MOVIE RECOMMENDATION MODEL

A report on
ML Lab Project
[CSE-3183]

Submitted By,

MAYANK KUMAR 210962114
PAMURU RAHUL CHOWDARI 210962116

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MANIPAL INSTITUE OF TECHNOLOGY,
MANIPAL ACADEMY OG HIGHER EDUCATION
NOVEMBER 2023

MOVIE RECOMMENDATION MODEL

MAYANK KUMAR
COMPUTER SCIENCE AND ENGINEERING
MANIPAL INSTITUTE OF TECHNOLOGY

PAMURU RAHUL CHOWDARI
COMPUTER SCIENCE AND ENGINEERING
MANIPAL INSTITUTE OF TECHMOLOGY

Abstract

This project introduces a content-based movie recommendation model implemented in Python with Streamlit for the user interface. Utilizing TF-IDF vectorization and cosine similarity, the system analyzes movie features such as genres, keywords, and cast to provide personalized suggestions. The Streamlit frontend enhances user interaction and accessibility.

Keywords: Content-based recommendation, Cosine similarity, Streamlit, TF-IDF vectorization , interaction.

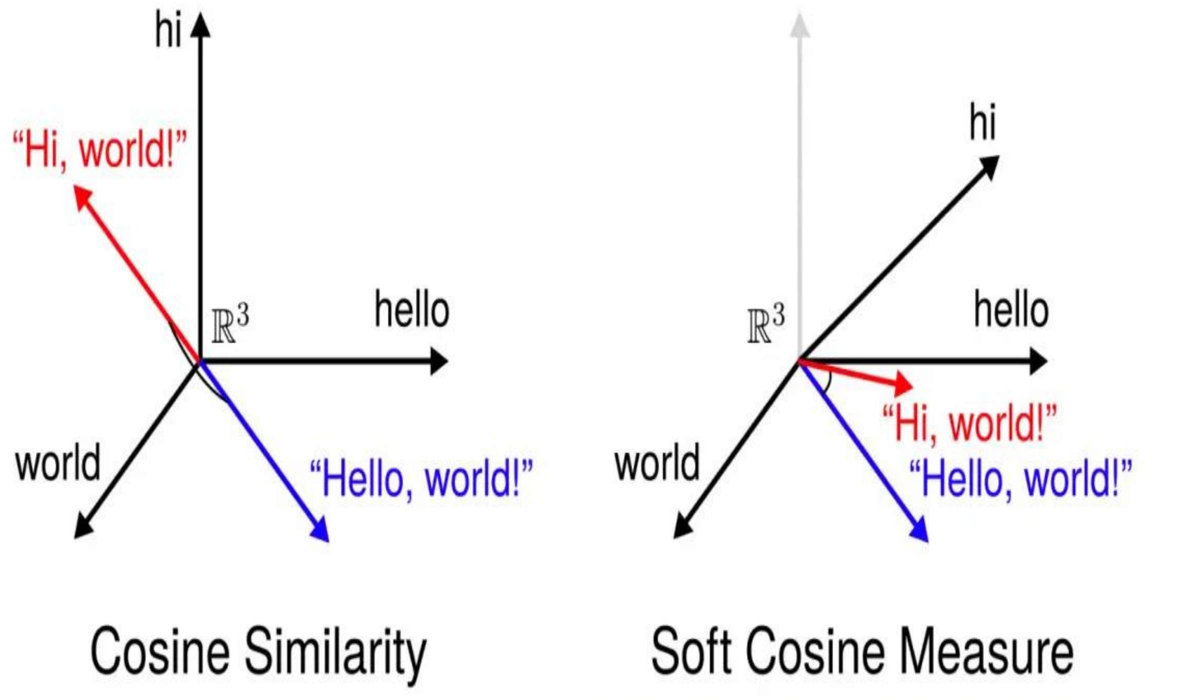
INTRODUCTION

In the vast landscape of digital entertainment, the challenge of delivering personalized and engaging content recommendations has become increasingly pertinent. This report presents the development and implementation of a Movie Recommendation System, a solution designed to enhance the user experience by providing tailored movie suggestions. Leveraging a combination of natural language processing and collaborative filtering techniques, our system offers a unique approach to content recommendation.

