NOTEBOOK

Netflix Movie Recommendation

Recommend movies to users

Learn

#1

Introduction

NETFLIX

Netflix was first founded in August of 1997 by two serial entrepreneurs, Marc Randolph and Reed Hastings. The company began out in Scotts Valley, California, and has grown to become one of the world's leading internet entertainment platforms.

For millions, Netflix is the de-facto place to go for movie and TV streaming. According to sites like fortune.com, its services alone constitute about 15% of all the world's internet bandwidth! Not bad for a company that started by posting DVDs by snail mail.

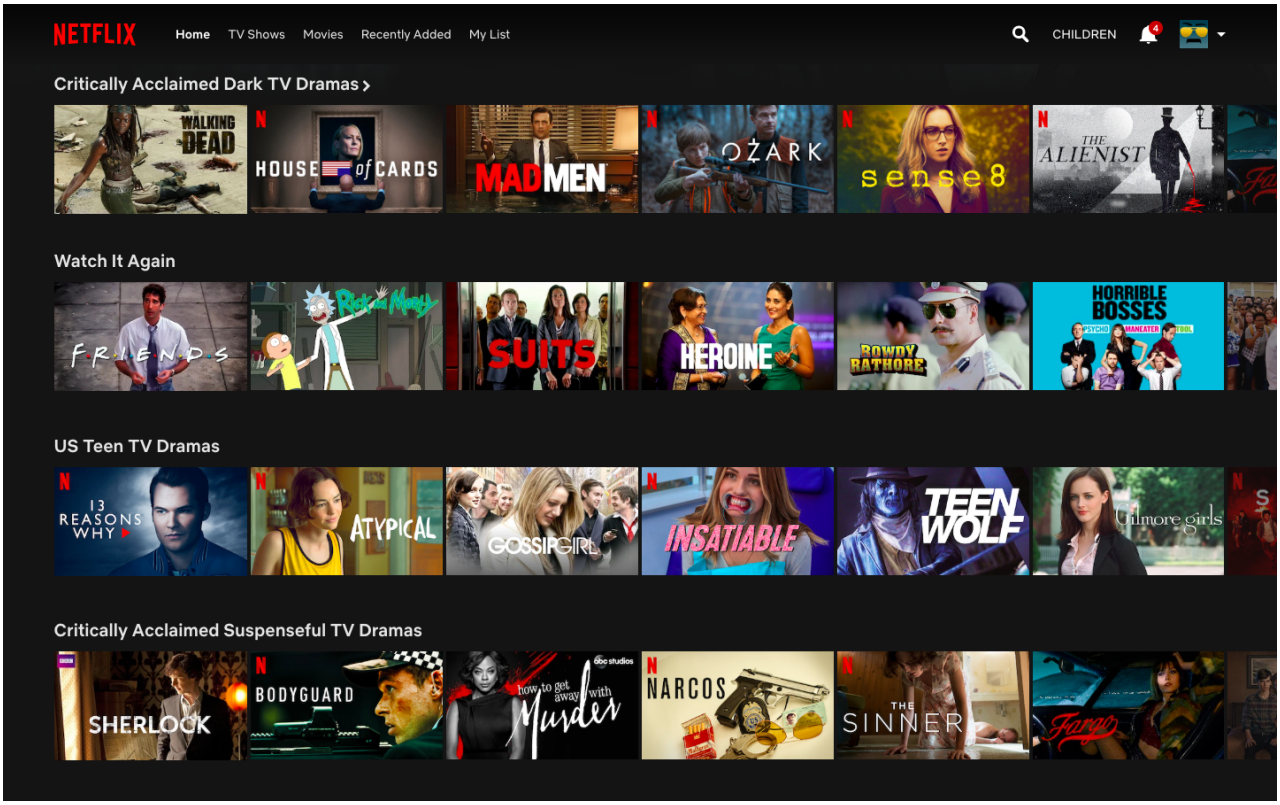
When it first opened, Netflix was purely a movie rental service. Users ordered movies on the Netflix website and received DVDs in the post. When they were finished with them, they would simply post them back to Netflix in the envelopes provided. At the time, this was seen as a boon to those who did not have a video rental store nearby (remember those?).

