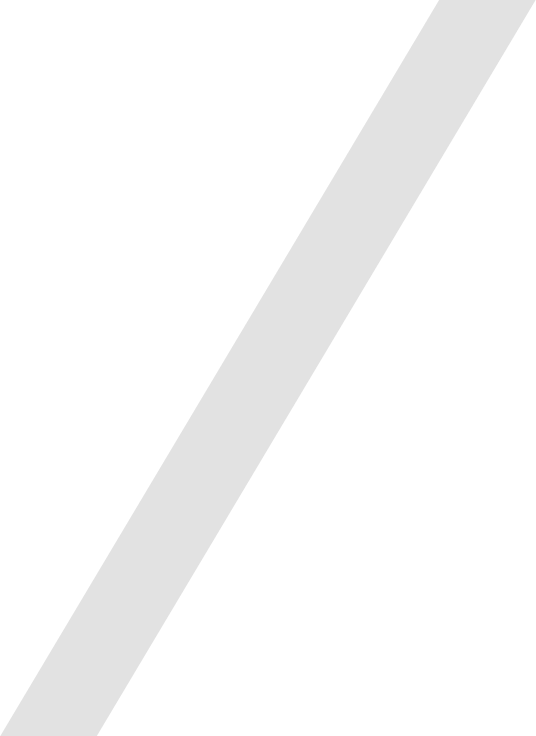
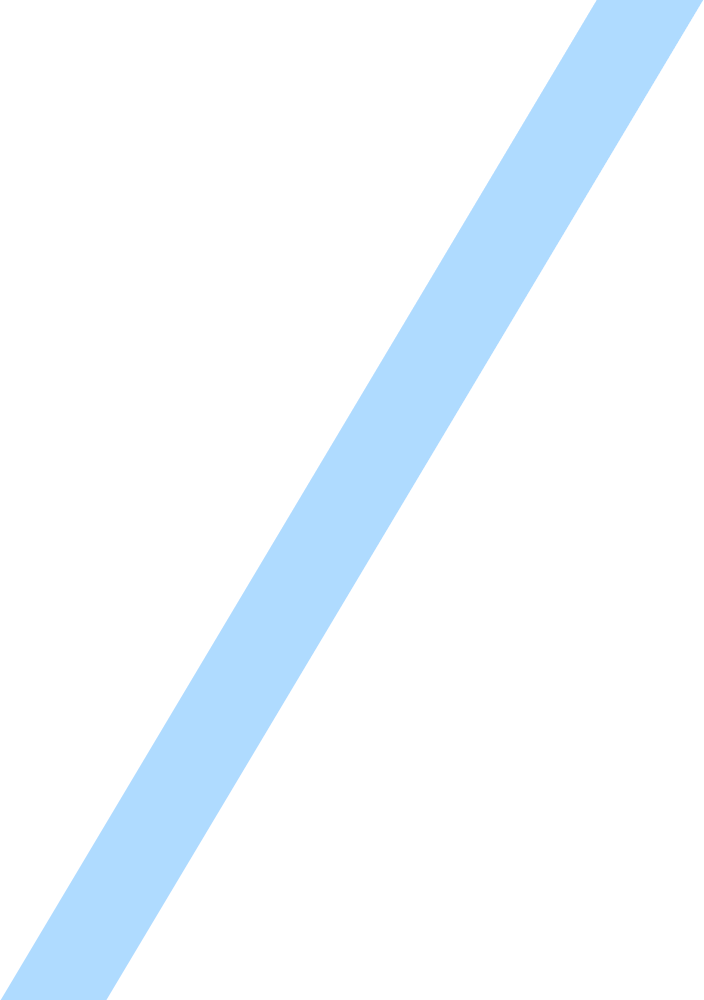
|  |
| --- |
| TECHNICAL REPORT |

|  |
| --- |
| Electrical & Computer Engineering & Computer Science (ECECS) |

|  |  |
| --- | --- |
| SPRING 22 |  |



Contents

[Executive Summary 2](#_Toc96341550)

[Technical Report 3](#_Toc96341551)

[Highlights of Project 3](file:///C:\Users\ardia\Desktop\TEACHING\SPRING%2022\DSCI6002%20Data%20Science\Projects\Technical%20report%20Template.docx#_Toc96341552)

[Submitted on: 3](file:///C:\Users\ardia\Desktop\TEACHING\SPRING%2022\DSCI6002%20Data%20Science\Projects\Technical%20report%20Template.docx#_Toc96341553)

[Abstract 4](#_Toc96341554)

[Methodology 5](#_Toc96341555)

[Results Section 5](#_Toc96341556)

[Discussion 6](#_Toc96341557)

[Conclusion 6](#_Toc96341558)

[Contributions/References 7](#_Toc96341559)

|  |
| --- |
| Netflix MovieRecommendation |

|  |  |  |
| --- | --- | --- |
| Executive Summary Netflix's aim is to connect people with the movies they enjoy. The goal of this project is to create a movie recommendation system for Netflix. Here, we will learn from data and suggest the greatest TV shows to users based on their own and others' actions. | | |
| person at a table writing in a notebook with people around | | |
| **Group 11:**  **Sravani Burra**  **Sai Charan Jana**  **Sai Pavan Kumar Gopularam**  **Git Link -**[**https://github.com/saipavan25/Movie\_Recomender\_system/tree/main**](https://github.com/saipavan25/Movie_Recomender_system/tree/main) | **Questions?**  Contact: Sravani Burra |  |

|  |
| --- |
| Technical Report |

|  |  |
| --- | --- |
| **Netflix Movie Recommendation System** |  |
| Highlights of Project-To be able to recommend a movie to the user based on the watch history of the userSubmitted on: 05-03-2022 |

## 

## Abstract:

On Netflix, there are a variety of movies, TV episodes, and documentaries to choose from, making it difficult to limit down your options. This is when our recommendation system comes into play, which could be directing us to some movies based on the previous viewing habits.

