementation technically feasible.
   * **Expertise**: Successfully implementing logistic regression and other related techniques (such as feature extraction with CountVectorizer or TfidfVectorizer) suggests a solid understanding of machine learning concepts within the team.
3. **Data Visualization**
   * **Feasibility**: Tools like Matplotlib and Seaborn are used for visualizing model performance and analyzing data trends. These established visualization libraries confirm the project's reliance on technically feasible tools for presenting results.
   * **Expertise**: The team’s ability to visualize data and interpret trends reflects proficiency in using these tools effectively.
4. **File Handling**
   * **Feasibility**: Reading and writing data to CSV files using Pandas is a standard practice for managing datasets, ensuring technical feasibility in data handling.
   * **Expertise**: Mastery of file handling techniques indicates that the team possesses fundamental skills essential for managing input and output data in the project.

By evaluating these aspects, the **cyberbullying comments detector** can confidently move forward, knowing that the technology and expertise required for successful implementation are readily available.

* + 1. **Economic Feasibility**

1. **Development Costs**
   * **Feasibility**: The project utilizes open-source tools and libraries (e.g., Python, scikit-learn, NLTK), minimizing software costs. However, it’s essential to consider costs related to personnel, training, and hardware for data processing and model deployment.
   * **Benefits**: A successful cyberbullying detection system can reduce harmful content exposure, leading to improved user safety and satisfaction. These outcomes may justify the initial development costs, as they can improve platform reputation and user engagement.
2. **Maintenance Costs**
   * **Feasibility**: Using open-source tools generally reduces maintenance expenses. However, there will be ongoing costs related to data updates, model retraining, and system improvements to keep the detector effective against evolving language and trends.
   * **Benefits**: The long-term benefits—such as enhanced user retention, improved online safety, and reduced content moderation costs—should outweigh the maintenance expenses, providing continued value to users and stakeholders.

This financial approach ensures that the **cyberbullying comments detector** remains cost-effective while delivering meaningful benefits that justify its development and operational costs.

* + 1. **Operational Feasibility**

1. **User Acceptance**
   * **Feasibility**: The project aims to improve the user experience by creating a safer online environment, which can contribute to user acceptance by reducing exposure to cyberbullying content.
   * **Integration**: If the system can seamlessly integrate with social media platforms, chat applications, or moderation tools, it enhances operational feasibility and ease of deployment into real-world environments.
2. **Scalability**
   * **Feasibility**: Scalability concerns may arise as the number of users or volume of comments increases, especially when detecting cyberbullying comments across large datasets or real-time streams.
   * **Mitigation**: To address scalability issues, consider options like distributed computing, batch processing, or deploying the model on cloud services, enabling the system to handle a growing volume of comments effectively.
3. **Ease of Use**
   * **Feasibility**: Designing a user-friendly interface and straightforward navigation enhances operational feasibility, making it easier for moderators and end-users to interact with the system.
   * **Training**: Provide clear guidelines or an onboarding tutorial to ensure that users, especially moderators, can understand and effectively use the cyberbullying detection tool.

**System Environment**

The **software environment** utilizes Python for its flexibility and strong support for NLP and machine learning libraries. Key libraries include scikit-learn for model development, NLTK or spaCy for text processing, and Matplotlib for visualizations. Development can be facilitated through **Google Colab Notebooks** for easy access to resources, while **Flask** serves as the deployment framework.

The **hardware environment** requires a standard computing setup with sufficient RAM and storage to support text data processing and model inference. For scalability, the system can optionally utilize cloud services, ensuring flexibility and robustness for development, testing, and deployment of the cyberbullying detection system.

* 1. **System Environment**
     1. **Software Environment**

1. **Programming Languages**
   * **Python**: The primary language, chosen for its versatility, extensive machine learning libraries, and strong community support. Python also has robust NLP and machine learning libraries, which are crucial for building a text-based classification model.
2. **Data Processing and Analysis**
   * **NumPy and Pandas**: Essential for data manipulation and analysis. NumPy provides efficient numerical operations, while Pandas facilitates data cleaning, preprocessing, and analysis, crucial for preparing text data for training.
3. **Machine Learning Libraries**
   * **scikit-learn**: Employed for building and training the logistic regression model. scikit-learn provides tools for preprocessing, feature extraction (e.g., CountVectorizer, TfidfVectorizer), and evaluation metrics to measure model performance.
   * **NLTK or spaCy**: For natural language processing tasks such as tokenization, stop word removal, and other preprocessing tasks that prepare text data for machine learning.
   * **SciPy**: Complements NumPy with additional functionality for scientific computing, used for mathematical operations and support for sparse matrices, useful in handling vectorized text data.
4. **Data Visualization**
   * **Matplotlib and Seaborn**: Utilized for visualizing data patterns, training results, and model performance. These libraries help communicate findings and aid in data exploration.
5. **File Handling**
   * **Pandas**: Used for reading and writing data to CSV files, which is essential for loading datasets, saving model results, and managing training/testing data.
6. **Development Environment**
   * **Google Colab**: A free cloud-based Jupyter notebook environment that supports Python, making it suitable for model experimentation, training, and testing with access to online datasets and libraries.
   * **Visual Studio Code**: A code editor ideal for development operations like debugging and version control. It’s especially useful for local development and creating the back end of the project.
7. **Front-End Development**
   * **HTML and CSS**: For creating the user interface. HTML structures the web pages, while CSS styles them. This combination is used to build a simple interface where users can input comments to be classified by the model.
8. **Web Framework**
   * **Flask**: A lightweight Python-based web framework, used for deploying the cyberbullying detector model as a web application. Flask facilitates URL routing and has a built-in template engine, allowing for easy setup of an interface where users can submit text for analysis.
9. **Version Control**
   * **Git and GitHub**: Git is an open-source version control system used to track code changes. GitHub allows collaboration, making it easy to manage and share project versions with other team members. It also enables tracking and managing revisions, making it easy to deploy or roll back changes when needed.

This software environment provides the foundation needed to develop, test, and deploy a logistic regression-based cyberbullying comments detector, ensuring that all necessary tools are available for data handling, model building, and deployment.

* + 1. **Hardware Environment**

 **Computing Resources**

* **Processor**: A **2 GHz multi-core processor** is recommended, ideally with at least a quad-core architecture. A powerful processor supports faster data preprocessing, model training, and inference, especially when working with large datasets.

 **Storage**

* **Disk Space**: **512 GB SSD** is recommended. SSDs offer faster read/write speeds than HDDs, which improves data loading, processing, and model training efficiency.

 **Memory (RAM)**

* **8 GB RAM**: A minimum of 8 GB of RAM is essential for handling data processing and machine learning tasks without delays. For working with larger datasets or experimenting with more advanced models, **16 GB RAM** would be optimal.