Today, Netflix streams movies and has more than 151 million paid subscribers in over 190 countries around the world. It offers a wide range of TV series, documentaries, and feature films across a wide variety of genres and languages, including original productions.

#2

Recommendation System

It was the year 2000 when Netflix introduced a personalized movie recommendation system, that uses members' ratings to predict choices for all Netflix members.

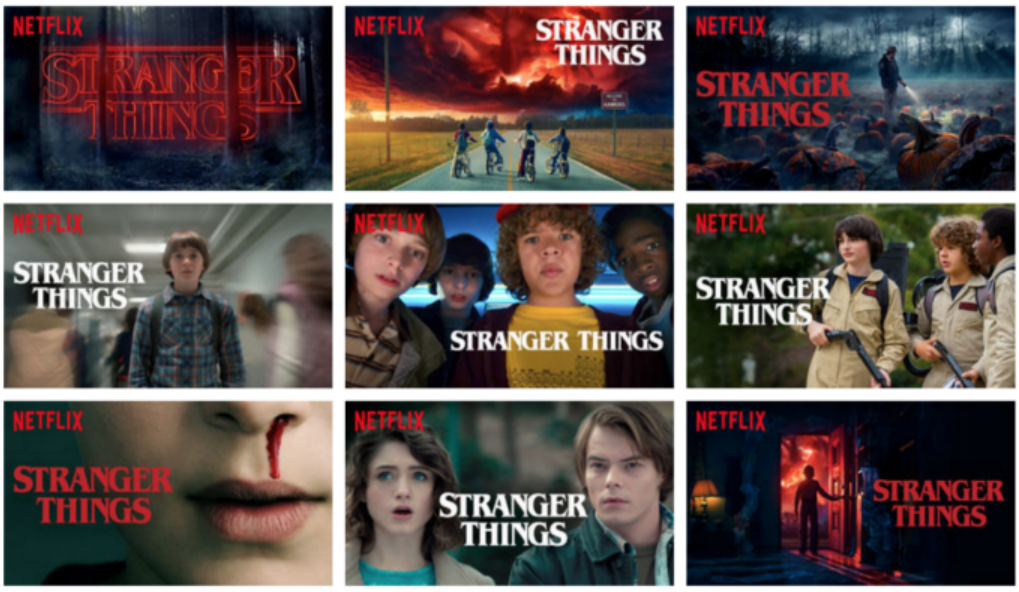


Recommendation algorithms are at the core of the Netflix product. It provides their members with personalized suggestions to reduce the amount of time and frustration to find something great content to watch. Because of the importance of our recommendations, they continually seek to improve them by advancing the state-of-the-art in the field. They do this by using the data about what content our members watch and enjoy along with how they interact with our service to get better at figuring out what the next great movie or TV show for them will be.

#3

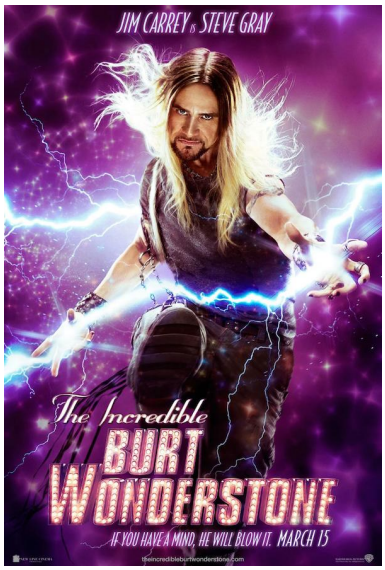
Artwork Personalization

Did you know that different user are shown different artworks of a particular show, and it's not just random? Netflix recommendation system also shows the best artwork for a show to a user.



Artwork for Stranger Things that each receive over 5% of impressions from our personalization algorithm. Different images cover a breadth of themes in the show to go beyond what any single image portrays.

