УДК 004

# Создание системы рекомендаций фильмов и сериалов при помощи Google Colaboratory

*Анишкова Анастасия Сергеевна*
*Приамурский государственный университет имени Шолом-Алейхема*
*Студент*

**Аннотация**
Целью исследования является создание системы рекомендаций фильма или сериала. Для реализации использовалась бесплатный облачный сервис, который предоставляет пользователям возможность работать с Python Google Colaboratory. Полученный результат можно использовать для подбора похожего контента.
**Ключевые слова:** Google Colaboratory, система рекомендаций, синусное сходство

# A system of recommendations for movies and TV shows using Google Coollaboratory

*Anishkova Anastasia Sergeevna*
*Sholom Aleichem Priamurskiy State University*
*Student*

**Abstract**
The purpose of the study is to create a recommendation system for a movie or TV series. For the implementation, a free cloud service was used, which provides users with the opportunity to work with Python Google Colaboratory. The result can be used to select similar content.
**Key words:** Google Colaboratory, recommendation system, sine similarity

## 1 Введение

### 1.1 Актуальность

Основная идея системы рекомендаций заключается в том, чтобы использовать алгоритмы машинного обучения для анализа предпочтений пользователя и предоставления ему рекомендаций на основе этих предпочтений. Это может быть особенно полезно, если пользователь уже просмотрел много фильмов или сериалов и хочет найти что-то новое, что ему может понравиться.

### 1.2 Обзор исследований

В. И. Федоренко, В. С. Киреев проанализировали подходов к построению гибридных рекомендательных систем в задаче рекомендации фильмов. В данной работе приводится сравнение некоторых методов построения гибридных рекомендаций для улучшения качества рекомендаций

[1], в рамках данной работы разработана простая система рекомендации фильмов на основе контекстного подхода и коллаборативной фильтрации. Для рекомендации на основе контекстного подхода использовалось косинусное сходство между фильмами, а для предсказания оценки пользователя алгоритм SVD создали А. Г. Викторенко, Е. В. Казаковцева [2], В. И. Федоренко, В. С. Киреев описали использование методов векторизации текстов на естественном языке для повышения качества контентных рекомендаций фильмов Рекомендательные системы становятся незаменимыми компонентами любой веб-системы, предлагающей пользователям контент. Одной из актуальных задач в области построения контентной фильтрации является задача автоматического формирования признакового описания объектов системы. Для составления признаков текста, например аннотации фильма, требуется специальный метод предобработки–векторизация. В данной работе приводится сравнение методов построения векторных [3], систему рекомендаций фильмов на основе DeepFM создал Ш. Жэнь [4].

## 2 Цель исследования

Создания системы рекомендаций фильмов и сериалов в Google Coolaboratory заключается в разработке алгоритма, который будет анализировать предпочтения пользователей и предлагать им фильмы и сериалы, которые они, скорее всего, оценят.

## 3 Материалы и методы

В данном исследование используется Google Colab — это бесплатная среда для разработки и выполнения программного кода в облаке. Она предоставляет возможность писать и запускать код на языке Python, используя только браузер, без установки специальных программ на компьютер. Google Colab основан на Jupyter Notebook, популярном ПО для написания и запуска кода.

## 4 Результаты

Подключим библиотеки для реализации анализа данных

Ссылка на датасет https://cloud.mail.ru/public/Rsen/Mb9a3eGrK

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Рисунок 1 – Подключение библиотек

Подгружаем данные и выводим на экран. По таблице видим, что датасет состоит из 12 признаков (см. рис.2)