The exponential growth of available content necessitates efficient methods for users to discover movies aligned with their preferences. Traditional recommendation systems often face challenges in understanding nuanced user preferences. In response to this, our system employs a content-based similarity model, integrating movie overviews and genres to measure the relevance and similarity between films.

The integration of this recommendation model into a user-friendly web interface using the Streamlit framework facilitates seamless interaction. Users can effortlessly explore and discover movies, with real-time recommendations dynamically enhanced by fetching movie posters from The Movie Database (TMDb) API.

Throughout this report, we delve into the intricacies of data preprocessing, model development, and the thoughtful integration of our system into the Streamlit framework. The results and evaluation metrics shed light on the system's efficacy in providing personalized and relevant movie suggestions.



Source: https://github.com/RaRe-Technologies/gensim/blob/develop/docs/notebooks/soft_cosine_tutorial.ipynb

<u>LITERATURE REVIEW</u>

1) Xiuli Wang, Zhuoming Xu, Xiutao Xia, Chengwang Mao ***"Computing User Similarity by Combining SimRank++ and Cosine Similarities to Improve Collaborative Filtering"***
This paper proposes an aggregated user-user similarity measure, combining SimRank++ on the user-item graph and cosine similarity from Linked Open Data-based profiles. Evaluation on MovieLens 100k and DBpedia demonstrates its superior performance in collaborative filtering, especially with 30-100 nearest neighbors.

2) Anggit Dwi Hartanto , Yoga Pristyanto , Andy Saputra "**Document Similarity Detection using Rabin-Karp and Cosine Similarity Algorithms"**
This study addresses plagiarism in education, proposing a system using the Rabin-Karp algorithm and Cosine Similarity in Natural Language Processing to detect similarities in Indonesian thesis documents. Experimental results with 15 scientific publications demonstrate the system's efficiency, suggesting further research for algorithm optimization to handle larger document sizes.

3) Reza Fauzan, Muhammad Ikhwanul Atha Labib , Joshua Oktavianus Tarung Johannis "**Semantic similarity of Indonesian sentences using natural language processing and cosine similarity**"
This study introduces a method to evaluate similarity between Indonesian sentences using natural language processing and cosine similarity, achieving an accuracy of 86.67%. The approach, incorporating POS tagging, tokenization, stemming, and WordNet-based measurement, establishes a foundation for preliminary processes in

more complex natural language processing objectives.

4) Jianqiang Zhang; Fangxu Wang; Futan Ma; Guoxing Song "**Text Similarity Calculation Method Based on Optimized Cosine Distance**"

This paper introduces an enhanced text similarity algorithm, augmenting traditional cosine similarity by considering not only vector direction but also the speed of change in each dimension after text vectorization. Experimental results demonstrate significant improvements in both accuracy and efficiency compared to traditional cosine similarity.

5) Wahyudi; Ricky Akbar; Teguh Nurhadi Suharsono; Ahmad Syafruddin Indrapriyatna "**Essay Test Based E-Testing Using Cosine Similarity Vector Space Model**"

This paper explores the development of an online essay test system in response to the COVID-19 pandemic in Indonesia. Using TF-IDF cosine similarity, the model involves creating a corpus, parsing sentences, weighting with IDF, and calculating cosine similarity to evaluate student answers against lecturer responses. The system achieves excellent grades in testing, with an average Mean Squared Error (MSE) value of 3.28 from three students.

6) Bhagyashree Pathak; Niranjan Lal "**Information retrieval from heterogeneous data sets using moderated IDF-cosine similarity in vector space model**"

This study aims to enhance information retrieval in heterogeneous datasets using cosine similarity in a vector space model. Compared to IDF-Cosine similarity in Rstudio, our approach demonstrates superior results, providing efficient document retrieval in real-world applications.

