MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Skill Based Mini Project Report

on

"Netflix movie recommendation system"

Submitted By:

Ram Shrivastava

0901CS201096

Faculty Mentor:

Dr. Ranjeet Kumar Singh, Assistant Professor, CSE

Submitted to:

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR - 474005 (MP) est. 1957

JAN-JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

CERTIFICATE

This is certified that **Ram Shrivastava** (0901CS201096) has submitted the project report titled **Netflix movie recommendation system** under the mentorship of **Dr. Ranjeet Kumar Singh**, in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering from Madhav Institute of Technology and Science, Gwalior.

R.K.Singh

Dr. Ranjeet Kumar Singh
Faculty Mentor
Assistant Professor
Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Dr. Ranjeet Kumar Singh**, **Assistant Professor**, Department of CSE

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

[illegible]

Ram Shrivastava
0901CS201096
2nd Year,
Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Department of Computer Science and Engineering, for allowing** me to explore this project. I humbly thank **Dr. Manish Dixit**, Professor and Head, Department of Computer Science and Engineering, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors. I am grateful to the guidance of **Dr. Ranjeet Kumar Singh**, Assistant Professor, Department of CSE, for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

[illegible]

Ram Shrivastava
0901CS201096
2nd Year,
Computer Science and Engineering

ABSTRACT

Providing a useful suggestion of products to online users to increase their consumption on websites is the goal of many companies nowadays People usually select or purchase a new product based on some friend's recommendations, comparison of similar products or feedbacks from other users. In order to do all these tasks automatically, a recommender system must be implemented. The recommender systems are tools that provide suggestions that best suit the client's needs, even when they are not aware of it. That offers of personalized content are based on past behaviour and it hooks the customer to keep coming back to the website. In this report, a movie recommendation mechanism within Netflix will be built.

The main types of recommender algorithm are Titles, Cast, Directors and Description. All of them will be introduced in this report. We will select the algorithms that best fit to the data and we will implement them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

Symbol	Description
Σ	Used for summation

Chapter 1: INTRODUCTION

1.1 About recommendation system

Netflix is a company that handles a big collection of television programs and movies, by streaming it at any time via online (computers or TV). This firm is profitable because the users do a monthly payment to get access to the platform. However, the clients can cancel their subscriptions at any time . Therefore, it is vital for the business to keep the users hooked to the platform and not to lose their interest. This is where recommendation systems start to play an important role, it is pivotal to provide valuable suggestions to users. The recommendation systems are increasing their popularity among the service providers, because they help to increase the number of items sold, offer a diverse selection of items, the user satisfaction increases, as well as the user fidelity to the company, and they are quite helpful to have a better understanding of what the user wants. Then, it is easier to lead the user to make better decisions from a wide variety of cinematographic products.

1.2 Parameters considered for recommendations

The recommender systems take into account not only information about the users but also about the items they consume; and so on and so forth. Nevertheless, there are many algorithms available to perform a recommendation system. For instance,

(i) Titles, where only the similar titles are recommended
(ii) Cast, which recommends the next show or movie having similar cast
(iii) Directors, sometimes users like same direction or obsessed with directors so it recommends on the basis of similar direction styles;
(iv) Description, the recommendation of items with similar information the user has liked or used in the past.

Selecting the algorithm that fits better the analysis is not an easy task, and neither expands the user's taste into neighbouring areas by improving the obvious.

This study is conducted on real data from the Netflix users and they have given to the movies they have seen. Furthermore, the movies file includes the year of release and the title of the movie as well.

1.3 Objective of project

Recommendation systems and link prediction are classic problems when a wealth of content is provided to users and one needs to know what content users have not yet seen but are most likely to enjoy. They are some of the most common problems seen in to- day's world. Recommender systems are used to suggest friends on Facebook, products you might like on Amazon, the next video on YouTube, more shows on Netflix, etc.

Our project is to build a recommender system for users based on the Netflix network, which is a bipartite graph consisting of users and movies, with edges between movies. Specifically, we will try to recommend movies a user has not seen that they may enjoy, based on the previous movies they saw. We plan to use the methods from the papers we surveyed as a baseline and improve upon them, by taking advantage of the unique structures of our network.

