SRI DHARMASTHALA MANJUNATHESHWARA COLLEGE
(AUTONOMOUS)
UJIRE - 574240

DEPARTMENT OF STATISTICS

CERTIFICATE

Certified that this is the bonafide record of project work done by Mr. Arvind N Myageri during the year 2020 as a part of her M.Sc (Statistic) fourth semester course work.

Reg. No.

1	8	1	7	0	8

Project Guide

Head of the Department

Examiner

1.
2.

Place: Ujire
Date:

Netflix Data Movie and T.V Series Recommendation system

Project Report submitted to the

SDM COLLEGE (Autonomous)

in partial fulfilment of the degree of

MASTER OF SCIENCE
IN
STATISTICS

By

Arvind N Myageri

Under the supervision of

Prof. Supriya Shivadasan Padmavati

Department of Post Graduate Studies

and Research in Statistics

SRI DHARMASTHALA MANJUNATHESHWARA

COLLEGE (Autonomous)

UJIRE - 574240

Karnataka, INDIA

June 2020

DECLARATION

I, Arvind N Myageri , hereby declare that the matter embodied in this report entitled 'Netflix data movie recommendation system' is a bonafide record of project work carried out by me under the guidance and supervision of Prof. Supriya Shivadasan Padmavati Department of Statistics, SDM College, Ujire - 574240, Karnataka, India. I further declare that no part of the work contained in the report has previously been formed the basis for the award of any Degree, Diploma, Associateship, Fellowship or any other similar title or recognition of any other university.

Date: (ARVIND N MYAGERI)

Place:Ujire E-mail-id: arvindmyageri7@gmail.com

CERTIFICATE

This is to certify that the project report entitled `Exploratory data analysis and prediction of chloric kidney disease' is a bonafide record of an authentic work carried out by Darshan M S, under my guidance and supervision in the Department of Post Graduate Studies and Research in Statistics, SDM College, Ujire, in partial fulfilment of the requirements for the award of the degree of Master of Science in Statistics, under Mangalore University, Mangalagangothri. I further certify that this report or part there of has not previously been presented or submitted else where for the award of any Degree, Diploma, Associateship, Fellowship or any other similar title of any other institution or university.

Date: (Supriya shivadasan padmavati)
Place: Ujire E-mail id:supriyasp@sdmcujire.in

ACKNOWLEDGEMENTS

Firstly, I would like to thank our Principal ***Dr. B. Yashovarma*** *for providing the necessary facilities for the completion of this project work in our college.*

I would also thank our Dean ***Prof. Ganapayya*** *for her support.*

It is my privilege to thank our HOD ***Prof. Shanthiprakash*** *for his suggestions and support.*

I am very grateful to my Research Supervisor, ***Prof. Supriya shivadasan padmavati****, Department of Statistics, SDM College, Ujire, for his kind help and encouragement throughout my project work.*

With great pleasure, I acknowledge my profound sense of gratitude to ***Prof. Dr.Savitha Rao****, Visiting Professor, Department of Statistics, SDM College, Ujire, for hir valuable guidance, supervision and advice.*

I gratefully acknowledge my teachers at the Department of Statistics, SDM Col- lege, Ujire, ***Asst. Prof. Dr. babitha.*** *and* ***Asst. Prof. Diksha*** *for their support during my project work.*

I am also thankful to all my family members and friends for their constant encouragement and help in each step.

My sincere thanks also goes to the students of SDM College, Ujire, who have helped me directly or indirectly during my project work.

*Finally to all who helped me in many ways, I say, '****Thank You****!'.*

(Arvind N Myageri)

Contents

Chapter 1.

1.1 INTRODUCTION:

What is Netflix? Netflix is a streaming service that allows our members to watch a wide variety of award-winning TV shows, movies, documentaries, and more on thousands of internet-connected devices. With Netflix, you can enjoy unlimited ad-free viewing of our content. There's always something new to discover, and more TV shows and movies are added every month!

TV Shows & Movies In over 190 countries, Netflix members get instant access to great content. Netflix has an extensive global content library featuring award-winning Netflix originals, feature films, documentaries, TV shows, and more. Netflix content will vary by region, and may change over time. The more you watch, the better Netflix gets at recommending TV shows and movies you'll love.You can play, pause, and resume watching without ads or commitments.Plus, you can download your favorite shows to your iOS or Android mobile device, or Windows 10 app. With downloads, you can watch while you're on the go and without an internet connection. Go ahead, binge a little!

Streaming devices watch anywhere, anytime, on thousands of devices. Netflix streaming software allows you to instantly watch content from Netflix through any internet-connected device that offers the Netflix app, including smart TVs, game consoles, streaming media players, set-top boxes, smartphones, and tablets. View our Internet Speed Recommendations to achieve the best performance. You can also stream Netflix directly from your computer or laptop. We recommend reviewing the System Requirements for web browser compatibility.

At some point each one of us must have wondered where all the recommendations that Netflix, Amazon, Google give us, come from. We often rate products on the internet and all the preferences we express and data we share (explicitly or not), are used by recommender systems to generate, in fact, recommendations. The two main types of recommender systems are either collaborative or content-based filters: these two names are pretty self-explanatory.

1.2 What is a recommender system?

A recommender system is a simple algorithm whose aim is to provide the most relevant information to a user by discovering patterns in a dataset. The algorithm rates the items and shows the user the items that they would rate highly. An example of recommendation in action is when you visit Amazon and you notice that some items are being recommended to you or when Netflix recommends certain movies to you. They are also used by Music streaming applications such as Spotify and Deezer to recommend music that you might like.

Below is a very simple illustration of how recommender systems work in the context of an e-commerce site.

