

```
In [1]: import numpy as np
import pandas as pd
from sklearn.feature_extraction import text
from sklearn.metrics.pairwise import cosine_similarity

In [2]: data = pd.read_csv("netflixData.csv")
print(data.head())

      Show Id      Title \
0  cc1b6ed9-cf9e-4057-8303-34577fb54477  (Un)Well
1  e2ef4e91-fb25-42ab-b485-be8e3b23dedb  #Alive
2  b01b73b7-81f6-47a7-86d8-acb63080d525  #AnneFrank - Parallel Stories
3  b6611af0-f53c-4a08-9ffa-9716dc57eb9c  #blackAF
4  7f2d4170-bab8-4d75-adc2-197f7124c070  #cats_the_mewvie

      Description \
0  This docuseries takes a deep dive into the luc...
1  As a grisly virus rampages a city, a lone man ...
2  Through her diary, Anne Frank's story is retol...
3  Kenya Barris and his family navigate relations...
4  This pawesome documentary explores how our fel...

      Director \
0      NaN
1      Cho Il
2  Sabina Fedeli, Anna Migotto
3      NaN
4      Michael Margolis

      Genres \
0  Reality TV
1  Horror Movies, International Movies, Thrillers
2  Documentaries, International Movies
3      TV Comedies
4  Documentaries, International Movies

      Cast Production Country \
0      NaN      United States
1      Yoo Ah-in, Park Shin-hye      South Korea
2      Helen Mirren, Gengher Gatti      Italy
3  Kenya Barris, Rashida Jones, Iman Benson, Genn...      United States
4      NaN      Canada

      Release Date Rating  Duration Imdb Score Content Type      Date Added
0      2020.0      TV-MA      1 Season      6.6/10      TV Show      NaN
1      2020.0      TV-MA      99 min      6.2/10      Movie      September 8, 2020
2      2019.0      TV-14      95 min      6.4/10      Movie      July 1, 2020
3      2020.0      TV-MA      1 Season      6.6/10      TV Show      NaN
4      2020.0      TV-14      90 min      5.1/10      Movie      February 5, 2020

In [ ]: Now I will check to see if the data contains null values or not

In [3]: print(data.isnull().sum())

Show Id      0
Title        0
Description  0
Director    2064
Genres      0
Cast        530
Production Country  559
Release Date      3
Rating          4
Duration         3
Imdb Score      608
Content Type     0
Date Added     1335
dtype: int64

In [ ]: This dataset do contain null values. But first we will select the columns that we can use to build the
recommendation system.

In [4]: data = data[["Title", "Description", "Content Type", "Genres"]]
print(data.head())

      Title \
0      (Un)Well
1      #Alive
2  #AnneFrank - Parallel Stories
3      #blackAF
4      #cats_the_mewvie

      Description Content Type \
0  This docuseries takes a deep dive into the luc...      TV Show
1  As a grisly virus rampages a city, a lone man ...      Movie
2  Through her diary, Anne Frank's story is retol...      Movie
3  Kenya Barris and his family navigate relations...      TV Show
4  This pawesome documentary explores how our fel...      Movie

      Genres
0  Reality TV
1  Horror Movies, International Movies, Thrillers
2  Documentaries, International Movies
3      TV Comedies
4  Documentaries, International Movies

In [ ]: The title column contains the titles of movies and TV shows on Netflix.
Description column describes the plot of the TV shows and movies.
The Content Type column tells us if it's a movie or a TV show.
The Genre column contains all the genres of the TV show or the movie.

In [ ]: Next, drop the rows containing null values and move further

In [6]: data = data.dropna()

In [ ]: Now I will clean the Title column as it contains some data preparation

In [8]: import nltk
import re
nltk.download('stopwords')
stemmer = nltk.SnowballStemmer("english")
from nltk.corpus import stopwords
import string
stopword=set(stopwords.words('english'))

def clean(text):
    text = str(text).lower()
    text = re.sub('[\.\*\?]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w\d\w*', '', text)
    text = [word for word in text.split(' ') if word not in stopword]
    text=" ".join(text)
    text = [stemmer.stem(word) for word in text.split(' ')]
    text=" ".join(text)
    return text

data["Title"] = data["Title"].apply(clean)

[nltk_data] Downloading package stopwords to /Users/tavi/nltk_data...
[nltk_data] Package stopwords is already up-to-date!

In [9]: import nltk
import re
nltk.download('stopwords')
stemmer = nltk.SnowballStemmer("english")
from nltk.corpus import stopwords
import string
stopword=set(stopwords.words('english'))

[nltk_data] Downloading package stopwords to /Users/tavi/nltk_data...
[nltk_data] Package stopwords is already up-to-date!