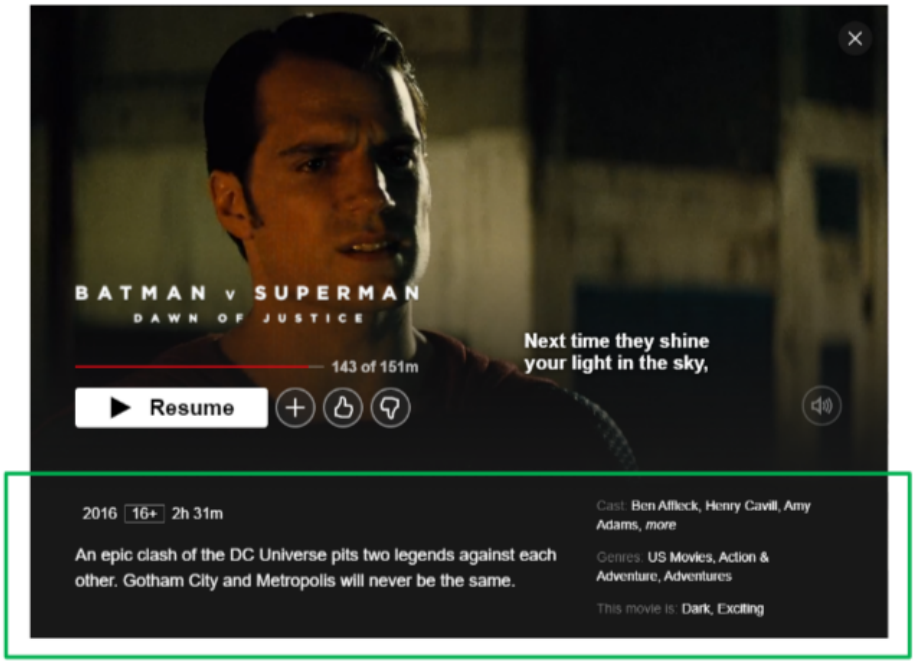
Let's say that a particular user likes Jim Carrey, then Netflix will show movie artwork with Jim Carrey, even he doesn't have a lead role in that movie or show. Like in the movie "The Incredible Burt Wonderstone" Steve Carell was playing the lead role, but Netflix will show movie poster of Jim Carrey



#4

Content Based Recommendation

It is a type of recommendation system in which recommended movies have similar content with the already liked movies. But first, we need to define the content. Content can be a story, or script. We can take brief summary of the movie, cast, and genres the content of the movie, and recommend movies with similar content.



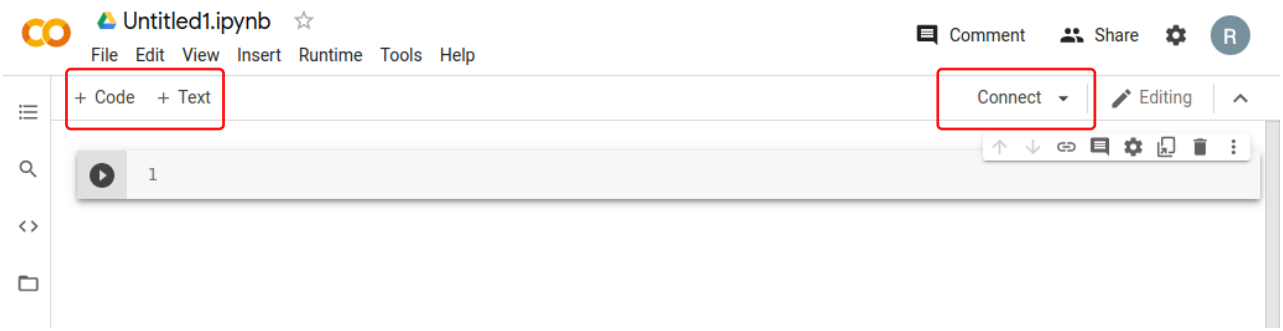
So, we will merge all of them in one text. But computers cannot understand the meaning of these texts, thus we need to apply some pre-processing operations to our "content". We will be using NLP (Natural Language Processing)

#Extra

Introduction to Colab

Nearly all machine learning and deep learning algorithms require good hardware. What if ... you don't have good hardware? Should you drop your dream to be a data scientist? No, there's an alternative Let us introduce you to Colab.