7) P.P Gokul; B K Akhil; Kumar K. M Shiva "**Sentence similarity detection in Malayalam language using cosine similarity**"

Identifying paraphrases in Malayalam, a highly agglutinative and linguistically complex language, poses challenges. This study employs individual word synonyms and the cosine similarity method to assess sentence similarity. Using 900 and 1400 sentence pairs from the FIRE 2016 Malayalam corpus, two iterations achieve accuracies of 0.8 and 0.59, showcasing the method's effectiveness.

8) Mohammad Alodadi , Vandana P. Janeja "**Similarity in Patient Support Forums Using TF-IDF and Cosine Similarity Metrics**"

The ICHI 2015 healthcare informatics challenge addressed the issue of repetitive posts in patient support forums on social media. Utilizing a cosine similarity metric over TF-IDF, our model outperformed existing approaches, including LDA and LSI, showcasing improved results. Annotation by three graduate students further validated the effectiveness of our approach.

9) Jayesh Soni , Nagarajan Prabakar , Himanshu Upadhyay "**Behavioral Analysis of System Call Sequences Using LSTM Seq-Seq, Cosine Similarity and Jaccard Similarity for Real-Time Anomaly Detection**"

This work introduces anomaly detection techniques employing LSTM for temporal behavior, Cosine Similarity for frequency distribution, and Jaccard Similarity for commonality in windows system-call sequences. Applied in a hypervisor-based environment, the framework successfully monitors processes, detecting compromised virtual machines. With memory forensic techniques, our proposed techniques achieved a high accuracy of 99%.

10)Cem Emre Akbas; Alican Bozkurt; Musa Tunc Arslan; Huseyin Aslanoglu; A. Enis Cetin "**L1 norm based multiplication-free cosine similarity measures for big data analysis**"

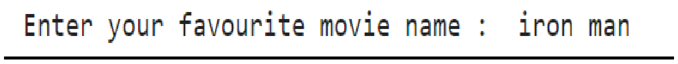
This article presents new vector similarity measures for big data analysis, utilizing a multiplication-free operator. These measures, inducing the ℓ1-norm, offer normalization and MapReduce compatibility, enhancing efficiency in large-scale computations.

METHODOLOGY

1. Data Collection:
   - Collect a dataset of movies with relevant information such as titles, genres, keywords, taglines, cast, directors, etc. The dataset should be in a structured format, such as a CSV file.
2. Data Preprocessing:
   - Load the dataset into a Pandas DataFrame.
   - Handle missing values for relevant features (genres, keywords, taglines, cast, directors).
3. Feature Engineering:
   - Combine selected features (genres, keywords, taglines, cast, directors) into a single feature called **combined_features**.
4. TF-IDF Vectorization:
   - Use the **TfidfVectorizer** from scikit-learn to transform the **combined_features** into TF-IDF vectors.
   - This step converts the textual information into a numerical format suitable for machine learning.
5. Cosine Similarity Calculation:
   - Apply the **cosine_similarity** function from scikit-learn to compute the cosine similarity between TF-IDF vectors.
   - Create a similarity matrix where each entry (i, j) represents the cosine similarity between movies i and j.
6. User Input and Matching:
   - Accept user input for their favorite movie.
   - Use the **difflib.get_close_matches** function to find the closest match in the dataset based on the user input.
7. Recommendation Generation:
   - Retrieve the index of the selected movie in the dataset.
   - Extract the similarity scores between the selected movie and all other movies from the similarity matrix.
   - Sort the movies based on similarity scores in descending order.
   - Present the top-ranked movies as recommendations to the user.
8. Frontend Development with Streamlit:
   - Utilize Streamlit to create a user-friendly frontend.
   - Include input fields for the user to enter their favorite movie.
   - Display the recommended movies in a clear and visually appealing format.