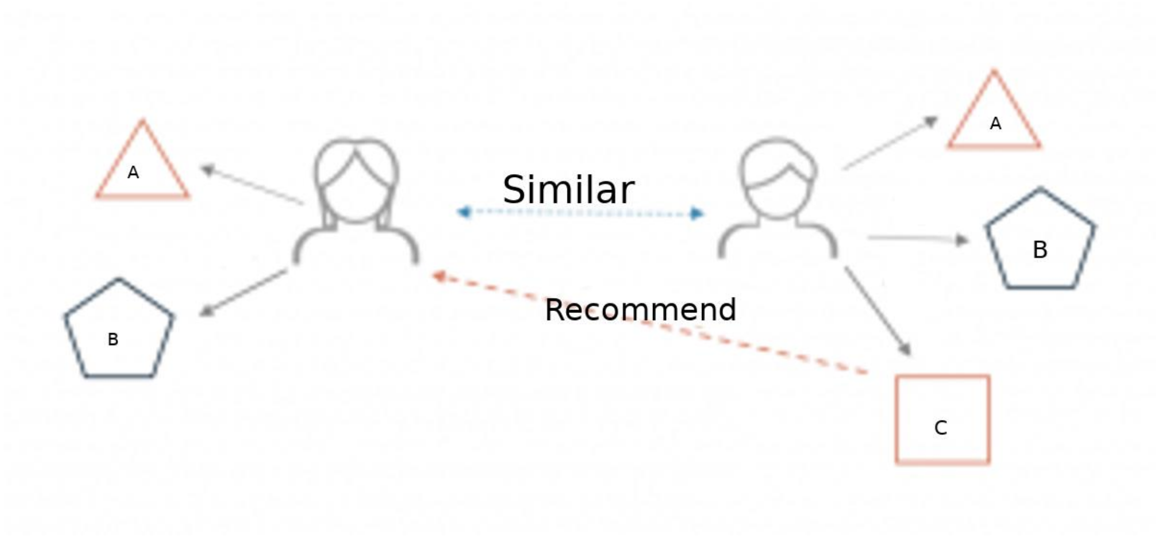


Fig.1.1 Recommender System

Two users buy the same items A and B from an e-commerce store. When this happens the similarity index of these two users is computed. Depending on the score the system can recommend item C to the other user because it detects that those two users are similar in terms of the items they purchase.

1.3 Data Description:

The Netflix dataset consists of 10 attributes and 6172 instances.

The 10 attributes are:

show_id	Unique ID for every Movie / Tv Show.
type	Identifier - A Movie or TV Show
title	Title of the Movie / Tv Show
director	Director of the Movie
cast	Actors involved in the movie / show
country	Country where the movie / show was produced
date_added	Date it was added on Netflix
release_year	Actual Release year of the move / show
rating	TV Rating of the movie / show
duration	Total Duration - in minutes or number of seasons

1.4 Objectives:

- To carry out Exploratory analysis of the Netflix data.
- To Create a Movie recommendation system using content based Filtering.
- To Create a TV. Show recommendation system using content based filtering and cosine similarity

Chapter 2.
Methodology

2.1 Exploratory data analysis:

2.1.1 Pie Chart:

A pie chart is a type of graph in which a circle is divided into sectors that each represents a proportion of the whole. Pie charts are a useful way to organize data in order to see the size of components relative to the whole, and are particularly good at showing percentage or proportional data. While pie charts are popular data representations, they can be hard to read, and it can be difficult to compare data from one pie chart to another. Pie charts are a useful way to visualize information that might be presented in a small table.

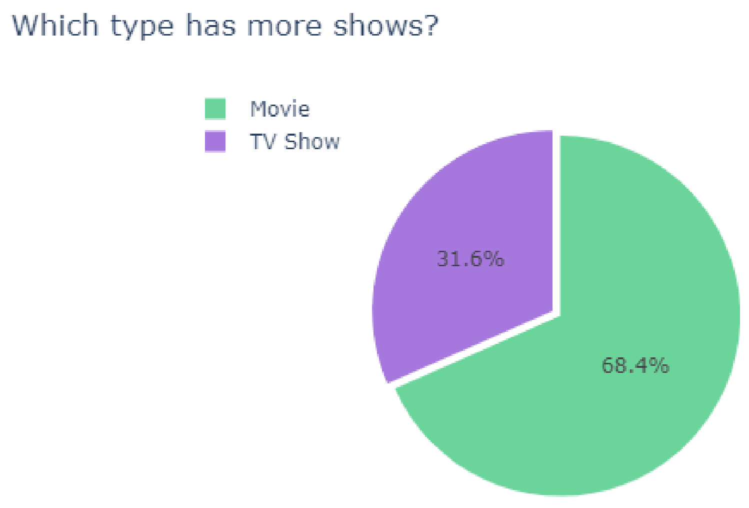


Fig.2.1.1.1

Netflix has more movies than T.V Shows.

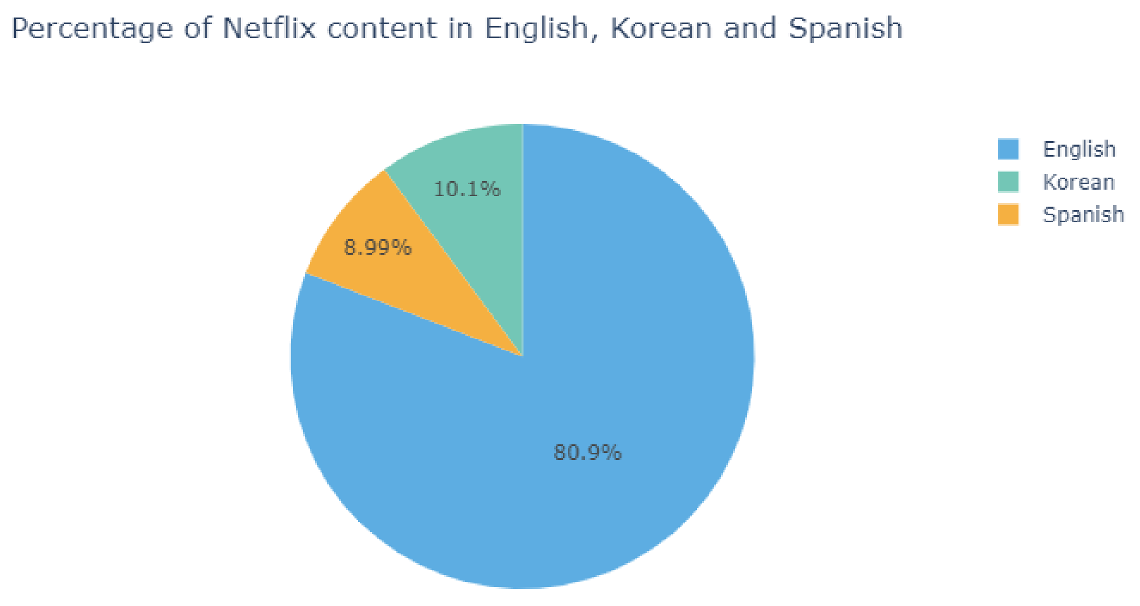


Fig.2.1.1.2

In Netflix, there's a majority of English language movies and T.V Shows with Korean and Spanish as the second and third language with most T.V Shows and Movies.

2.1.2 Scatter Plot:

Scatter plots are used to plot data points on a horizontal and a vertical axis in the attempt to show how much one variable is affected by another. Each row in the data table is represented by a marker whose position depends on its values in the columns set on the X and Y axes.

A third variable can be set to correspond to the color or size of the markers, thus adding yet another dimension to the plot.