Chapter 2: TECHNOLOGIES USED

2.1 Tools used

- Python
- Scikit learn
- Jupyer Notebook
- VS Code
- Flask
- HTML and CSS

2.2 Hardware and software used

As our project is totally web based and based on machine learning so there is no external hardware is used in this.

This web app needs to be presented to the user in such a way that it should be attractive and interactive. So, we used the various frontend technologies such as HTML, Bootstrap (CSS framework – which provide premade elements to our website) and Flask framework for the backend work. Mainly we used python language for coding and csv file for our dataset.

For coding we have used the Visual Studio Code editor and Jupyter notebook for python.

Chapter 3: Methodology

3.1 About dataset

In this dataset, we have 6235 movies and TV Shows in it till 2019 which are on Netflix. It contains their titles with the year of their releasing and cast, directors and the description.

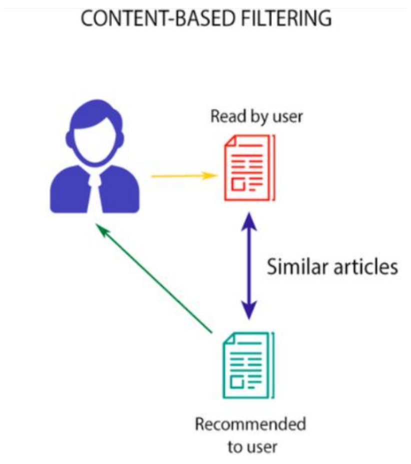
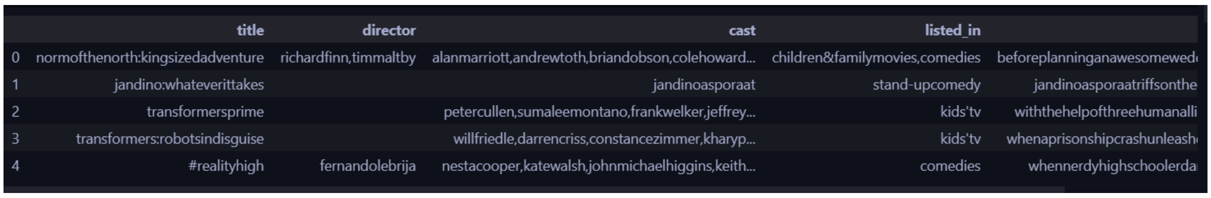


Fig. Content Based Filtering

After cleaning the data, top 5 content of our dataset:



	title	director	cast	listed_in	
0	normofthenorth:kingsizedadventure	richardfinn,timmaltby	alanmarriott,andrewtoth,briandobson,colehoward...	children&familymovies,comedies	beforeplanninganawesomewed
1	jandino:whateverittakes		jandinoasporaat	stand-upcomedy	jandinoasporaatriffsonthe
2	transformersprime		petercullen,sumaleemontano,frankwelker,jeffrey...	kids'tv	withthehelpofthreehumanalli
3	transformers:robotsindisguise		willfriedle,darrencriss,constancezimmer,kharyp...	kids'tv	whenaprisonshipcrashunleash
4	#realityhigh	fernandolebrija	nestacooper,katewalsh,johnmichaelhiggins,keith...	comedies	whennerdyhighschoolerda

Fig. After cleaning the dataset, head

Our movie recommendation engine works by suggesting movies to the user based on the metadata information. The similarity between the movies is calculated and then used to make recommendations. For that, our text data should be preprocessed and converted into a vectorizer using the CountVectorizer. As the name suggests, CountVectorizer counts the frequency of each word and outputs a 2D vector containing frequencies.

We don’t take into account the words like a, an, the (these are called “stopwords”) because these words are usually present in higher amounts in the text and don’t make any sense.

There exist several similarity score functions such as cosine similarity, Pearson correlation coefficient, etc. Here, we use the cosine similarity score as this is just the dot product of the vector output by the CountVectorizer.

3.2 Cosine similarity function

Cosine similarity among two objects measures the angle of cosine between the two objects. It compares two documents on a normalized scale. It can be done by finding the dot product between the two identities.