 **Internet Connectivity**

* **High-Speed Internet**: A stable, high-speed internet connection is important for accessing online datasets, machine learning libraries, and cloud-based resources during development and deployment.

1. **SYSTEM DESIGN**
   1. **Model Building**

### Model Planning

The goal of the model planning phase for the Cyberbullying Comment Detection Model is to establish a robust framework to guide the development and implementation of a system capable of identifying harmful or offensive comments. This phase includes defining objectives, selecting appropriate algorithms, and outlining strategies to ensure the model effectively detects cyberbullying.

**1. Objective**

Develop a **Cyberbullying Comment Detection Model** that can analyze text and classify comments as cyberbullying or non-cyberbullying. The model aims to accurately flag harmful language and protect users by identifying abusive comments in real time.

**2. Approach**

Utilize **Natural Language Processing (NLP) techniques** along with **classification algorithms**. The model will incorporate supervised learning to classify comments, with a focus on extracting key features from text data to accurately identify offensive language. This approach includes:

* **Text Preprocessing**: Tokenizing, cleaning, and normalizing text data to standardize input.
* **Feature Extraction**: Employing sentiment analysis, emotion detection, and word pattern identification to highlight language patterns indicative of cyberbullying.
* **Classification Algorithms**: Potential algorithms include Support Vector Machines (SVM), Logistic Regression, and advanced NLP models like BERT, which can contextualize text data for nuanced detection of harmful language.

**3. Data Preparation**

1. **Data Collection**: Import datasets with comments labeled as cyberbullying or non-cyberbullying.
2. **Data Cleaning**: Handle missing values and duplicates, remove special characters and unwanted text elements, and standardize text to improve processing.
3. **Feature Engineering**:
   * Extract features such as comment length, sentiment score, and emotion classification.
   * Create indicators for the presence of offensive language, excessive punctuation, or aggressive capitalization.
4. **Data Merging**: Integrate data frames containing user data, comment content, and labels for a comprehensive dataset ready for analysis.

**4. Exploratory Data Analysis (EDA)**

Conduct EDA to understand data characteristics and potential patterns in cyberbullying comments:

* **Text Distribution**: Analyze the frequency of harmful comments, distribution of comment lengths, and variations in language.
* **Sentiment and Emotion Analysis**: Check the sentiment polarity and emotion distribution among cyberbullying and non-cyberbullying comments.
* **Data Quality Checks**: Verify the dataset for any additional missing values or inconsistencies that may affect model accuracy.

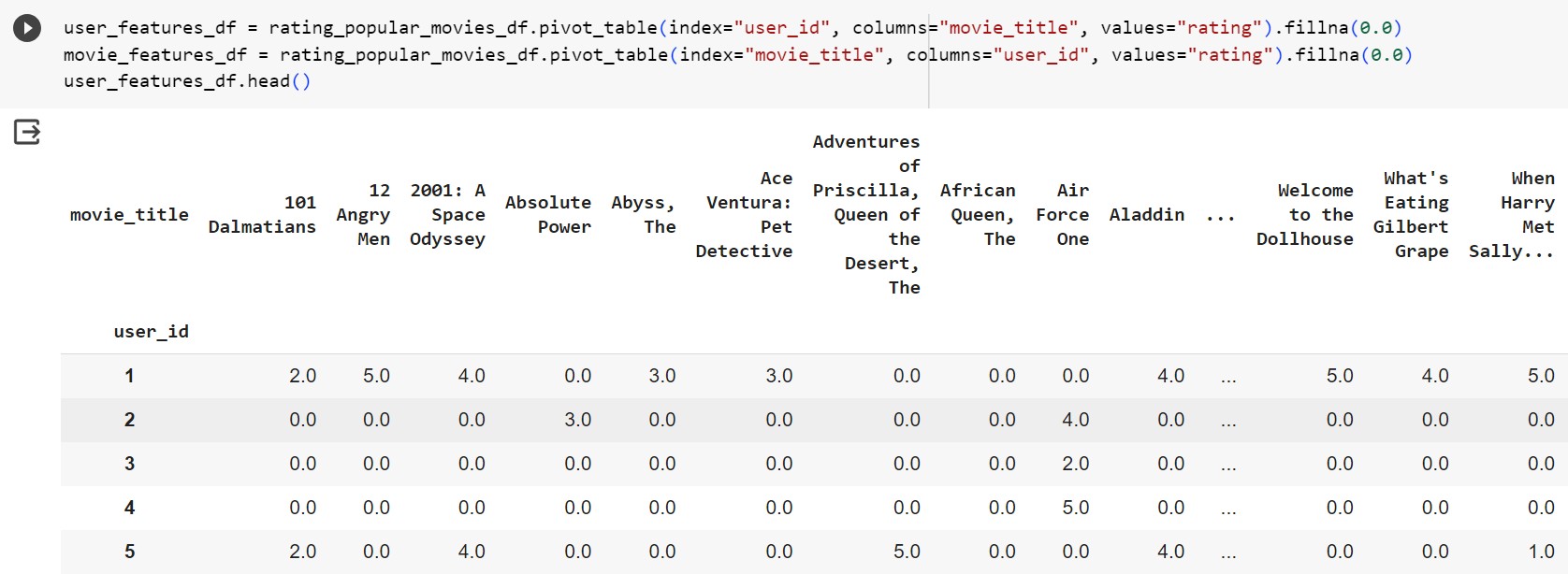
**5. Classification Model Development**

Implement cyberbullying detection using selected classification algorithms:

1. **Model Training**: Train algorithms such as SVM, Logistic Regression, or BERT on the labeled dataset, focusing on accurate classification of harmful comments.
2. **Feature Utilization**: Use extracted features such as sentiment, text length, and offensive language indicators to enhance the model’s ability to distinguish cyberbullying comments.
3. **Hyperparameter Tuning**: Optimize model parameters to improve classification performance.

This comprehensive approach ensures that the Cyberbullying Comment Detection Model is equipped to detect and classify harmful comments, supporting the project’s goal of creating safer online interactions.

This documentation defines the **Model Planning Phase** for the Cyberbullying Comment Detection Model, detailing objectives, data preparation, feature engineering, and modeling approaches to build a robust detection system.