Colab is a service provided by Google which lets you access a virtual machine hosted on google servers. These virtual machines have dual-core Xeon processors, with 12GB of RAM. You can even use GPU for your neural networks. Colab is an interactive python notebook (ipynb), which means that with writing python code, you can also write normal text, include images.
To create a new colab notebook, just go to "https://colab.research.google.com", and create a new notebook. You will get something like below



You can connect to runtime by clicking the "Connect" button in the right corner. You can add a new Code cell, or text cell using respective buttons in the toolbar.

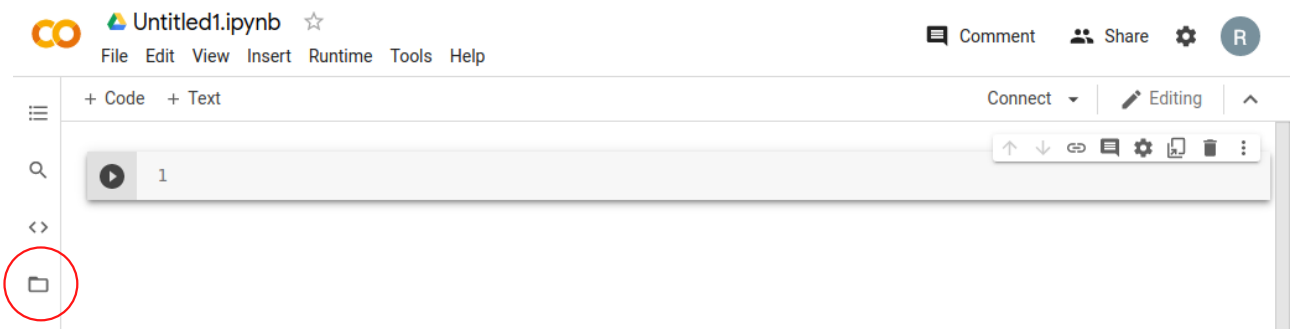
#Extra

Introduction to Colab

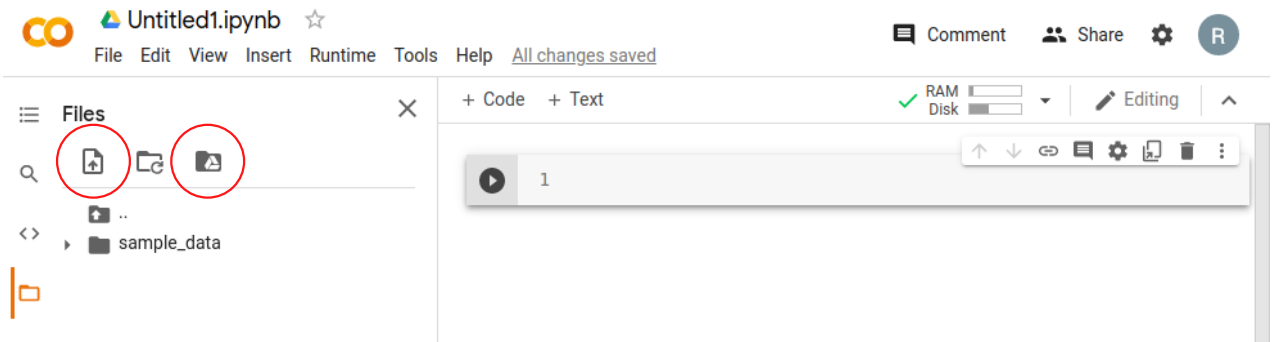
Uploading files

Sometimes you need to use a file from your PC. For that, you can upload the required files to google colab. Colab provides 100 GB in a colab session. If you want to access some files from your drive, you can even do so by connecting the drive to colab.