In [10]: def clean(text):
    text = str(text).lower()
    text = re.sub('[\.\*\?]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w\d\w*', '', text)
    text = [word for word in text.split(' ') if word not in stopword]
    text=" ".join(text)
    text = [stemmer.stem(word) for word in text.split(' ')]
    text=" ".join(text)
    return text

data["Title"] = data["Title"].apply(clean)

In [ ]: Before moving forward, lets take a look at some samples of the Titles

In [11]: print(data.Title.sample(10))

3027      megalobox
1934      haikyu
5435  tomasz jachimek jacek stramik laugh live
719      blind intersect
4621      angel
1620      fight
2975      marriag mortgag
1254      delhi
4813      figurin araromir
4422      steve job
Name: Title, dtype: object

In [20]: import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

In [25]: indices = pd.Series(data.index,
index=data["Title"]).drop_duplicates()

In [27]: data['listed_in'] = data['listed_in'].apply(lambda x: x.replace(' ', '').lower())

-----
KeyError                                Traceback (most recent call last)
File ~/anaconda3/lib/python3.11/site-packages/pandas/core/indexes/base.py:3790, in Index.get_loc(self, key)
    3789 try:
-> 3790     return self._engine.get_loc(casted_key)
    3791 except KeyError as err:

File index.pyx:152, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:181, in pandas._libs.index.IndexEngine.get_loc()

File pandas/_libs/hashtable_class_helper.pxi:7080, in pandas._libs.hashtable.PyObjectHashTable.get_item()
File pandas/_libs/hashtable_class_helper.pxi:7088, in pandas._libs.hashtable.PyObjectHashTable.get_item()
KeyError: 'listed_in'

The above exception was the direct cause of the following exception:

KeyError                                Traceback (most recent call last)
Cell In[27], line 1
----> 1 data['listed_in'] = data['listed_in'].apply(lambda x: x.replace(' ', '').lower())

File ~/anaconda3/lib/python3.11/site-packages/pandas/core/frame.py:3896, in DataFrame.__getitem__(self, key)
    3894 if self.columns.nlevels > 1:
    3895     return self._getitem_multilevel(key)
-> 3896     indexer = self.columns.get_loc(key)
    3897 if is_integer(indexer):
    3898     indexer = [indexer]

File ~/anaconda3/lib/python3.11/site-packages/pandas/core/indexes/base.py:3797, in Index.get_loc(self, key)
    3792 if isinstance(casted_key, slice) or (
    3793     isinstance(casted_key, abc.Iterable)
    3794     and any(isinstance(x, slice) for x in casted_key)
    3795 ):
    3796     raise InvalidIndexError(key)
-> 3797     raise KeyError(key) from err
    3798 except TypeError:
    3799     # If we have a listlike key, _check_indexing_error will raise
    3800     # InvalidIndexError. Otherwise we fall through and re-raise
    3801     # the TypeError.
    3802     self._check_indexing_error(key)
KeyError: 'listed_in'

In [28]: print(data.head())
print(data.columns)

      Show Id      Title \
0  cc1b6ed9-cf9e-4057-8303-34577fb54477  (Un)Well
1  e2ef4e91-fb25-42ab-b485-be8e3b23dedb  #Alive
2  b01b73b7-81f6-47a7-86d8-acb63080d525  #AnneFrank - Parallel Stories
3  b6611af0-f53c-4a08-9ffa-9716dc57eb9c  #blackAF
4  7f2d4170-bab8-4d75-adc2-197f7124c070  #cats_the_mewvie

      Description \
0  This docuseries takes a deep dive into the luc...
1  As a grisly virus rampages a city, a lone man ...
2  Through her diary, Anne Frank's story is retol...
3  Kenya Barris and his family navigate relations...
4  This pawesome documentary explores how our fel...

      Director \
0      NaN
1      Cho Il
2  Sabina Fedeli, Anna Migotto
3      NaN
4      Michael Margolis

      Genres \
0  Reality TV
1  Horror Movies, International Movies, Thrillers
2  Documentaries, International Movies
3      TV Comedies
4  Documentaries, International Movies

      Cast Production Country \
0      NaN      United States
1      Yoo Ah-in, Park Shin-hye      South Korea
2      Helen Mirren, Gengher Gatti      Italy
3  Kenya Barris, Rashida Jones, Iman Benson, Genn...      United States
4      NaN      Canada

      Release Date Rating  Duration Imdb Score Content Type      Date Added
0      2020.0      TV-MA      1 Season      6.6/10      TV Show      NaN
1      2020.0      TV-MA      99 min      6.2/10      Movie      September 8, 2020
2      2019.0      TV-14      95 min      6.4/10      Movie      July 1, 2020
3      2020.0      TV-MA      1 Season      6.6/10      TV Show      NaN
4      2020.0      TV-14      90 min      5.1/10      Movie      February 5, 2020
Index([ 'Show Id', 'Title', 'Description', 'Director', 'Genres', 'Cast',
      'Production Country', 'Release Date', 'Rating', 'Duration',
      'Imdb Score', 'Content Type', 'Date Added'],
      dtype='object')

In [31]: genres_column = "Title"

In [32]: data[genres_column] = data[genres_column].apply(lambda x: x.replace(' ', '').lower())

In [33]: count_vectorizer = CountVectorizer(tokenizer=lambda x: x.split(','), preprocessor=lambda x: x)
count_matrix = count_vectorizer.fit_transform(data[genres_column])

/Users/tavi/anaconda3/lib/python3.11/site-packages/sklearn/feature_extraction/text.py:525: UserWarning: The parameter 'token_pattern' will not be used since 'tokenizer' is not None
  warnings.warn(

In [36]: similarity = cosine_similarity(count_matrix, count_matrix)

In [38]: indices = pd.Series(data.index, index=data["Title"]).drop_duplicates()

In [39]: def netFlix_recommendation(title, similarity=similarity):
    if title not in indices:
        return "Title not found in the dataset."

    index = indices[title]
    similarity_scores = list(enumerate(similarity[index]))
    similarity_scores = sorted(similarity_scores, key=lambda x: x[1], reverse=True)
    similarity_scores = similarity_scores[1:11] # Skip the first item as it is the item itself
    movie_indices = [i[0] for i in similarity_scores]

    return data['title'].iloc[movie_indices]

In [40]: print(netFlix_recommendation("Girlfriend's Day"))

Title not found in the dataset.

In [41]: print(netFlix_recommendation("girlfriend"))

Title not found in the dataset.

In [ ]: 
```