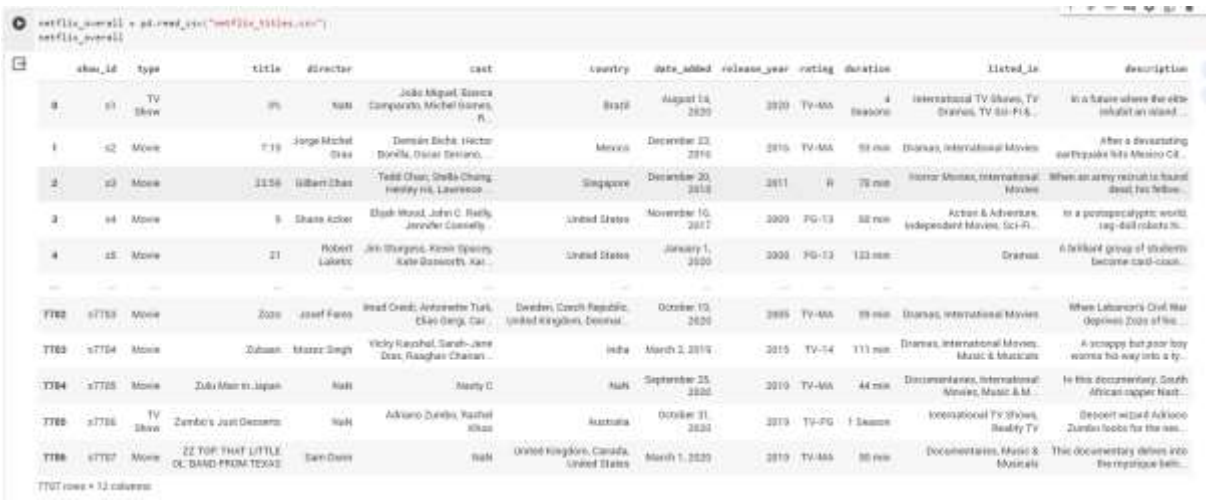


Рисунок 2 – Считывание данных

Далее разобьем данные на две группы: фильмы и сериалы и изобразим график, чтобы было понятно какой категории больше. По графику можем сделать вывод, что фильмов гораздо больше, чем сериалов на платформе Netflix (см. рис.3).


Рисунок 3 – График «фильмы и сериалы»

Теперь посмотрим, сколько времени длятся фильмы, сериалы и визуализируем на графики при помощи библиотеки plotly (см. рис.4).

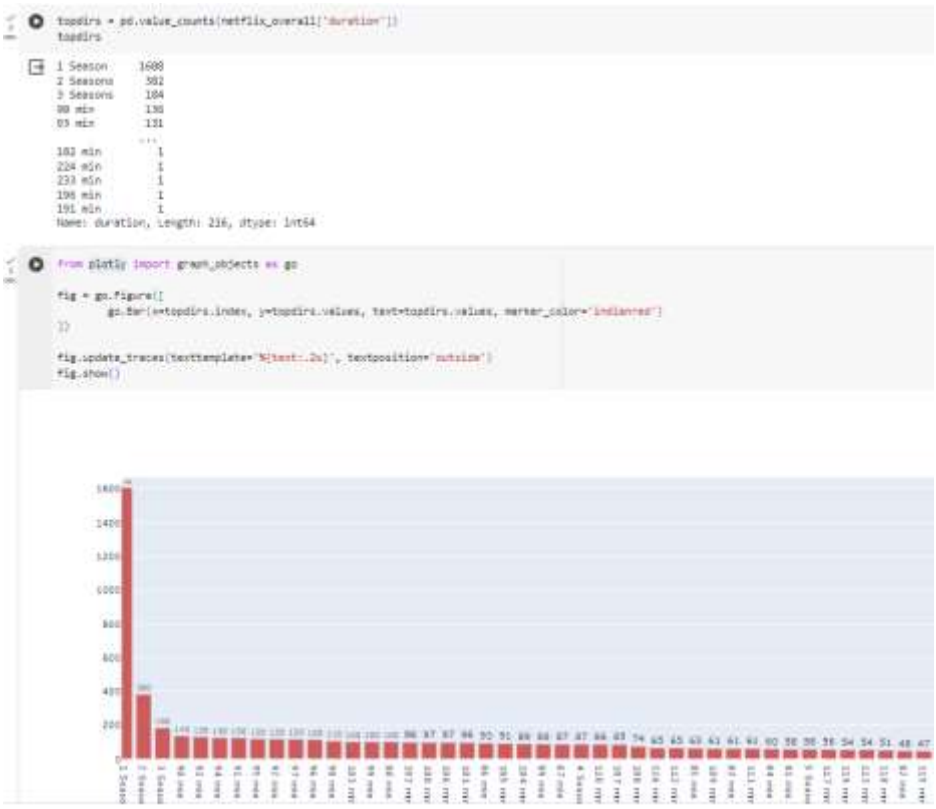
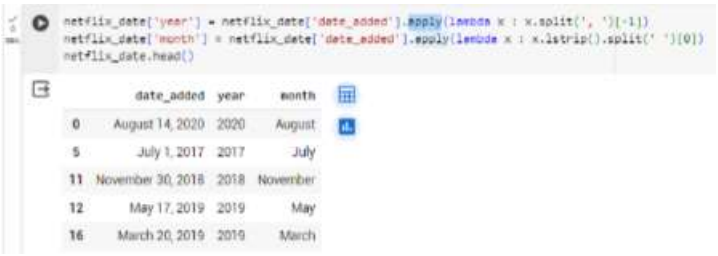