The relationship between two variables is called their correlation. If the markers are close to making a straight line in the scatter plot, the two variables have a high correlation. If the markers are equally distributed in the scatter plot, the correlation is low, or zero. However, even though a correlation may seem to be present, this might not always be the case. Both variables could be related to some third variable, thus explaining their variation, or, pure coincidence might cause an apparent correlation.

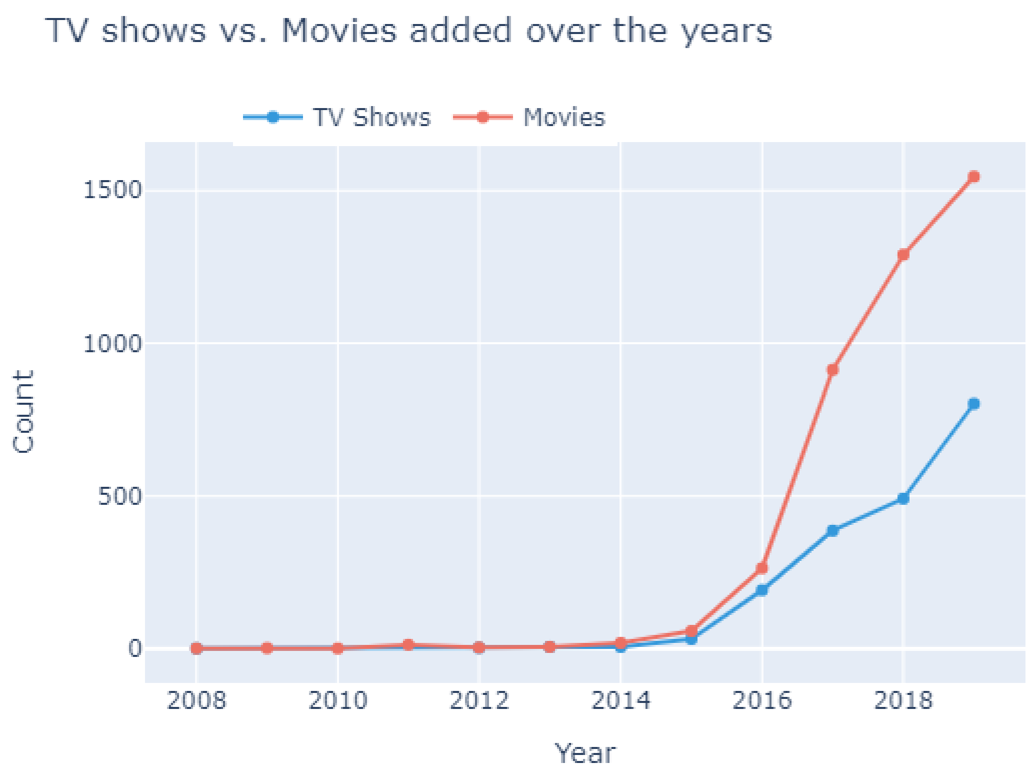


Fig2.1.2.1

As we know, Netflix works with content providers, distributors, producers, and creators to acquire licensing for TV shows and movies to stream on our service. so, the data shows Netflix purchasing and inversting more in movies licenses for the customers.

2.1.3 Bar Plot:

A bar chart is a chart or graph that presents categorical data with rectangular bars with heights or lengths proportional to the values that they present. A bar graph shows the comparison among discrete categories. One axis of the chart shows the specie categories being compared, and the other axis represents the measured value.