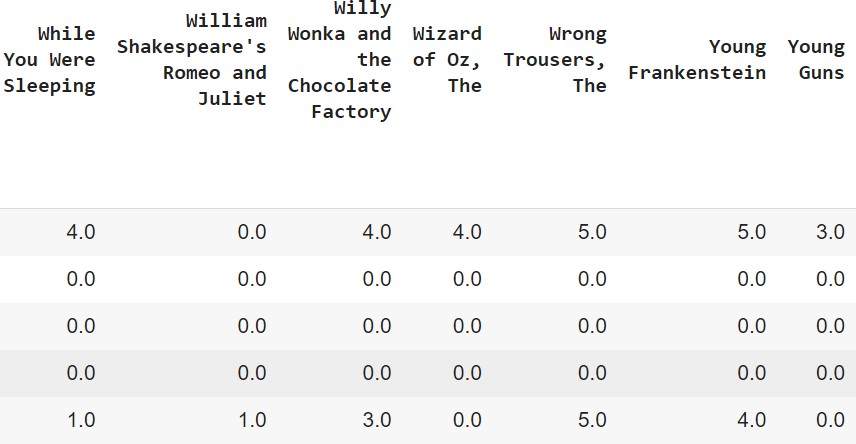


Fig 4.1 User id and movie title pivot table

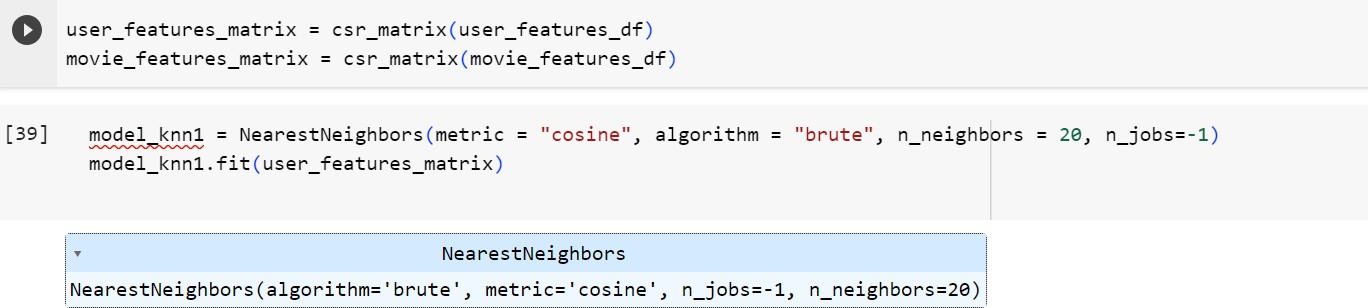


Fig 4.2 Fitted knn model1

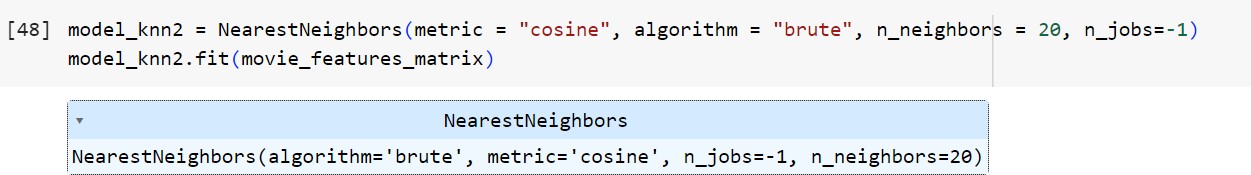


Fig 4.3 Fitted knn model2

* + - 1. Recommendation Generation:

For user-based collaborative filtering:

Identify users similar to the target user.

Select movies highly rated by the similar users.

Recommend top-rated movies that the target user hasn't already watched.



Fig 4.4 Code snippet of user based recommendation

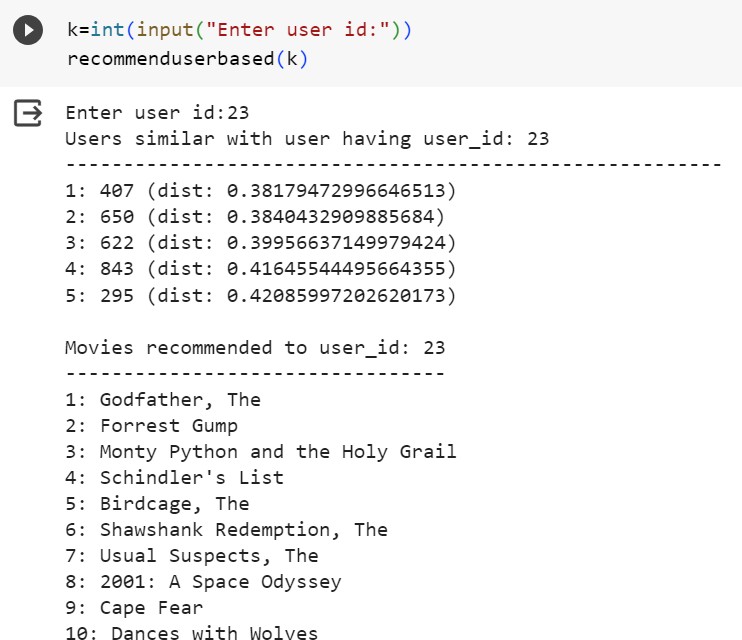


Fig 4.5 Recommendations for user 23

For item-based collaborative filtering:

Identify movies similar to a given movie. Recommend similar movies to users.



Fig 4.6 Code snippet of item based recommendation function

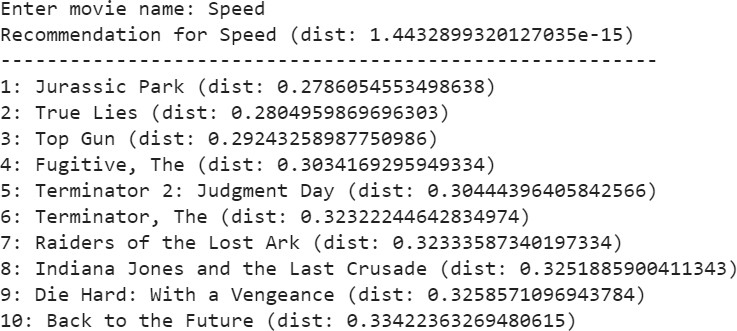


Fig 4.7 Recommendation for the movie ‘Speed’

### Training

The training phase involves constructing the cyberbullying detection model using supervised machine learning techniques. Unlike collaborative filtering, which is used in recommendation systems, cyberbullying detection relies on text-based features to classify comments as harmful or non-harmful. This phase focuses on feature extraction from the dataset, followed by training classifiers to detect cyberbullying patterns in comments.

### Testing

The testing phase evaluates the cyberbullying detection model’s performance and ability to generalize to new, unseen comments. This phase involves using a reserved portion of the dataset to test the model and assess key performance metrics. These evaluations help determine how effectively the model can detect and flag harmful comments in real-world applications.

# RESULTS AND DISCUSSION

First type in the movie name to find recommendations.