Рисунок 4 – График «Длительность фильмов и сериалов»

Следующим шагом посмотрим, какой месяц положительный для выпуска фильма или сериала. Для этого необходимо убедиться, что в данных нет пропусках. В этом случае в данных присутствуют пропуски, поэтому их нужно удалить. После проведения таких манипуляций выведем 5 записей с содержанием информации о дате на экран (см. рис.5).


Рисунок 5 – Удаление пропусков из данных и вывод данных

Дату разделим на месяц и год при помощи метода функции apply и выведем данные на экран (см. рис.6).

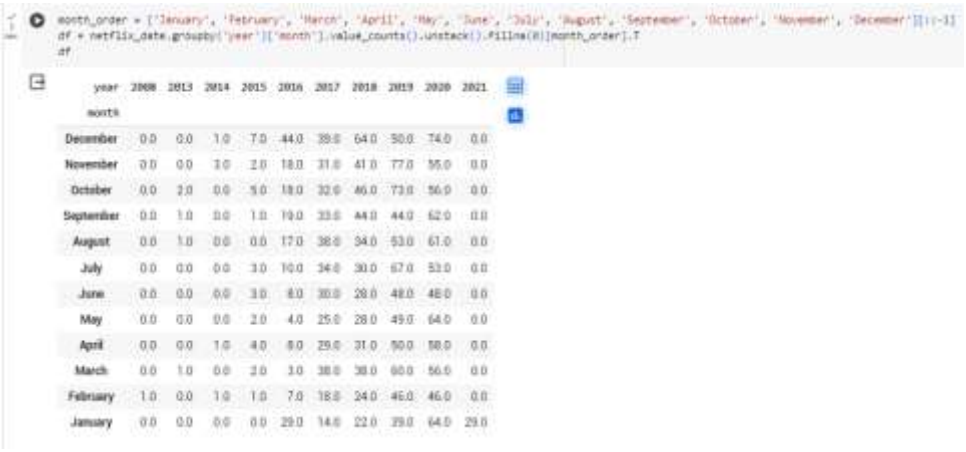


```
netflix_date['year'] = netflix_date['date_added'].apply(lambda x : x.split(', ')[-1])
netflix_date['month'] = netflix_date['date_added'].apply(lambda x : x.lstrip().split(' ')[0])
netflix_date.head()
```

| | date_added | year | month |
|---|---|---|---|
| 0 | August 14, 2020 | 2020 | August |
| 5 | July 1, 2017 | 2017 | July |
| 11 | November 30, 2018 | 2018 | November |
| 12 | May 17, 2019 | 2019 | May |
| 16 | March 20, 2019 | 2019 | March |