We consider users and movies in any recommendation system. A recommendation system matches content based on similarities between a group of users and a group of movies.

Collaborative Filtering is based on user ratings, and it will suggest movies that we haven't seen but that users who are similar to us have and enjoy. This filter evaluates the movies both users watched and how they rated them in order to assess whether or not they are similar. This type of algorithm will basically forecast the rate of a movie for a person who hasn't seen it yet based on the rates of comparable users by looking at the objects in common.

[Here you can find a few useful tips on coming up with a great pitch](https://www.youtube.com/watch?v=n3RKsY2H-NE)

Cover Page:

Name of the institutional publisher – IEEE

Title of the report – Movie Recommendation System Using Cosine similarity and sentimental Analysis

Names of authors- Harsh Khatter, Nishtha Goel, Naina Gupta

Date of Publication - 01 October 2021

[Link to IEEE Paper](https://ieeexplore.ieee.org/abstract/document/9544794)

Introductory Section:

The goal of this project is to create a movie recommendation system for Netflix. The data I utilized for this analysis came from Netflix. It is made up of two comma separated files or CSV files.

Movies.csv

Ratings.csv

Review of available research:

## Multimedia is considered as one of the best sources of entertainment. People of all age groups love to watch movies. Movie Recommender System is essential in our social lives as it enhances the field of entertainment.

The proposed machine learning model is trained, tested, and a sentiment classifier is generated which classify the sentiments as a good or a bad sentiment. The recommender system is generated by applying Cosine similarity and making API Calls. As a result, the live working of the system generates accurate and personalized recommendations along with the analysis of sentiments for the end users. It is also concluded that Cosine Similarity provides better and efficient results for a recommender system.

## 

## Methodology:

* **Data loading**

Here, we have two data files

* Movies
* Ratings

There are three columns in the movies.csv dataset

* MovieId: the movie's unique identifier.
* title of the film
* genres: movies genres

Table

Description automatically generated

There are four columns in the ratings.csv file:

* userId: the unique identifier for the user who rated the film.
* Id: the movie's unique identifier.
* user ratings: each user's rating (from 0 to 5)
* Timestamp: The time the movie was rated.

Table

Description automatically generated

* **Data viewing:**

Here we will have a look at how the data spread:

Table

Description automatically generated

Here we can see that the number of movies considered in this dataset are 10329.

Table

Description automatically generated

Here we can see that the average rating for all the movies is 3.5%

* **Data cleaning**

Here, we have cleaned the noise from data.

* **Data Merging:**

Now that we have two datasets it is difficult to work with both the datasets so let’s merge both the datasets on the basis of using Movie Id as a primary key for both of them.

Graphical user interface, text, application, email

Description automatically generated

* **Data mapping:**

Mapping title and rating of the movie.

Table

Description automatically generated

Chart

Description automatically generated

Mapping Title and Rating on the basis of count:

Table

Description automatically generated

Chart

Description automatically generated

## 

## Data Handling:

All the data that is being used or generated in this entire process is stored in Buckets. These programs don’t run in our local machine but on the cloud side.

## Web Application:

Flask is used to deploy the whole running system and convert into a web application by pickling it. The web application looks as:

Text, letter

Description automatically generated

## Results Section:

Here we can see that when the movie name Toy Story was given our recommender system recommends all the movies that are of same genre and with good ratings.

Text

Description automatically generated

Here is another example with the movie name “Jumanji”.

Text

Description automatically generated

## Discussion:

Graphical user interface, text, application

Description automatically generated

The Key functionalities that are implementing the Recomenditions are:

Cosine Similarity

TF-IDF Vectoriser.

## Cosine Similarity:

Cosine similarity estimates the likeness between two vectors of an internal item space. It is estimated by the cosine of the point between two vectors and decides if two vectors are pointing in generally a similar bearing. Estimating archive likeness in text analysis is frequently utilized.

## TF-IDF Vectorizer:

Term frequency-inverse document frequency is a text vectorizer that transforms the text into a usable vector. It combines 2 concepts, Term Frequency (TF) and Document Frequency (DF). The term frequency is the number of occurrences of a specific term in a document

.

## Conclusion:

Netflix might meet their customers' needs by gathering information from their watch lists.  Finally, Netflix is encouraged to gain a competitive advantage by closely monitoring their competitors' behavior they give to their customers, before taking their next steps. All these suggestions would help Netflix have a more promising future in the video streaming business.