Lesser the angle between the two vectors more is the similarity. It means if the angle between two vectors is small, they are almost alike each other and if the angle between the two vectors is large then the vectors are very different from each other.

As the below diagram shows, the angle between v1 and v2 is. Lesser the angle between the two vectors more is the similarity. It means if the angle between two vectors is small, they are almost alike each other and if the angle between the two vectors is large then the vectors are very different from each other.

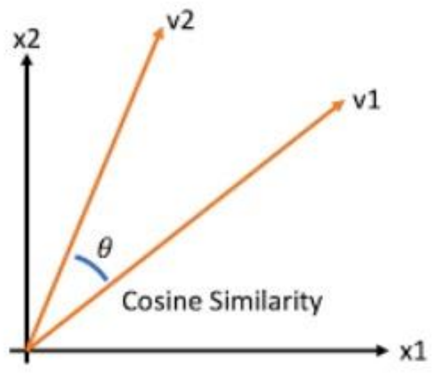


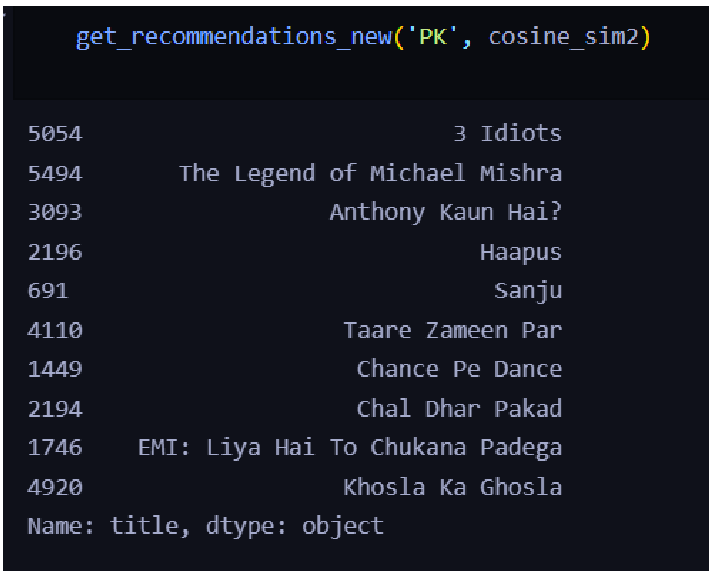
Fig. Cosine similarity

3.3 get_recommendations_new() function

The get_recommendations_new() function takes the title of the movie and the similarity function as input. It follows the below steps to make recommendations.

- Get the index of the movie using the title.
- Get the list of similarity scores of the movies concerning all the movies.
- Enumerate them (create tuples) with the first element being the index and the second element is the cosine similarity score.
- Sort the list of tuples in descending order based on the similarity score.
- Get the list of the indices of the top 10 movies from the above sorted list. Exclude the first element because it is the title itself.
- Map those indices to their respective titles and return the movies list.

Create a function that takes in the movie title and the cosine similarity score as input and outputs the top 10 movies similar to it.