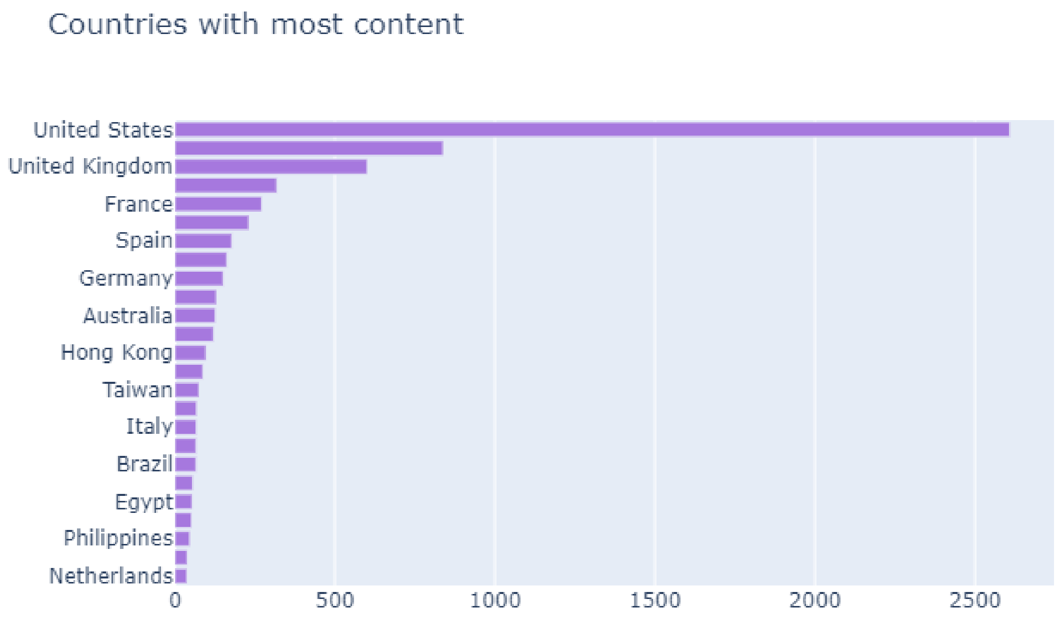


Fig.2.1.3.1

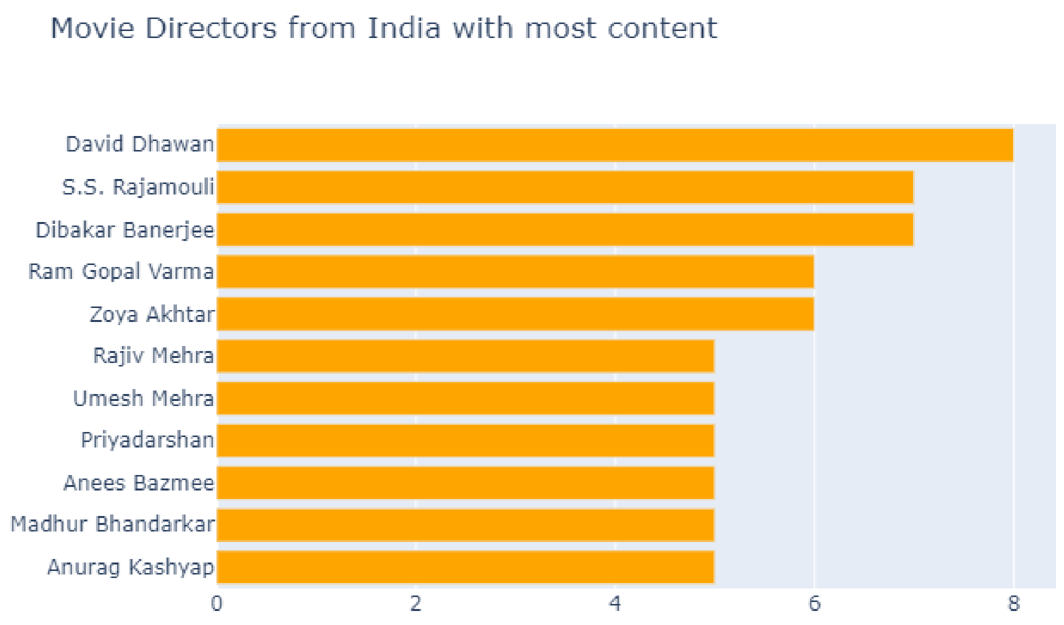


Fig.2.1.3.2

Koduri Srisaila Sri Rajamouli, professionally known as S. S. Rajamouli, is an Indian film director, and screenwriter known for his works predominantly in the Telugu cinema.He is known for directing high fantasy works such as Magadheera (2009), Eega (2012), Baahubali: The Beginning (2015), and Baahubali 2: The Conclusion (2017) which received the American Saturn Award for Best International Film, and the Australian Telstra People's Choice Award.

Dibakar Banerjee (born 21 June 1969) is an Indian film director, screenwriter, producer and advertisement-filmmaker known for his work in Hindi films. He also runs his own film production company, Dibakar Banerjee Productions. As a film maker, he is known for Khosla Ka Ghosla (2006), Oye Lucky! Lucky Oye! (2008), both of which won National Film Awards.His next film was the experimental Love Sex Aur

Dhokha (2010). It was followed by the political drama Shanghai (2012) and Bombay Talkies (2013), which was made as a celebration of the centenary year of Indian cinema.

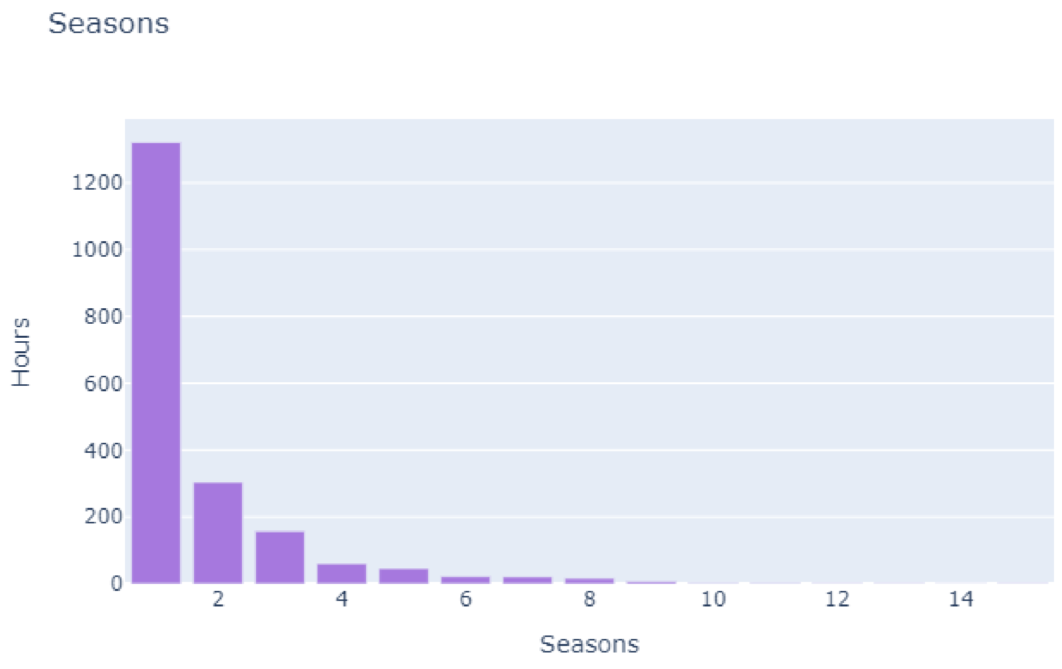
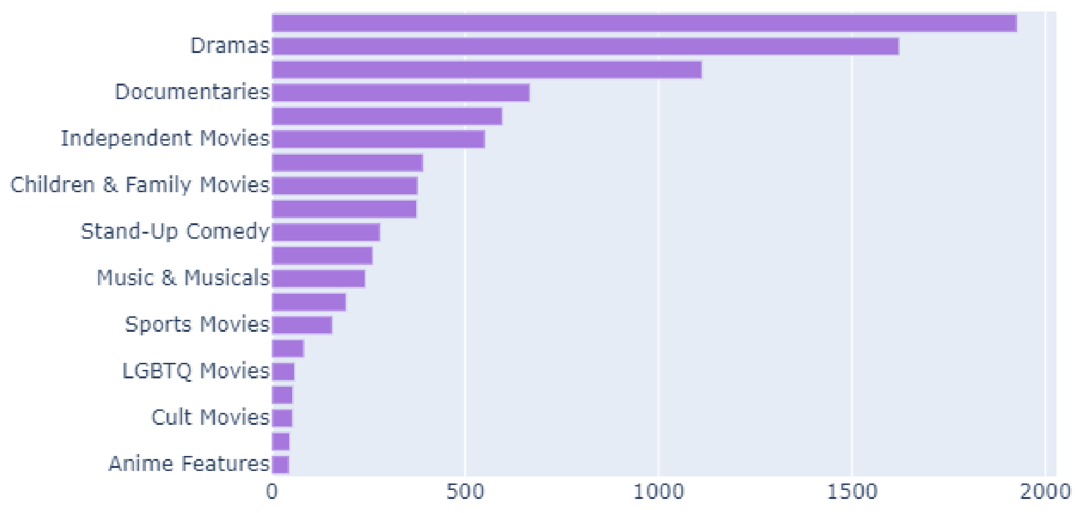