Рисунок 6 – Разделение даты

Далее сделаем сводную таблицу, где отобразим данные по месяцу и году, на пересечение отображено сколько фильмов было выпущено (см. рис.7).



```
month_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December'][::-1]
df = netflix_date.groupby('year')['month'].value_counts().unstack().fillna(0)[month_order].T
df
```

| year | 2008 | 2013 | 2014 | 2015 | 2016 | 2017 | 2018 | 2019 | 2020 | 2021 |
|---|---|---|---|---|---|---|---|---|---|---|
| month | | | | | | | | | | |
| December | 0.0 | 0.0 | 1.0 | 7.0 | 44.0 | 39.0 | 64.0 | 50.0 | 74.0 | 0.0 |
| November | 0.0 | 0.0 | 1.0 | 2.0 | 18.0 | 31.0 | 41.0 | 77.0 | 55.0 | 0.0 |
| October | 0.0 | 2.0 | 0.0 | 5.0 | 18.0 | 32.0 | 46.0 | 73.0 | 56.0 | 0.0 |
| September | 0.0 | 1.0 | 0.0 | 1.0 | 19.0 | 33.0 | 44.0 | 44.0 | 62.0 | 0.0 |
| August | 0.0 | 1.0 | 0.0 | 0.0 | 17.0 | 38.0 | 54.0 | 53.0 | 61.0 | 0.0 |
| July | 0.0 | 0.0 | 0.0 | 3.0 | 10.0 | 34.0 | 30.0 | 67.0 | 53.0 | 0.0 |
| June | 0.0 | 0.0 | 0.0 | 3.0 | 8.0 | 30.0 | 28.0 | 48.0 | 48.0 | 0.0 |
| May | 0.0 | 0.0 | 0.0 | 2.0 | 4.0 | 25.0 | 28.0 | 45.0 | 64.0 | 0.0 |
| April | 0.0 | 0.0 | 1.0 | 4.0 | 8.0 | 29.0 | 31.0 | 50.0 | 58.0 | 0.0 |
| March | 0.0 | 1.0 | 0.0 | 2.0 | 3.0 | 38.0 | 38.0 | 60.0 | 56.0 | 0.0 |
| February | 1.0 | 0.0 | 1.0 | 1.0 | 7.0 | 18.0 | 24.0 | 46.0 | 46.0 | 0.0 |
| January | 0.0 | 0.0 | 0.0 | 0.0 | 29.0 | 14.0 | 22.0 | 39.0 | 64.0 | 29.0 |

Рисунок 7 – Сводная таблица

Теперь визуализирую сводную таблицу, можно сделать вывод, что в 2019 году в январе и декабре было выпущено достаточно мало контента, хотя эти месяцы очень прибыльные (см. рис.8).