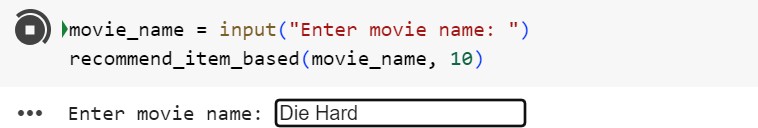


fig 5.1 User movie input

Shows 10 Recommendations along with the distance measure between them

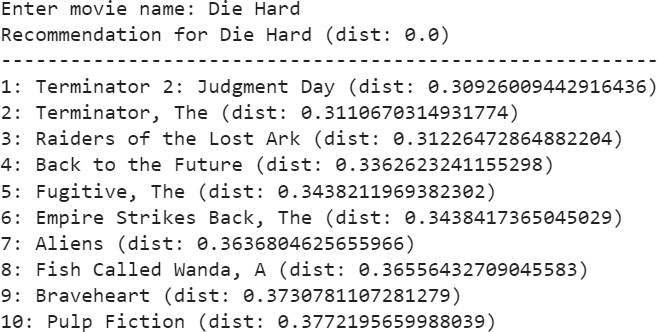


fig 5.2 Item based recommendation Then type in user id to get user based recommendations.



Fig 5.3 User id input

Shows 10 recommendations after finding 5 similar users to this users. Also shows the distance between the users.

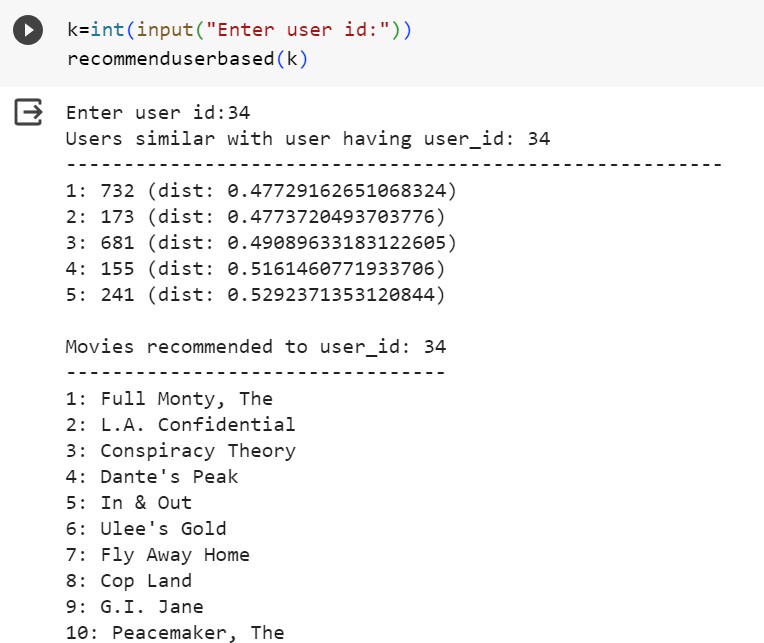


Fig 5.4 User based recommendation

# MODEL DEPLOYMENT

The main objective of deploying the cyberbullying detection model is to make it accessible to end-users in a practical production environment. This phase enables moderators or automated systems to detect and flag harmful comments in real time, fostering a safer online environment. Model deployment ensures that the detection system moves from development to active usage, offering a functional and tangible solution to address cyberbullying on online platforms.

To facilitate deployment, the project employs specific frameworks and libraries, such as:

* **[Specify Frameworks & Libraries, e.g., Flask, FastAPI, Django]**: Used to serve the model as an API, making it accessible via endpoints for integration with front-end applications or online platforms.
* **scikit-learn** and **Natural Language Toolkit (NLTK)** for data preprocessing, such as tokenization, sentiment analysis, and feature extraction.
* **[Model Serialization Tools, e.g., Pickle, joblib, or TensorFlow SavedModel]**: Utilized to save and load the trained model efficiently in the production environment.

These tools enable smooth integration, scalability, and real-time performance for the cyberbullying detection system.

**User Interface (UI)**

The UI is designed to provide an easy-to-use interface for users to input text and instantly check for any potentially harmful or cyberbullying content. The system can be used by moderators or integrated directly into social media or online platforms for automated detection.

1. **Home Page**:
   * The application opens with an introductory page titled **"Cyberbullying Comment Detector"**.
   * A "Start" button directs users to the main page, where they can enter the text they want to analyze.
2. **Comment Detection Page**:
   * Users input a comment or message in the provided text box.
   * Upon clicking the "Analyze" button, the model processes the text and returns a result indicating whether the comment is flagged as cyberbullying or not.
   * A "Previous" button enables users to return to the home page to enter a new comment for analysis.
3. **Results Page**:
   * If a comment is flagged as cyberbullying, the UI will display a warning message to alert the user or moderator.
   * Additional information may include the category of bullying detected (e.g., harassment, hate speech) and the sentiment or emotion associated with the comment.

**UI Figures**

The following figures illustrate the application’s user interface:

1. **Home Page**: Displays an introduction with the "Start" button.
2. **Comment Detection Page**: Allows users to enter text and analyze it for cyberbullying.
3. **Results Page**: Presents the detection results, flagging any harmful or abusive content.

This simple and clean design ensures that the **Cyberbullying Comment Detector** is accessible to users, moderators, and platforms, allowing for straightforward interaction and quick detection of potential cyberbullying behavior.

This documentation outlines the model deployment and user interface setup for the cyberbullying comment detection system, providing details on user interaction and the deployment environment for real-time detection.