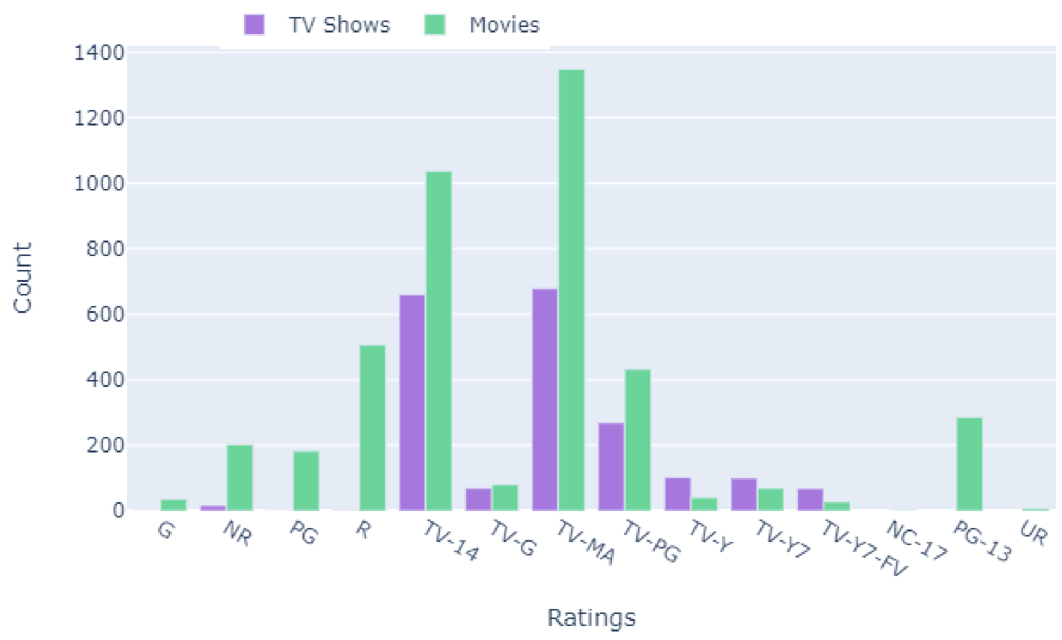
Fig.2.1.3.3

Grey's Anatomy and NCIS have highest no. of seasons on Netflix.

Content added over the years



Rating by content type



TV-MA: MATURE AUDIENCE ONLY:

This program is specifically designed to be viewed by adults and therefore may be unsuitable for children under 17. This program contains one or more of the following: graphic violence (V), explicit sexual activity (S), or crude indecent language (L).

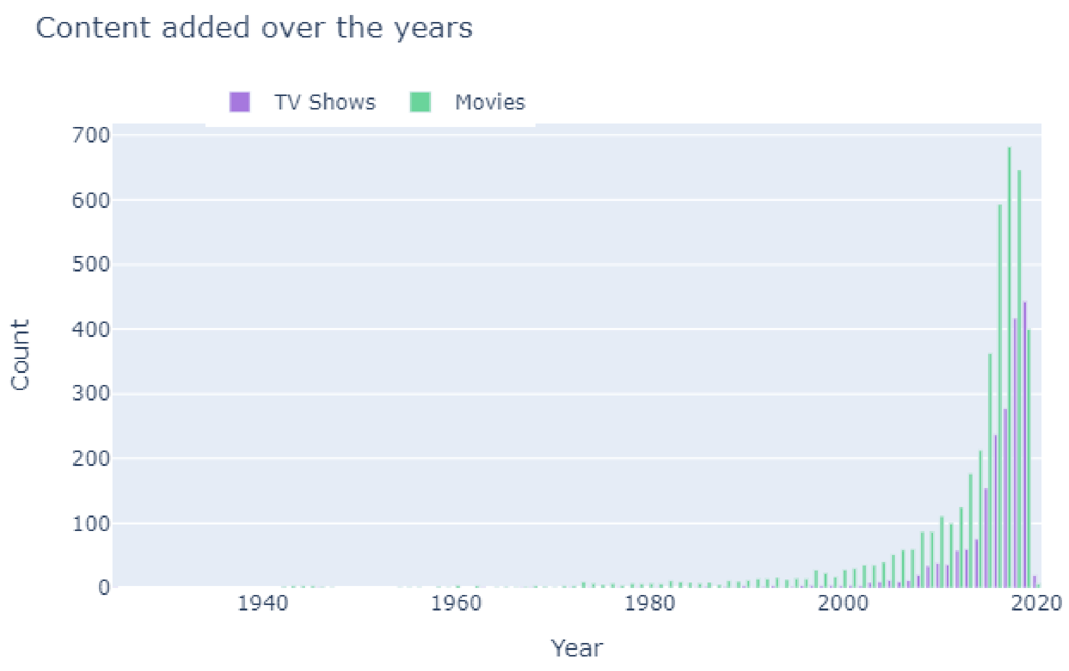
TV-14: PARENTS STRONGLY CAUTIONED:

This program contains some material that parents would find unsuitable for children under 14 years of age. Parents are strongly urged to exercise greater care in monitoring this program and are cautioned against letting children under the age of 14 watch unattended. This program contains one or more of the following: intense violence (V), intense sexual situations (S), strong coarse language (L), or intensely suggestive dialogue (D).

2.1.4 Histogram:

A histogram is a graphical representation of the distribution of numerical data.

When the class intervals are of equal length, apart from a constant, it is an estimate of the probability density function of a continuous variable (quantitative variable) and was first introduced by Karl Pearson. It is a kind of bar graph. To construct a histogram, the first step is to form intervals that divide the entire range of values into a series of intervals and then count how many values fall into each interval. The intervals are usually specified as consecutive, non-overlapping intervals of a variable. The intervals must be adjacent, and are often (but are not required to be) of equal size.



Above plot is show deduce that Netflix start releasing more content from 2000 and It is showing increasing trend over the year in the TV show as well as in the movies. These are rough estimates based on a list published by Netflix of more than 850 titles that are streamed exclusively on Netflix, including originals and first-run titles that the company syndicates and premieres in some of its markets, as of Dec. 31, 2018. Netflix also released more than 10,100 minutes of original documentaries, about 8,500 minutes of movies, 4,800 minutes of kids programming, and 3,600 minutes of stand-up comedy.

2.2 Different types of recommendation engines

The most common types of recommendation systems are content-based and collaborative filtering recommender systems. In collaborative filtering, the behavior of a group of users is used to make recommendations to other users. The recommendation is based on the preference of other users. A simple example would be recommending a movie to a user based on the fact that their friend liked the movie. There are two types of collaborative models Memory-based methods and Model-based methods. The advantage of memory-based techniques is that they are simple to implement and the resulting recommendations are often easy to explain. They are divided into two:

2.2.1: Collaborative filters:

This type of filter is based on users' rates, and it will recommend us movies that we haven't watched yet, but users similar to us have, and like. To determine whether two users are similar or not, this filter considers the movies both of them watched and how they rated them. By looking at the items in common, this type of algorithm will basically predict the rate of a movie for a user who hasn't watched it yet, based on the similar users' rates