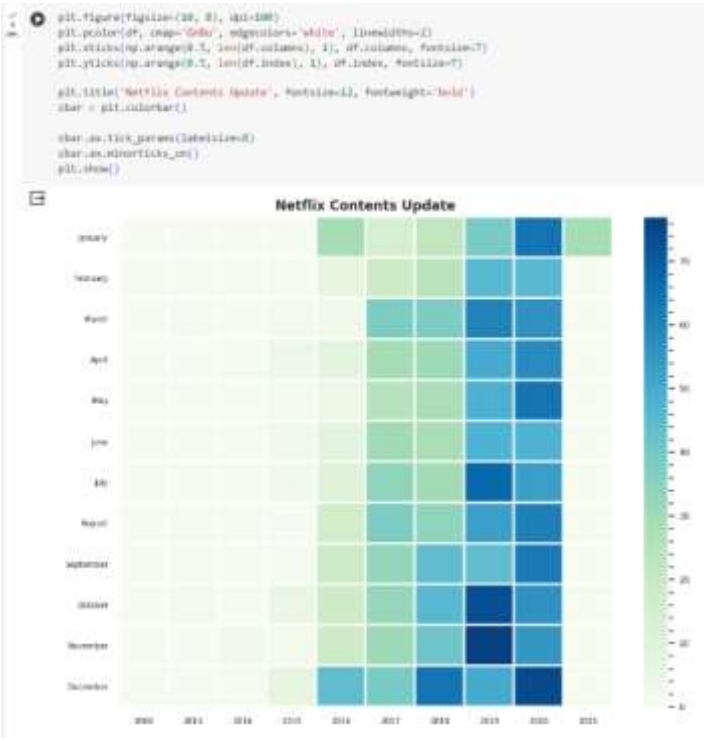



Рисунок 8 – Визуализация сводной таблицы

Далее посмотрим топ 15 фильмов (см. рис.9).

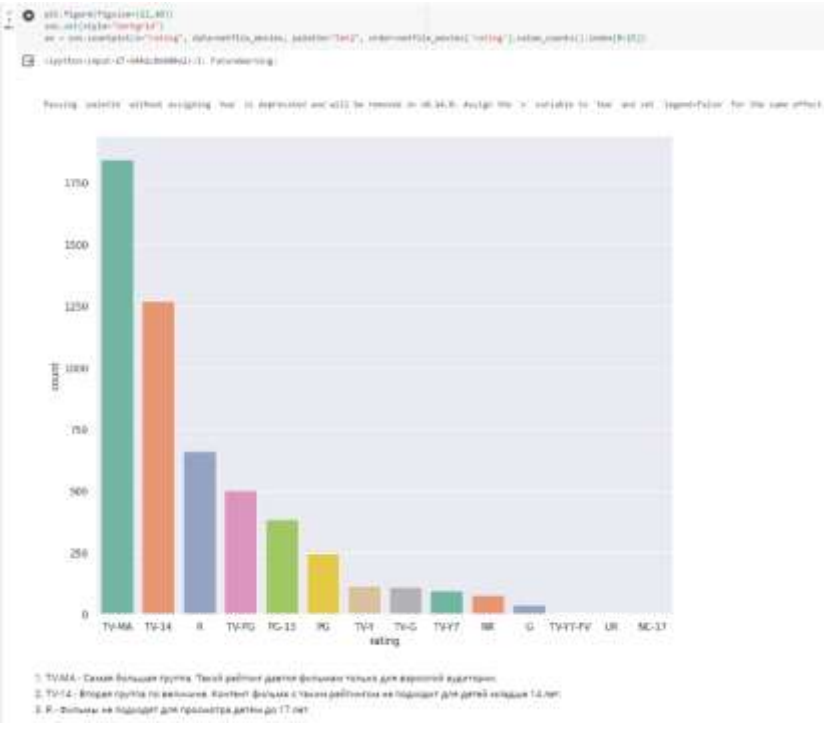



Рисунок 9 – График «Топ 15 фильмов»

Теперь отобразим, в каком году чаще выпускались фильмы, по результатам можно сделать вывод, что в 2027 году больше всего было выпущено фильмов (см. рис.10)