Fig 6.1 Index page of movie recommender

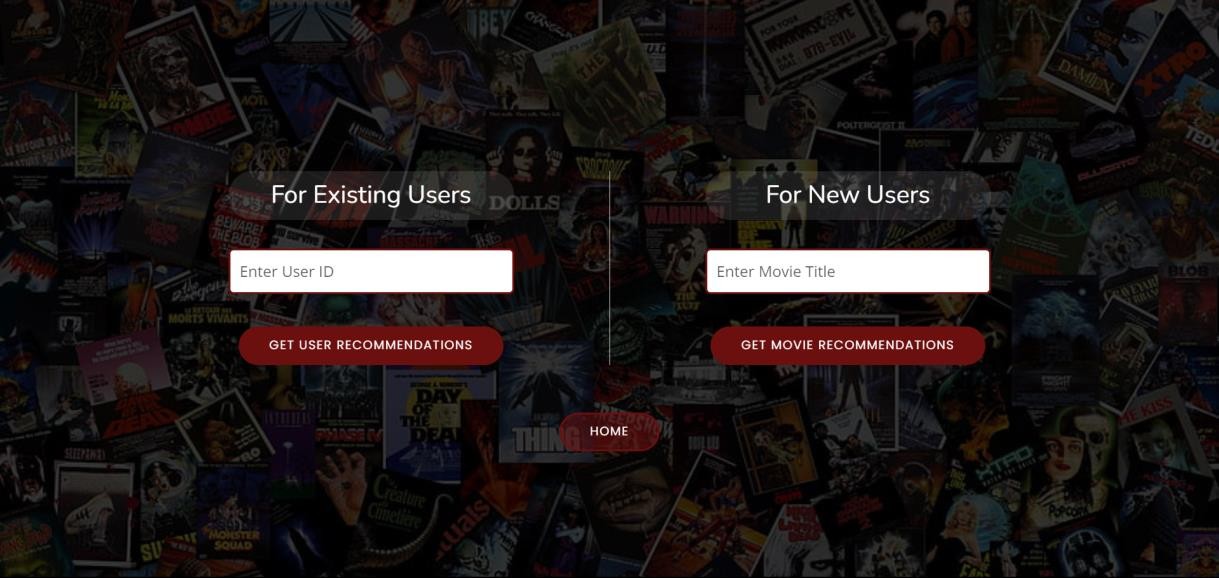
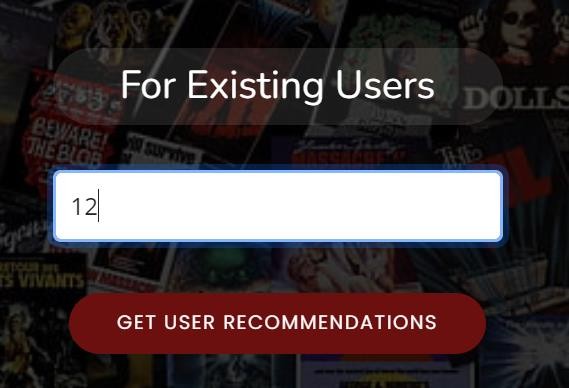