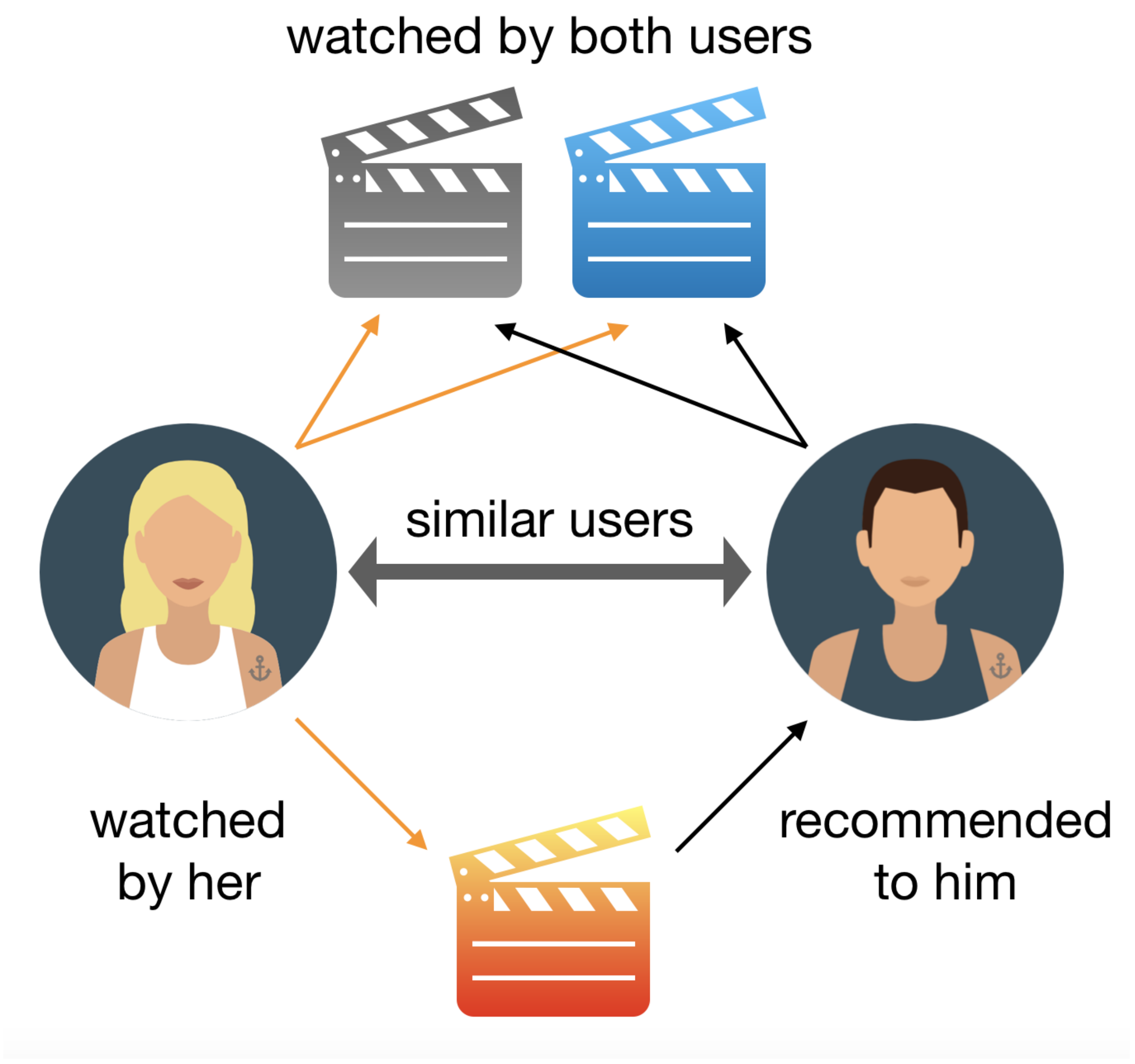


Fig 2.1Collaborative-based filter.

In order to work accurately, this type of filter needs rates, and not all users rate products constantly. Some of them barely or never rate anything! Another characteristic of this method is the diversity in the recommendations, that can be good or bad, depending on the case. For example, let's say user A likes dystopian movies and dark comedy a lot. User B also enjoys dystopian movies but never watched dark comedy. The collaborative filter will recommend dark comedy shows to user B, based on the common taste that the two users have for dystopian movies. This scenario can go two ways: either user B finds out that he/she likes dark comedy a lot, and in that case, great, a lot of new things to watch on his/her list! Or, user B really enjoys a lighter comedy style, and in that case the recommendation has not been successful.

2.2.2.1:User-based collaborative filtering:
In this model, products are recommended to a user based on the fact that the products have been liked by users similar to the user. For example, if Derrick and Dennis like the same movies and a new movie come out that Derick like, then we can recommend that movie to Dennis because Derrick and Dennis seem to like the same movies.

2.2.2.2:Item-based collaborative filtering:
These systems identify similar items based on users' previous ratings. For example, if users A, B, and C gave a 5-star rating to books X and Y then when a user D buys book Y they also get a recommendation to purchase book X because the system identifies book X and Y as similar based on the ratings of users A, B, and C.

Model-based methods are based on Matrix Factorization and are better at dealing with sparsity. They are developed using data mining, machine learning algorithms to predict users' rating of unrated items. Examples of such model-based methods include Decision trees, Rule-based Model, Bayesian Model, and latent factor models.

2.3 Content-based filters:

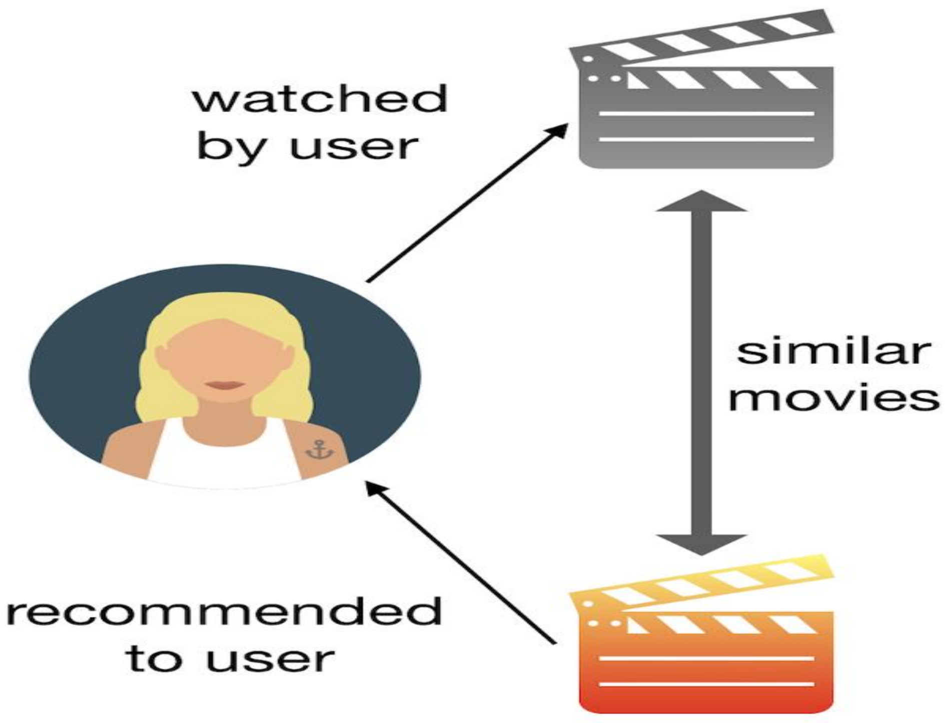


Fig 2.2 Content-based filter.