To upload something, open file pane from left toolbar.



Now, just upload the file using first button, and you can also mount google drive using last button



#3

Code

Upload

First you have to open shared .ipynb notebook in google colab, then upload the movies.csv file in colab files.

Reading dataset

```
import pandas as pd
movies = pd.read_csv('movies.csv')
movies.head(2)
```

Creating movie CONTENT

```
movies['content'] = movies['title'] + ' ' + movies['overview'] + ' '
+ movies['keywords'] + ' ' + movies['cast'] + ' ' + movies
['director']
```

#3.1

Natural Language Processing

Natural language processing is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data.

We will be using a three-step process to transform our "content" into a form that could be understood by a computer algorithm and with which it can extract meaningful insights.

#3.1.1

Tokenization

Tokenization is the process of breaking down sentence or paragraphs into smaller chunks of words called tokens

Bruce Wayne, a billionaire, believes that Superman is a threat to humanity after his battle in Metropolis.

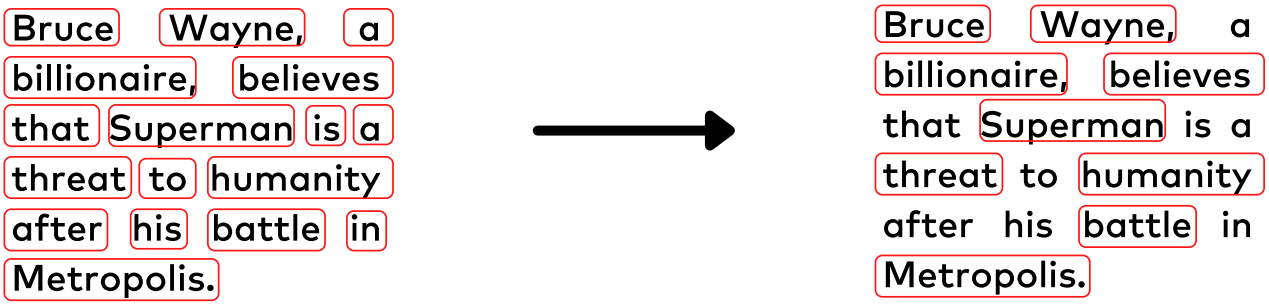
Bruce | Wayne, | a | billionaire, | believes | that | Superman | is | a | threat | to | humanity | after | his | battle | in | Metropolis.

#3.1.2

Stop Words Removal

On removal of some words, the meaning of the sentence doesn't change, like and, am. Those words are called stopwords and should be removed before feeding to any algorithm.

In datasets, some non-stop words repeat very frequently. Those words too should be removed to get an unbiased result from the algorithm.



#3.1.3

Vectorization

After tokenization, and stop words removal, our "content" are still in string format. We need to convert those strings to numbers based on their importance (features). We use TF-IDF vectorization to convert those text to vector of importance.

With TF-IDF we can extract important words in our data. It assign rarely occuring words a high number, and frequently occuring words a very low number.