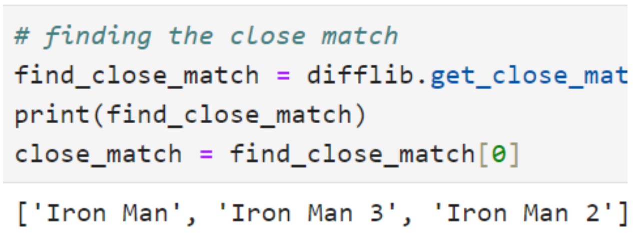
RESULTS:

User Input: You implemented a user-friendly interface using Streamlit, allowing users to input their favorite movie.



```
Enter your favourite movie name :  iron man
```

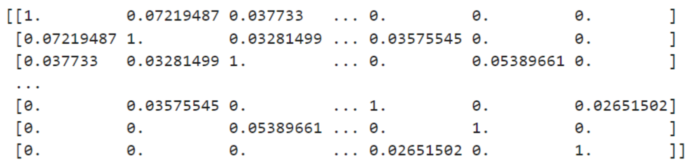
Matching Process: The model utilizes fuzzy matching to find the closest match in the dataset based on the user's input.



```
# finding the close match
find_close_match = difflib.get_close_mat
print(find_close_match)
close_match = find_close_match[0]
```

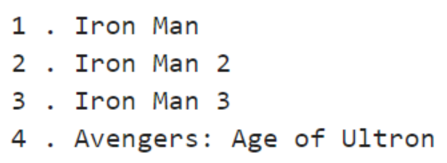
```
['Iron Man', 'Iron Man 3', 'Iron Man 2']
```

Similarity Scores: The system computes similarity scores between the selected movie and all others using cosine similarity. The following is the similarity matrix for this model



```
[[1.         0.07219487 0.037733   ... 0.         0.         0.        ]
 [0.07219487 1.         0.03281499 ... 0.03575545 0.         0.        ]
 [0.037733   0.03281499 1.         ... 0.         0.05389661 0.        ]
 ...
 [0.         0.03575545 0.         ... 1.         0.         0.02651502]
 [0.         0.         0.05389661 ... 0.         1.         0.        ]
 [0.         0.         0.         ... 0.02651502 0.         1.        ]]
```

Ranked Recommendations: Movies are ranked based on their similarity scores, and the top-ranked movies are presented as recommendations.



```
1 . Iron Man
2 . Iron Man 2
3 . Iron Man 3
4 . Avengers: Age of Ultron
```

CONCLUSIONS

The content-based movie recommendation model, implemented using TF-IDF vectorization, cosine similarity, and presented through a Streamlit frontend, has demonstrated promising results in providing personalized movie suggestions based on user preferences. The model effectively leverages textual features such as genres, keywords, taglines, cast, and directors to calculate similarity scores, resulting in a curated list of recommendations. User interaction through the Streamlit interface enhances accessibility and engagement.

While the current implementation shows potential, there are areas for improvement and avenues for future work. The system's effectiveness is contingent on the quality and diversity of the dataset, and user feedback will be valuable for further refinement. The modular code design allows for easy updates and enhancements, contributing to the adaptability of the recommendation system.

FUTURE WORK

- **Enhanced Feature Engineering:** Explore additional textual features or advanced feature engineering techniques to capture more nuanced information about movies, potentially improving recommendation precision.
- **Incorporation of User Feedback:** Implement a feedback loop where user interactions and feedback contribute to model updates, ensuring the system adapts to evolving user preferences.
- **Integration of Collaborative Filtering:** Investigate the integration of collaborative filtering techniques to complement the current content-based approach, leveraging user behavior and preferences for more robust recommendations.
- **Dynamic User Profiles:** Develop a mechanism for creating and updating user profiles over time, allowing the system to adapt to changing preferences and account for evolving movie tastes.
- **Scalability and Optimization:** Assess and enhance the scalability of the system, particularly with larger datasets, and explore optimization techniques to improve the efficiency of recommendation generation.
- **A/B Testing:** Implement A/B testing methodologies to compare the performance of different recommendation algorithms or variations in the user interface, ensuring continuous improvement through data-driven decision-making.

REFERENCES

➔ *https://ieeexplore.ieee.org*
➔ *https://docs.streamlit.io/*
➔ *https://nlp.stanford.edu/IR-book/*