This type of filter does not involve other users if not ourselves. Based on what we like, the algorithm will simply pick items with similar content to recommend us.

In this case there will be less diversity in the recommendations, but this will work either the user rates things or not. If we compare this to the example above, maybe user B potentially likes dark comedy but he/she will never know, unless he/she decides to give it a try autonomously, because this filter will only keep recommending action movies or similar. Content-based systems use metadata such as genre, producer, actor, musician to recommend items say movies or music. Such a recommendation would be for instance recommending XXX that featured Vin Diesel because someone watched and liked The Fate of the Furious. Similarly, you can get music recommendations from certain artists because you liked their music. Content-based systems are based on the idea that if you liked a certain item you are most likely to like something that is similar to it.

2.4 Cosine similarity:

The dot product between two vectors is equal to the projection of one of them on the other. Therefore, the dot product between two identical vectors (i.e. with identical components) is equal to their squared module, while if the two are perpendicular (i.e. they do not share any directions), the dot product is zero. Generally, for n-dimensional vectors, the dot product can be calculated as shown below.

$$\boldsymbol{u} \cdot \boldsymbol{v} = [u_1\ u_2\ \dots\ u_n] \cdot \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = u_1 v_1 + u_2 v_2 + \dots + u_n v_n = \sum_{i=1}^{n} u_i v_i$$

Dot product.

The dot product is important when defining the similarity, as it is directly connected to it. The definition of similarity between two vectors u and v is, in fact, the ratio between their dot product and the product of their magnitudes.

$$similarity = cos(\theta) = \frac{\boldsymbol{u} \cdot \boldsymbol{v}}{\|\boldsymbol{u}\| \|\boldsymbol{v}\|} = \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2} \sqrt{\sum_{i=1}^{n} v_i^2}}$$

Similarity.

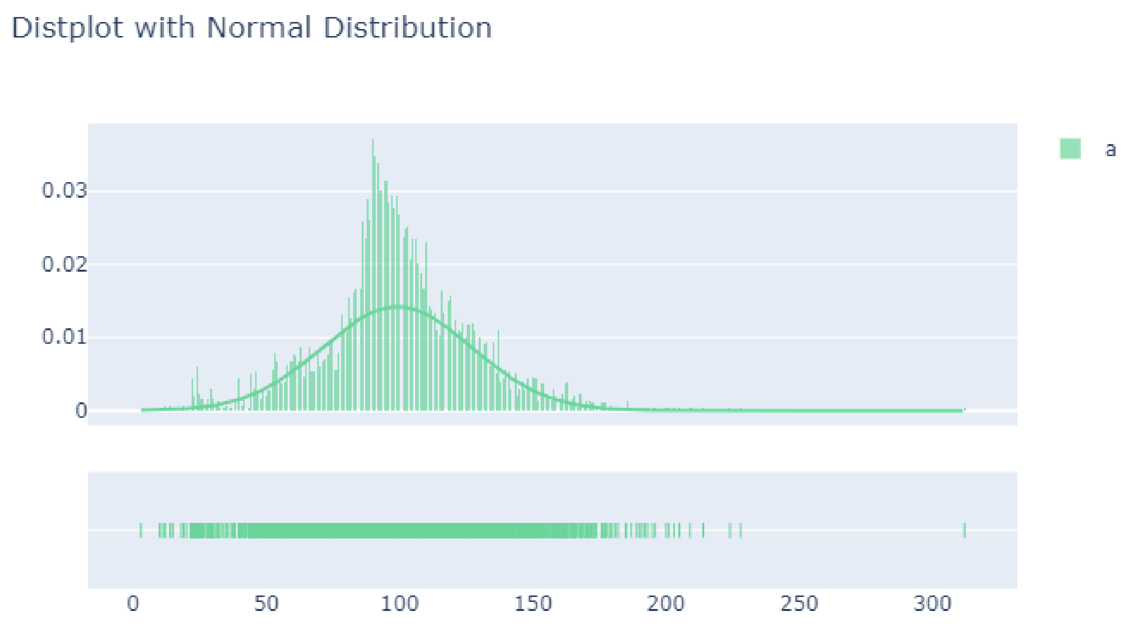
By applying the definition of similarity, this will be in fact equal to 1 if the two vectors are identical, and it will be 0 if the two are orthogonal. In other words, the similarity is a number bounded between 0 and 1 that tells us how much the two vectors are similar. Pretty straightforward!

Chapter 3

Results And Discussions:

3.1 Normal Distribution plot:

The normal probability plot (Chambers et al., 1983) is a graphical technique for assessing whether or not a data set is approximately normally distributed. The data are plotted against a theoretical normal distribution in such a way that the points should form an approximate straight line.



3.2 Data Cleaning:

Walkthrough of building a recommender system:

We are going to use the movie lens to build a simple item similarity-based recommender system. The first thing we need to do is to import pandas and numpy.

```
import plotly.graph_objs as go

from plotly.offline import init_notebook_mode, iplot

import pandas as pd


df = pd.read_csv("netflix_titles.csv")
```

There's a feature within the nltk package that allows us to extract the key words from a text, and it even assigns scores to each word. I did not really consider the scores for this basic recommender, however I used this Rake function to extract key words from the description column, so instead of using the entire sentences describing the plot, I only considered the most relevant words in the description.

```
!pip install rake-nltk

from rake_nltk import Rake

from sklearn.metrics.pairwise import cosine_similarity

from sklearn.feature_extraction.text import CountVectorizer



new_data=df[['type','listed_in','director','cast','country','rating','t
itle','description']]

new_data.head()
```

# REMOVE NaN VALUES AND EMPTY STRINGS:

```
new_data.dropna(inplace=True)

blanks = []  # start with an empty list

col=['type','listed_in','director','cast','country','rating']

for i,col in new_data.iterrows():  # iterate over the DataFrame

    if type(col)==str:            # avoid NaN values
```

```
        if col.isspace():         # test 'review' for whitespace
            blanks.append(i)     # add matching index numbers to the
list


new_data.drop(blanks, inplace=True)
```