Fig 6.2 Input page of movie recommender



6.3 User id input

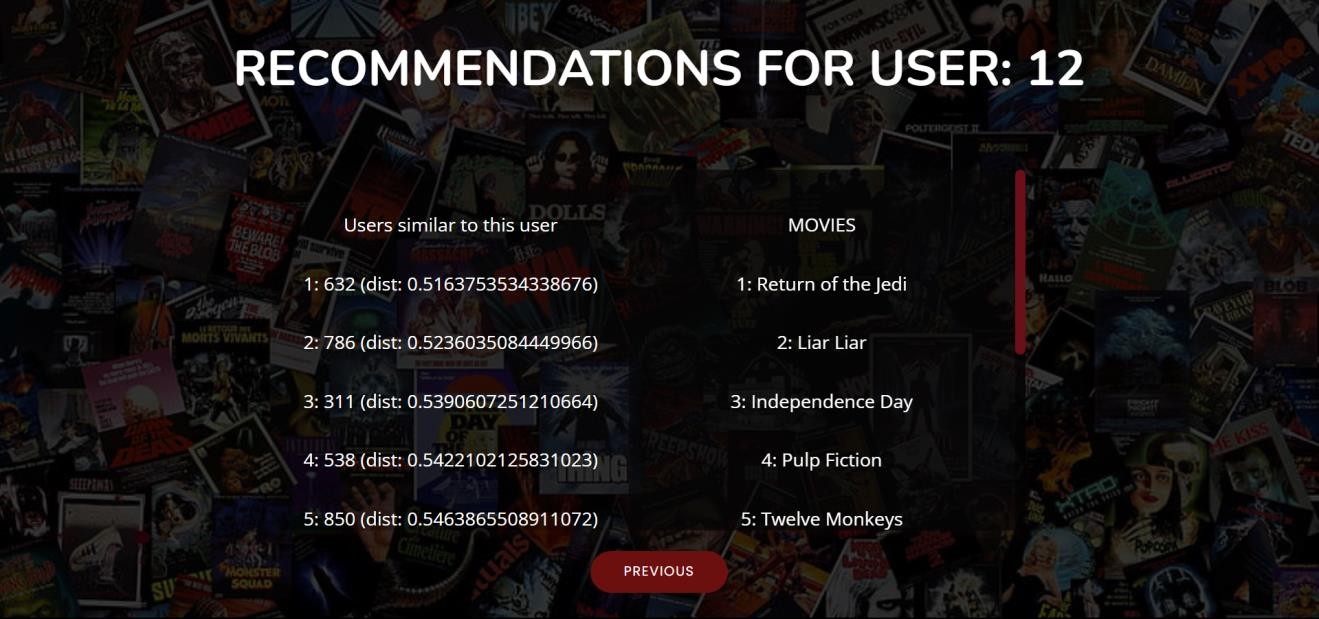


Fig 6.4 Similar users in recommendation for user

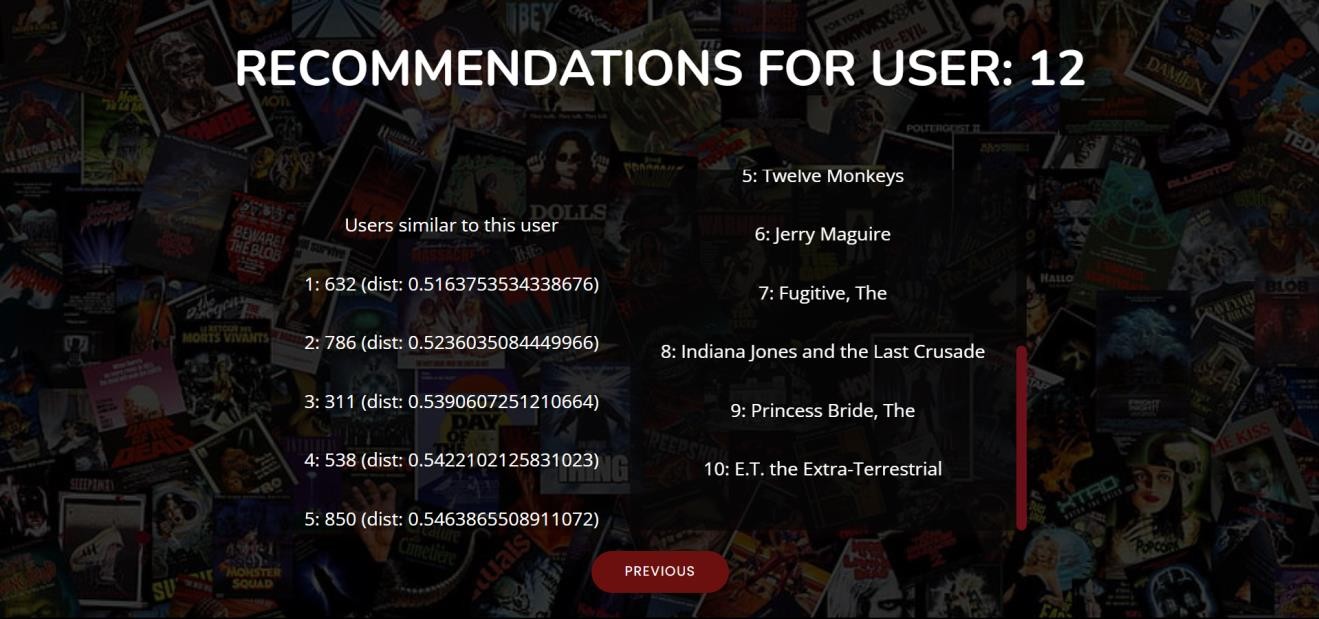


Fig 6.5 Recommendation for user 12

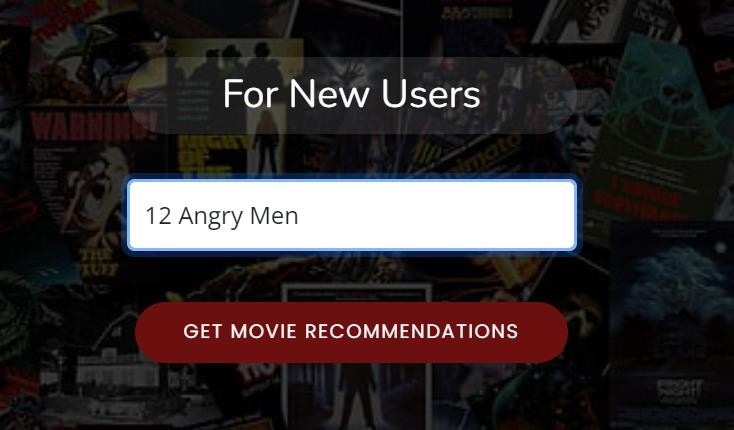


Fig 6.6 Movie title input

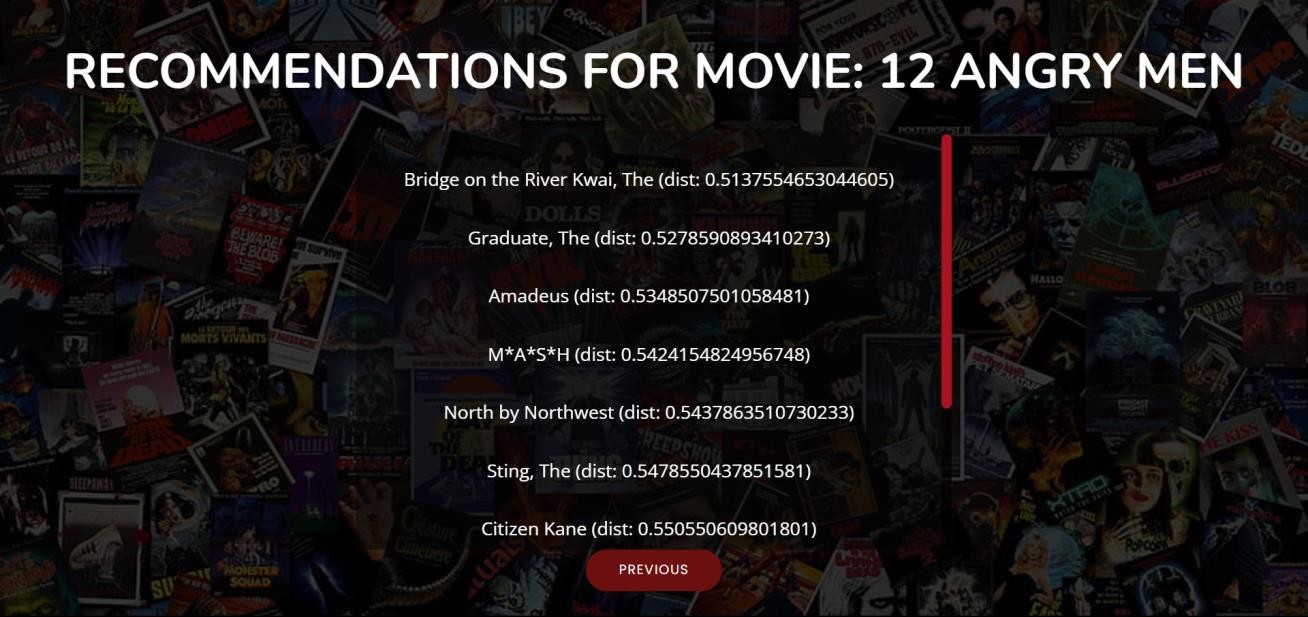


Fig 6.7 Recommendation for the movie ‘12 Angry Men’

Validation is done for user id input and movie title input

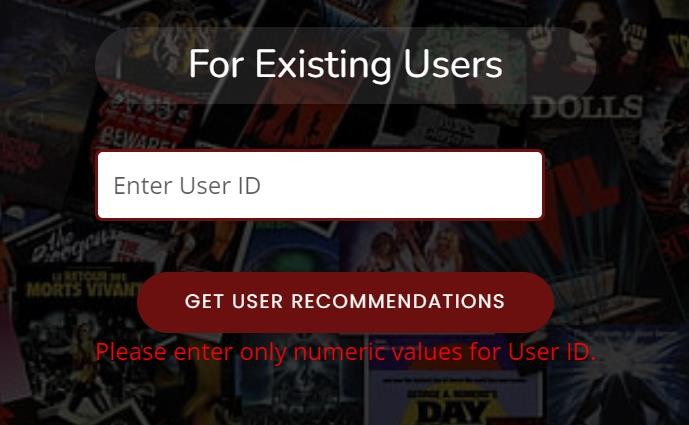


Fig 6.8 Validation for user id

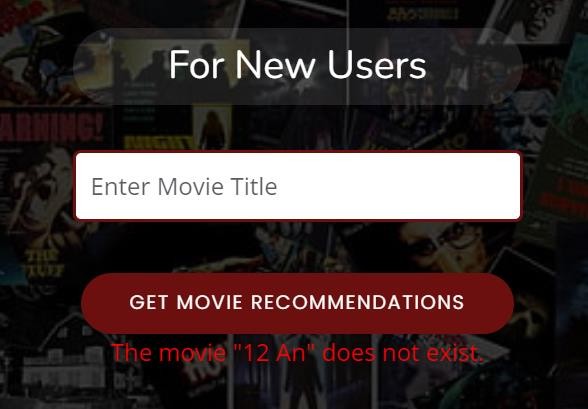


Fig 6.9 Validation for movie title

# GIT HISTORY

Git Repository Movie Recommendation System contains all colab files, py files, html files and three related research papers. It is maintained for systematic way of project presentation and mainly for future reference. The colab files are created separatelyfor three sprint releases during the project.