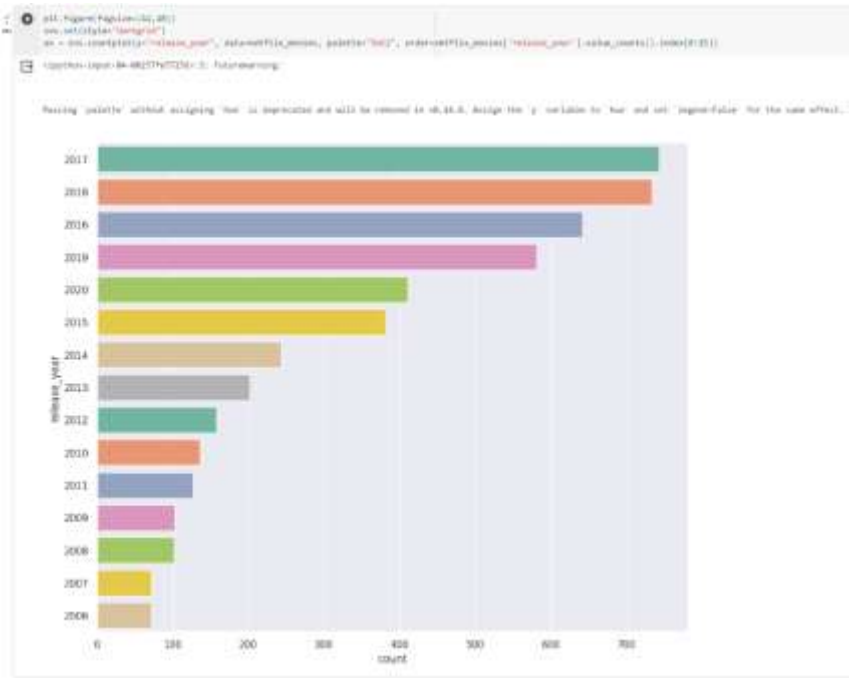



Рисунок 10 – График «Анализ года выпуска»

Далее выведем топ 10 стран, которые выпускают фильмы (см. рис.11).

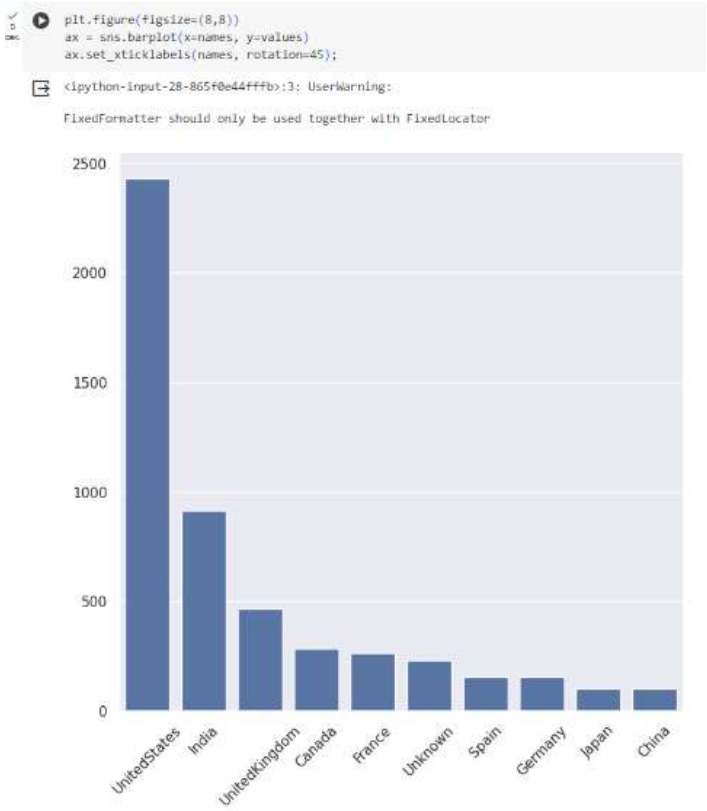



Рисунок 11 – Топ 10 стран

Следующий график будем о том, какая продолжительность фильмы самая популярная. Для отображения гистограммы необходимо заменить значения min на пустоту. Меняем строковое значение на целочисленное значение. После завершения улучшения данных визуализируем их (см.

рис.12). По результатам графика видно, что самая частотная длительность – это 75-120 мин.