#initializing the new column

```
new_data['Key_words/desc'] = ''
```

#instantiating Rake, by default it uses english stopwords from NLTK and discards all puntuation characters as well

```
for i,n in new_data.iterrows():
    desc = n['description']
    r = Rake()
```

#extracting the words by passing the text

```
    r.extract_keywords_from_text(desc)
```

#getting the dictionary whith key words as keys and their scores as values

```
    score_for_keyword = r.get_word_degrees()
```

#assigning the key words to the new column for the corresponding movie

```
    n['Key_words/desc']=list(score_for_keyword.keys())
```

#discarding the commas between the actors' full names and getting only the first three names

```
new_data['cast'] = new_data['cast'].map(lambda x:x.split(',')[:3])
```

#putting the genres in a list of words

```
new_data['listed_in'] = new_data['listed_in'].map(lambda
x:x.lower().split(','))

new_data['type'] = new_data['type'].map(lambda x:x.lower().split(','))

new_data['country'] = new_data['country'].map(lambda
x:x.lower().split(','))

new_data['rating'] = new_data['rating'].map(lambda
x:x.lower().split(','))
```

```
new_data['director'] = new_data['director'].map(lambda x:x.split(','))

new_data.drop('description',axis=1, inplace=True)

new_data.head(10)
```

#merging together first and last name for each actor and director, so it's considered as one word and there is no mix up between people sharing a first name

```
for i,n in new_data.iterrows():

    n['cast'] = [x.lower().replace(' ','') for x in n['cast']]

    n['type'] = [x.lower().replace(' ','') for x in n['type']]

    n['rating'] = [x.lower().replace(' ','') for x in n['rating']]

    n['country'] = [x.lower().replace(' ','') for x in n['country']]

    n['director'] = ''.join(n['director']).lower()


new_data = new_data.set_index('title')

new_data.head(10)
```

#creating a new column

```
new_data['bag_of_words'] = ''


cols = new_data.columns

for i,j in new_data.iterrows():

    words = ''

    for k in cols:

        if k!='director':

            words = words + ' '.join(j[k])+ ' '

        else:

            words = words + j[k] + ' '


    j['bag_of_words'] = words

new_data.head(10)
```

3.3 Feature Extraction and Modelling:

# instantiating and generating the count matrix

```
count = CountVectorizer()

CV = count.fit_transform(clean_data['bag_of_words'])
```

# generating the cosine similarity matrix

```
similarity= cosine_similarity(CV,CV)

similarity
```

The similarity matrix looks like this.

```
array([[1.        , 0.13655775, 0.09534626, ..., 0.14625448, 0.09475587
,
        0.1539981 ],
       [0.13655775, 1.        , 0.07161149, ..., 0.13730876, 0.13344013
,
        0.14457873],
       [0.09534626, 0.07161149, 1.        , ..., 0.051131  , 0.0248452
,
        0.0269191 ],
       ...,
       [0.14625448, 0.13730876, 0.051131  , ..., 1.        , 0.38110797
,
        0.45421208],
       [0.09475587, 0.13344013, 0.0248452 , ..., 0.38110797, 1.
,
        0.40128618],
       [0.1539981 , 0.14457873, 0.0269191 , ..., 0.45421208, 0.40128618
,
        1.        ]])
```

all the numbers on the diagonal are 1 because, of course, every movie is identical to itself. The matrix is also symmetrical because the similarity between A and B is the same as the similarity between B and A.

#function that takes in movie title as input and returns the top 10 recommended movies

```
listing = pd.Series(clean_data.index)

def recommendations(Title, cosine_sim = similarity):


    recommended_movies = []
```

    # getting the index of the movie that matches the title

```
    idx = listing[listing == Title].index[0]
```

    # creating a Series with the similarity scores in descending
order

```
    score_series = pd.Series(cosine_sim[idx]).sort_values(ascending =
False)
```

    # getting the indexes of the 10 most similar movies

```
    top_10_indexes = list(score_series.iloc[1:11].index)
```

   # populating the list with the titles of the best 10 matching
movies

```
    for i in top_10_indexes:

        recommended_movies.append(list(clean_data.index)[i])

    return recommended_movies

recommendations('PK')

['3 Idiots',
 'Merku Thodarchi Malai',
 'Sanju',
 'Hattrick',
 'Harishchandrachi Factory',
 'English Babu Desi Mem',
 'Dil Chahta Hai',
 'War Chhod Na Yaar',
 'Kacche Dhaagey',
 'Mahabharat']
```

Reference

1. Peng, Xiao, Shao Liangshan, and Li Xiuran. "Improved Collaborative Filtering Algorithm

in the Research and Application of Personalized Movie Recommendations",

2013 Fourth International Conference on Intelligent Systems Design and Engineering

Applications, 2013.

2. Munoz-Organero, Mario, Gustavo A. Ramíez-González, Pedro J. Munoz-Merino, and

Carlos Delgado Kloos. "A Collaborative Recommender System Based on SpaceTime Similarities", IEEE Pervasive Computing, 2010.

3. Al-Shamri, M.Y.H.. "Fuzzy-genetic approach to recommender systems based on a novel

hybrid user model", Expert Systems With Applications, 200810

4. Hu Jinming. "Application and research of collaborative filtering in e-commerce

recommendation system", 2010 3rd International Conference on Computer Science and

Information Technology, 07/2010

5. Xu, Qingzhen Wu, Jiayong Chen, Qiang. "A novel mobile personalized recommended

method based on money flow model for stock exchange.(Researc", Mathematical Problems

in Engineering, Annual 2014 Issue

6. Yan, Bo, and Guanling Chen. "AppJoy : personalized mobile application discovery",

Proceedings of the 9th international conference on Mobile systems applications and services

- MobiSys 11 MobiSys 11, 2011.

7. Davidsson C, Moritz S. Utilizing implicit feedback and context to recommend mobile

applications from first use.In: Proc. of the Ca RR 2011. New York: ACM Press, 2011. 19-

22.http://dl.acm.org/citation.cfm?id=1961639[doi:10.1145/1961634.1961639]

8. Bilge, A., Kaleli, C., Yakut, I., Gunes, I., Polat, H.: A survey of privacy-preserving

collaborative filtering schemes. Int. J. Softw. Eng. Knowl. Eng. 23(08), 1085–

1108 (2013)CrossRefGoogle Scholar

9. Calandrino, J.A., Kilzer, A., Narayanan, A., Felten, E.W., Shmatikov, V.: You might also

like: privacy risks of collaborative filtering. In: Proceedings of the IEEE Symposium on

Security and Privacy, pp. 231–246, Oakland, CA, USA (2011)