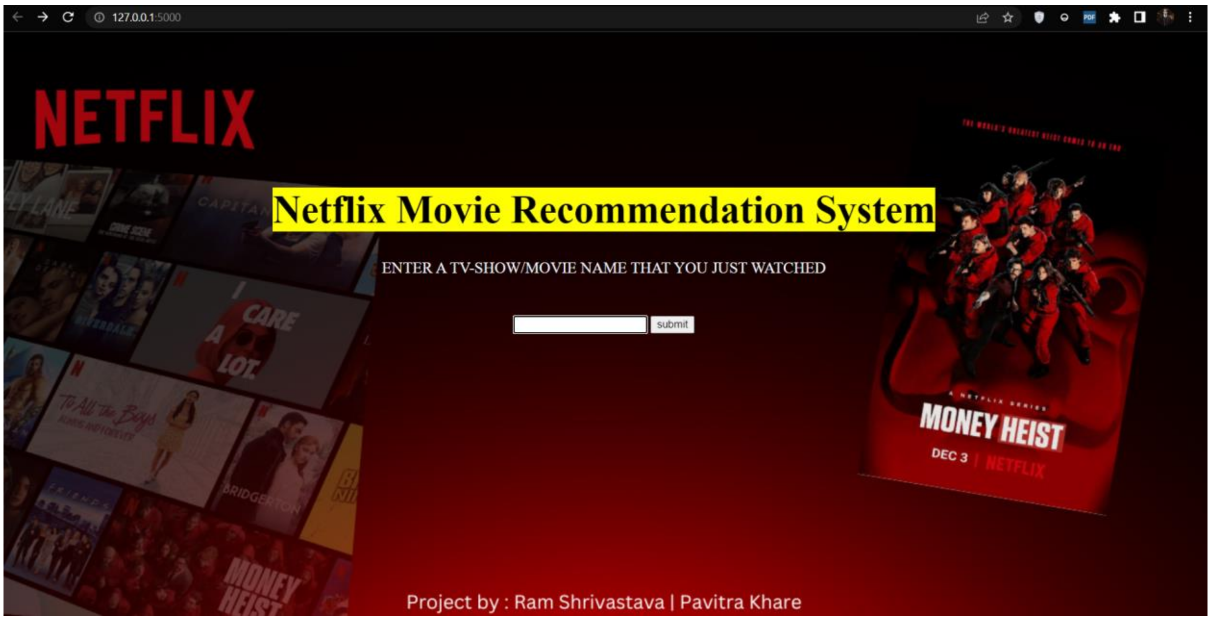
```
get_recommendations_new('PK', cosine_sim2)
```

```
5054                          3 Idiots
5494         The Legend of Michael Mishra
3093                 Anthony Kaun Hai?
2196                            Haapus
691                              Sanju
4110                  Taare Zameen Par
1449                   Chance Pe Dance
2194                   Chal Dhar Pakad
1746      EMI: Liya Hai To Chukana Padega
4920                  Khosla Ka Ghosla
Name: title, dtype: object
```