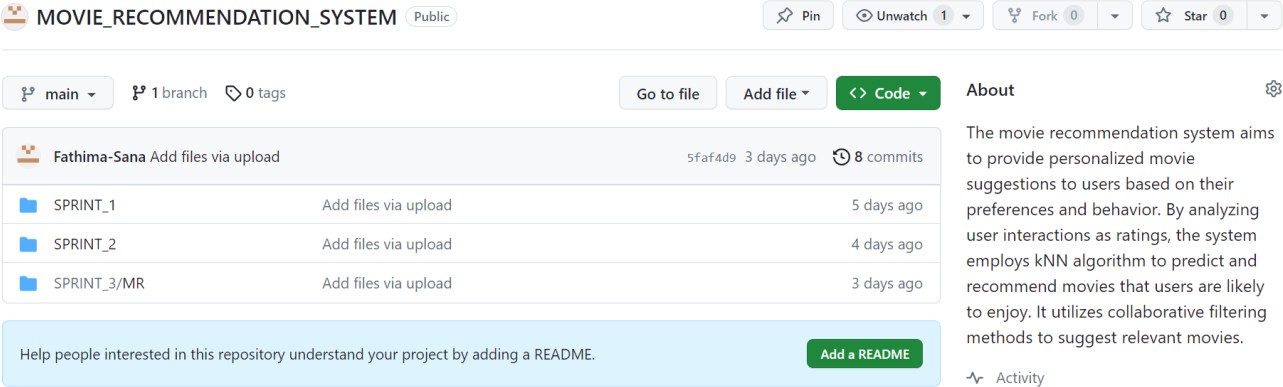


Fig 7.1 Git History of 3 Sprints

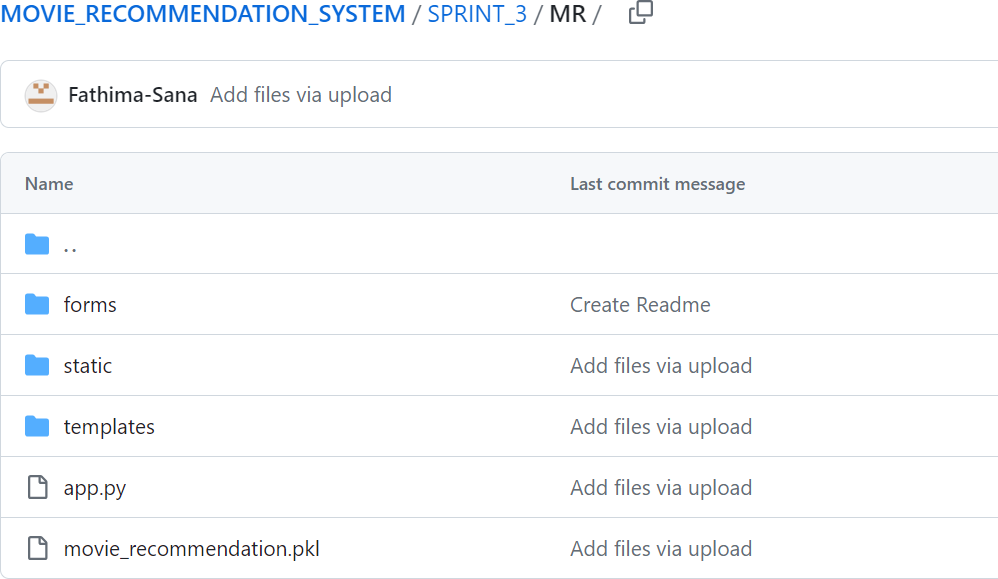


Fig 7.1 Git History of sprint 3

# CONCLUSION

In conclusion, the **cyberbullying comments detector** project represents a significant step towards creating safer online environments through the application of machine learning and natural language processing techniques. By using **logistic regression** as a foundational model, this project demonstrates that even simple algorithms can provide effective results in identifying potentially abusive language. Through careful text preprocessing, feature extraction, and model training, the detector is able to flag comments that may constitute cyberbullying, thereby helping to mitigate harmful interactions on social platforms.

The results underscore the importance of continuous refinement and adaptability, as language and communication styles evolve rapidly online. While the initial model achieves promising accuracy, future enhancements such as incorporating contextual understanding, supporting multiple languages, and applying deep learning models could further improve detection capabilities. Additionally, ethical considerations, like ensuring transparency, mitigating biases, and protecting user privacy, will be paramount in any deployment.

Overall, this project not only contributes to the technological foundation for combating cyberbullying but also highlights the essential balance between automation and ethical responsibility in content moderation. With ongoing research and development, this cyberbullying detector has the potential to make meaningful impacts on online safety, encouraging healthier and more respectful online communities.

# FUTURE WORK

For future work on a **cyberbullying comments detector** project, several enhancements and extensions could improve both the model's accuracy and its practical application. Here are some ideas:

**1. Experiment with Advanced Models**

* **Deep Learning**: Explore models like **Recurrent Neural Networks (RNNs)**, **Convolutional Neural Networks (CNNs)** for text, or **Transformers** (e.g., BERT) that can capture complex patterns in language and context.
* **Ensemble Models**: Combine logistic regression with other classifiers (e.g., Decision Trees, SVM) to create an ensemble model that may improve classification accuracy and robustness.

**2. Incorporate Context and Sentiment Analysis**

* **Contextual Understanding**: Integrate models that consider the context of conversations, rather than just isolated comments, to better understand potential sarcasm, irony, or escalation patterns in abusive language.
* **Sentiment Analysis**: Add sentiment-based features to help identify negative, hostile, or aggressive tones in comments, which are common markers of cyberbullying.

**3. Integrate Natural Language Understanding (NLU) Techniques**

* **Entity Recognition**: Use NLU to detect specific entities (names, places) in comments to assess if someone is being targeted.
* **Contextual Embeddings**: Implement contextual embeddings (e.g., BERT, ELMo) to capture nuanced meanings in words and phrases, allowing the model to better handle language ambiguity.

**4. Multi-language and Cultural Adaptation**

* **Multi-language Support**: Train models on multilingual datasets to detect bullying in various languages.
* **Cultural Sensitivity**: Account for differences in language usage, slang, and context that vary between cultures, improving the model’s sensitivity to diverse user bases.

**5. Real-time Detection and Feedback Mechanism**

* **Real-time Classification**: Implement real-time processing capabilities for applications where immediate detection is required.
* **User Feedback Integration**: Allow users to provide feedback on model accuracy, enabling the model to learn from misclassifications and improve iteratively.