You can learn more about it from:
https://en.wikipedia.org/wiki/Tf-idf

#3.1

Code

NLP

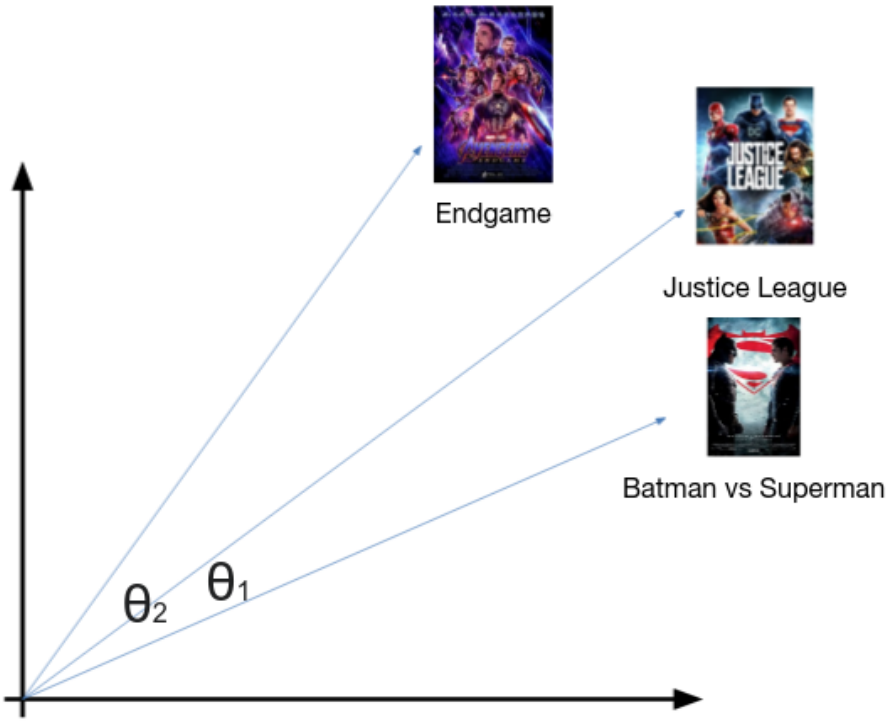
We can perform all those steps with just Tf-Idf Vectorizer of sklearn. We will pass 1000 as max_features (can be changed).

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(max_features=1000)
movie_vectors = vectorizer.fit_transform(movies['content'].values)
```

#4

Cosine Similarity

After using generating TF-IDF vectors, we will get a vector for each movie "content". As they are vector, they can be visualized on a cartesian plane.



As in above graph, we have plotted vectors of three movies on a cartesian plane. To find similar movies, we just need to calculate angle between two vectors. If angle is small, then we can say that those movies are similar.

You might think that we can't place all movies in this small cartesian plane. Correct, we can't. But the TF-IDF vectors generally have 500+ dimensions. Now, think how many movies can be placed in 500+ dimension cartesian plane.

#4

Code

Cosine Similarity

We will be using sklearn cosine_similarity function. As we need to calculate similarities among all movies, we will pass movies as X, and Y.

```
from sklearn.metrics.pairwise import cosine_similarity
similarity = cosine_similarity(movie_vectors)
```

	0	1	2	3	4	5	6
0	1.000000	0.120720	0.054727	0.181850	0.157633	0.169431	0.085322
1	0.120720	1.000000	0.099872	0.124117	0.216898	0.046506	0.091782
2	0.054727	0.099872	1.000000	0.073028	0.045837	0.046143	0.034967
3	0.181850	0.124117	0.073028	1.000000	0.126776	0.082809	0.146062
4	0.157633	0.216898	0.045837	0.126776	1.000000	0.080019	0.157128
...	...	...	...	...	...	...	...
4804	0.086665	0.102234	0.079478	0.128345	0.151762	0.060347	0.093782
4805	0.050144	0.085246	0.026120	0.082814	0.058291	0.053500	0.060803
4806	0.106024	0.151463	0.088255	0.141403	0.127700	0.128241	0.147168
4807	0.062859	0.137537	0.041129	0.088217	0.119581	0.070877	0.122555
4808	0.047848	0.065207	0.043192	0.128880	0.084958	0.054001	0.133388

The above dataframe shows cosine similarity of all movies. Cosine similarity of 1 means that movie is very similar, and 0 means that movies are not similar.

#4

Code

Combining all

```
def recommend(movie):
    # find movie index from dataset
    movies_index = movies[movies['title'] == movie].index[0]

    # finding cosine similarities of movie
    distances = similarity[movies_index]

    # sorting cosine similarities
    movies_list =
sorted(list(enumerate(distances)),reverse=True,key=lambda x:x[1])
[1:6]

    for i in movies_list:
        print(movies.iloc[i[0]].title)
```

Learn

Knowledge Discovery through Brand Stories

Visit Us at :
techlearn.live

Email : support@techlearn.live
Phone : +91-9154796743