Fig. get_recommendations_new() function

Chapter 4: Results and conclusions

4.1 Home page and final page



127.0.0.1:5000

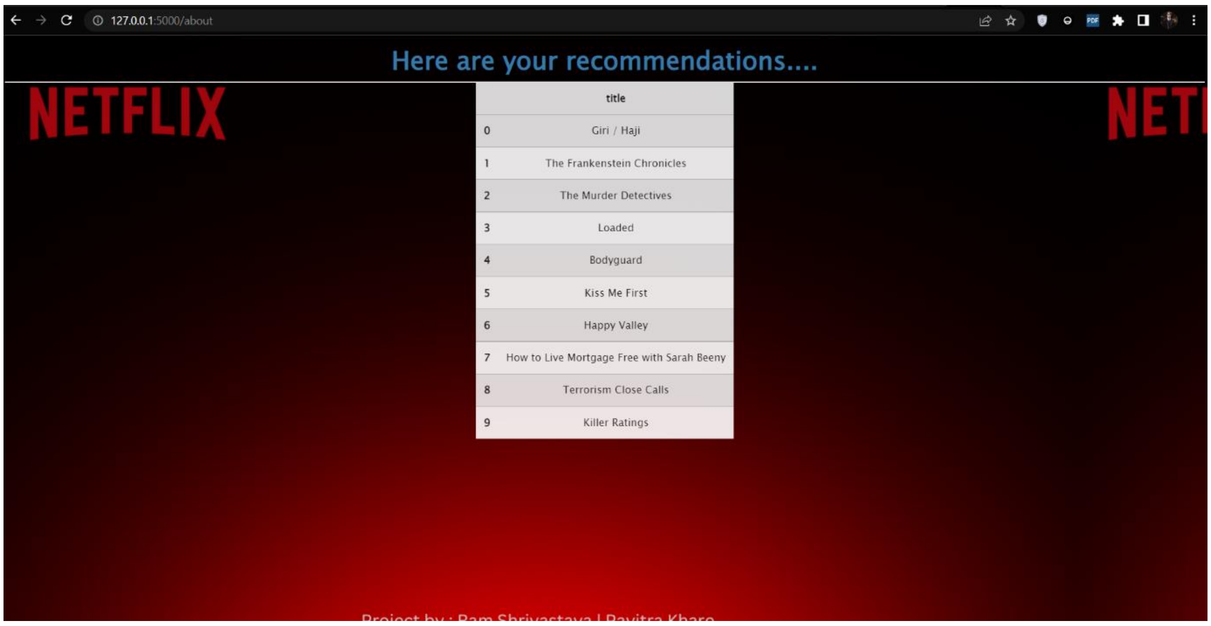
NETFLIX

Netflix Movie Recommendation System

ENTER A TV-SHOW/MOVIE NAME THAT YOU JUST WATCHED

Project by : Ram Shrivastava | Pavitra Khare

Fig. Home page



127.0.0.1:5000/about

Here are your recommendations....

	title
0	Giri / Haji
1	The Frankenstein Chronicles
2	The Murder Detectives
3	Loaded
4	Bodyguard
5	Kiss Me First
6	Happy Valley
7	How to Live Mortgage Free with Sarah Beeny
8	Terrorism Close Calls
9	Killer Ratings

Fig. Recommendations

4.2 Results and discussion

These two screenshots are of homepage and recommendations page. Here we can type any movie or TV show we just watched and can get recommendation for another similar shows.

In the deep learning framework recommendation system, we have used Cosine Similarity and Content-based filtering to predict our result and recommend a movie to the user by running the code in python using NumPy and panda libraries.

The formula used to measure how similar the movies are based on their similarities of different properties. Mathematically, it shows the cosine of the angle of two vectors projected in a multidimensional space. The cosine similarity is very beneficial since it helps in finding similar objects.

$$CosSim(x,y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

4.3 Conclusion and scope

We have illustrated the modelling of a movie recommendation system by making the use of content-based filtering in the movie recommendation system. Recommendations systems have become the most essential fount of a relevant and reliable source of information in the world of internet. Simple ones consider one or a few parameters while the more complex ones make use of more parameters to filter the results and make it more user friendly. With the inclusion of advanced deep learning and other filtering techniques like collaborative filtering and hybrid filtering a strong movie recommendation system can be built. This can be a major step towards the further development of this model as it will not only become more efficient to use but also increase the business value even further.

References

1. https://www.kaggle.com/
2. https://www.netflix.com/
3. https://github.com/

Appendix

```
import pandas as pd
from flask import Flask, render_template,request
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

def clean_data(x):
    return str.lower(x.replace(" ", ""))

def create_soup(x):
    return x['title']+ ' ' + x['director'] + ' ' + x['cast'] + ' ' +x['listed_in']+' '+ x['description']

def get_recommendations(title, cosine_sim):
    global result
    title=title.replace(' ','').lower()
    idx = indices[title]
    sim_scores = list(enumerate(cosine_sim[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:11]
    movie_indices = [i[0] for i in sim_scores]
    result =  netflix_overall['title'].iloc[movie_indices]
    result = result.to_frame()
    result = result.reset_index()
    del result['index']
    return result

netflix_overall = pd.read_csv('netflix_titles.csv')
netflix_data = pd.read_csv('netflix_titles.csv')
netflix_data = netflix_data.fillna('')
```

```
new_features = ['title', 'director', 'cast', 'listed_in', 'description']
netflix_data = netflix_data[new_features]
for new_features in new_features:
    netflix_data[new_features] = netflix_data[new_features].apply(clean_data)
netflix_data['soup'] = netflix_data.apply(create_soup, axis=1)
count = CountVectorizer(stop_words='english')
count_matrix = count.fit_transform(netflix_data['soup'])
global cosine_sim2
cosine_sim2 = cosine_similarity(count_matrix, count_matrix)
netflix_data=netflix_data.reset_index()
indices = pd.Series(netflix_data.index, index=netflix_data['title'])
#get_recommendations('PK', cosine_sim2)



app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/about',methods=['POST'])
def getvalue():
    moviename = request.form['moviename']
    get_recommendations(moviename,cosine_sim2)
    df=result
    return render_template('result.html',  tables=[df.to_html(classes='data')], titles=df.columns.values)

if __name__ == '__main__':
    app.run(debug=False)
```