**6. Dataset Expansion and Annotation**

* **Larger Dataset Collection**: Gather a larger, diverse dataset that includes various forms of cyberbullying (e.g., direct abuse, exclusionary tactics, trolling).
* **Crowdsourced Annotations**: Consider crowdsourcing the annotation of data to get a broader perspective on what constitutes bullying, which can help refine the model’s understanding.

**7. Explainable AI and Transparency**

* **Explainable Models**: Implement interpretable models or techniques (e.g., SHAP, LIME) to explain why a comment was flagged as cyberbullying, providing transparency and fostering user trust.
* **Bias Mitigation**: Assess and mitigate biases in the model that might lead to over-flagging or under-flagging certain groups or types of language.

**8. Ethical and Privacy Considerations**

* **Data Privacy**: Ensure that the model complies with privacy regulations (e.g., GDPR) and respects users’ personal information.
* **Ethical Guidelines**: Develop ethical guidelines to address potential issues with misclassification and ensure that flagged comments are handled appropriately to avoid unfair repercussions.

**9. Deployment and Integration with Existing Systems**

* **API Development**: Create an API for the model to allow easy integration with social media platforms, forums, or educational platforms for content moderation.
* **Integration with Moderation Tools**: Embed the model into existing moderation workflows, enabling seamless escalation of flagged comments to human moderators when necessary.

These improvements would enhance the accuracy, fairness, adaptability, and ethical handling of the cyberbullying detection model, making it more robust and effective across different platforms and user contexts.

# APPENDIX

## Minimum Software Requirements

For a **cyberbullying comments detector** using **logistic regression**, the software requirements are minimal and focus primarily on essential Python packages and libraries for text processing and model building. Here’s a baseline setup:

**1. Python**

* **Version**: Python 3.7 or later
* Python provides a rich ecosystem of libraries that support data preprocessing, machine learning, and natural language processing.

**2. Python Libraries**

* **Data Manipulation and Processing**
  + pandas: For data loading, manipulation, and cleaning.
  + numpy: For numerical operations and handling arrays.
* **Text Processing**
  + nltk or spaCy: For tokenization, stop word removal, and other natural language processing tasks.
  + re (Python’s regex library): Useful for basic text cleaning tasks.
* **Machine Learning**
  + scikit-learn: For building the logistic regression model, as well as utilities for splitting the dataset, feature extraction, and evaluation metrics.
* **Vectorization**
  + scikit-learn also provides tools like CountVectorizer or TfidfVectorizer to convert text data into numerical features.

**3. Integrated Development Environment (IDE)**

* **Jupyter Notebook** or **Google Colab**: For interactive coding and testing.
* **Alternatively**: Any Python-compatible IDE, such as PyCharm, Visual Studio Code, or Anaconda, will work.

**4. Optional Tools for Deployment**

* **Flask** or **FastAPI**: If you plan to deploy the model as a web service API.
* **Docker**: For containerization, which helps in creating an isolated environment for the application.

**5. Operating System Compatibility**

* Works on **Windows, macOS, or Linux**.

This minimal software setup is suitable for building, training, and testing a logistic regression-based cyberbullying detection model. These tools provide everything you need for data preprocessing, model building, and evaluation.

* Pickel:

Used for serialization and deserialization of Python objects, particularly for saving and loading machine learning models.

* Colab Notebook:

If using Colab notebooks for development and testing, having Colab installed is recommended.

* Other Dependencies:

Additional libraries and modules specified in the project code, including Nearest Neighbors, Scipy's sparse matrix (csr\_matrix), and other utility modules.

## Minimum Hardware Requirements

The minimum hardware requirements for a **cyberbullying comments detector** project using **logistic regression** will generally be modest compared to more complex deep learning models. Here’s a baseline setup to handle data processing, training, and inference:

**1. Processor (CPU)**

* **Recommended**: Intel i5 or AMD Ryzen 5 or higher
* Logistic regression is not particularly computationally demanding, so a mid-range CPU should suffice for training and testing.

**2. Memory (RAM)**

* **Minimum**: 8 GB
* **Recommended**: 16 GB
* Text preprocessing (e.g., tokenization, vectorization) can be memory-intensive, so 16 GB of RAM is ideal for smoother handling of data, particularly with larger datasets.

**3. Storage**

* **Minimum**: 10 GB of free space
* **Recommended**: SSD with 20+ GB free
* Space for the dataset, processed data, and model files is essential. An SSD will help with faster read/write operations, speeding up data loading and preprocessing.

**4. Graphics Processing Unit (GPU)**

* **Not necessary** for logistic regression
* Logistic regression can be efficiently run on a CPU. A GPU is only needed if you experiment with deep learning models later on.

**5. Operating System**

* **Any OS** that supports Python and common ML libraries (e.g., Windows 10+, macOS, or Linux).

**6. Software Requirements**

* Python 3.7+ and libraries like scikit-learn, pandas, numpy, and nltk or spaCy for text processing.

This setup should be adequate for experimenting with various text preprocessing techniques and building a logistic regression model for cyberbullying detection.

# REFERENCES

1. Gupta, Meenu, et al. "Movie recommender system using collaborative filtering." *2020 international conference on electronics and sustainable communication systems (ICESC)*. IEEE, 2020.
2. Raghavendra, C. K., and K. C. Srikantaiah. "Similarity based collaborative filtering model for Movie Recommendation Systems." *2021 5th international conference on intelligent computing and control systems (ICICCS)*. IEEE, 2021.
3. Anwar, Taushif, and V. Uma. "Comparative study of recommender system approaches and movie recommendation using collaborative filtering." *International Journal of System Assurance Engineering and Management* 12 (2021): 426-436.
4. [https://ww](http://www.kaggle.com/code/gpreda/user-based-collaborative-filtering-using-knn)w.kagg[le.com/code/gpreda/user](http://www.kaggle.com/code/gpreda/user-based-collaborative-filtering-using-knn)-[based](http://www.kaggle.com/code/gpreda/user-based-collaborative-filtering-using-knn)-[collaborative-filtering-using-knn](http://www.kaggle.com/code/gpreda/user-based-collaborative-filtering-using-knn)
5. [https://ww](http://www.kaggle.com/code/gpreda/item-based-collaborative-filtering-using-knn)w.kagg[le.com/code/gpreda/item](http://www.kaggle.com/code/gpreda/item-based-collaborative-filtering-using-knn)-[based](http://www.kaggle.com/code/gpreda/item-based-collaborative-filtering-using-knn)-[collaborative-filtering-using-knn](http://www.kaggle.com/code/gpreda/item-based-collaborative-filtering-using-knn)
6. https://bootstrapmade.com/tempo-free-onepage-bootstrap-theme/