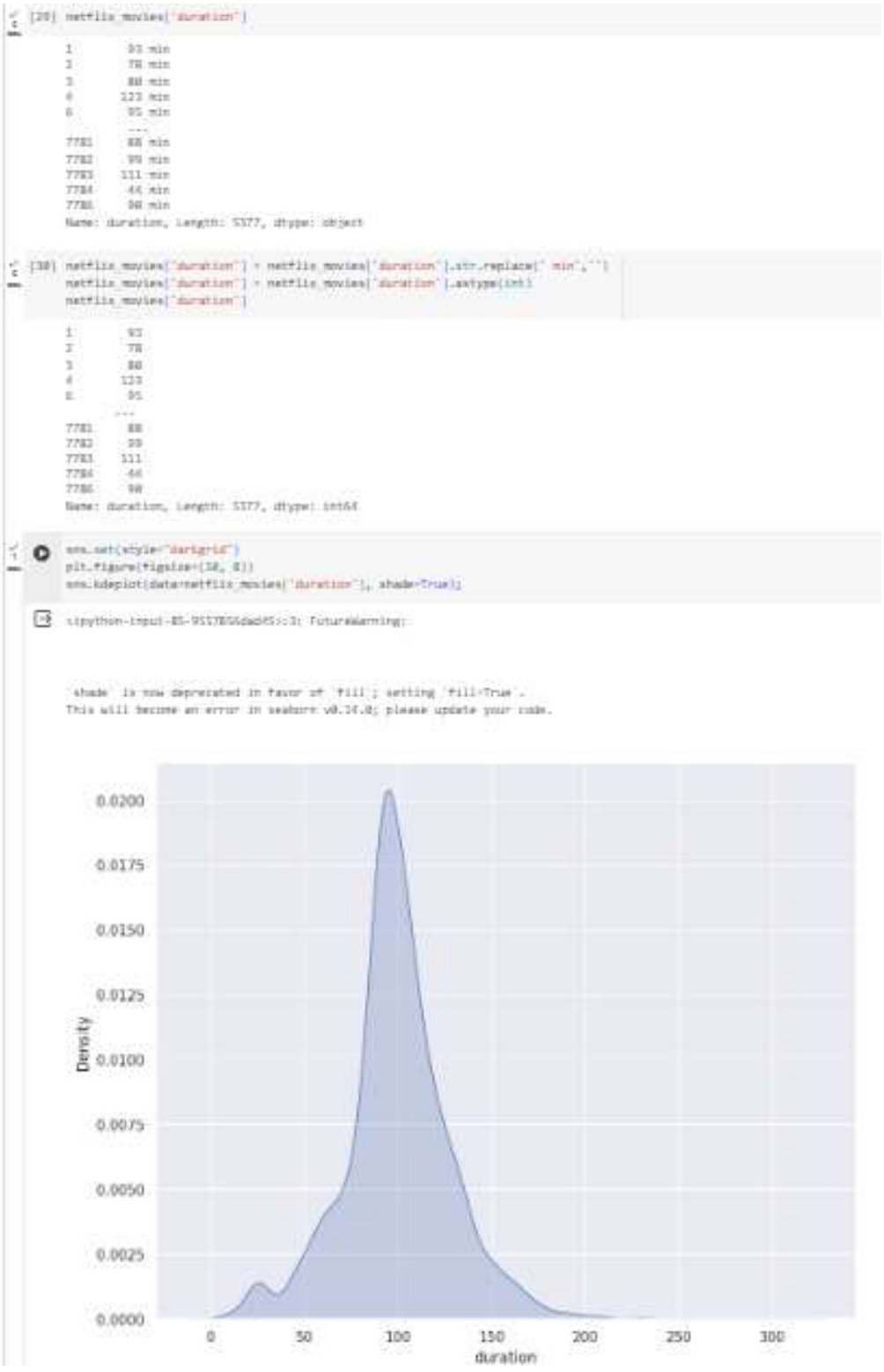



Рисунок 12 – Визуализация частотности длительности фильмов

Проанализируем, какие самые частотные жанры фильмов. Можно сделать вывод, что самые топовые жанры это: интернациональные фильмы, драмы и комедии (см. рис. 13).

Приступим к анализу сериалов, для начала посмотрим какие страны, выпускали сериалы (см. рис.13).

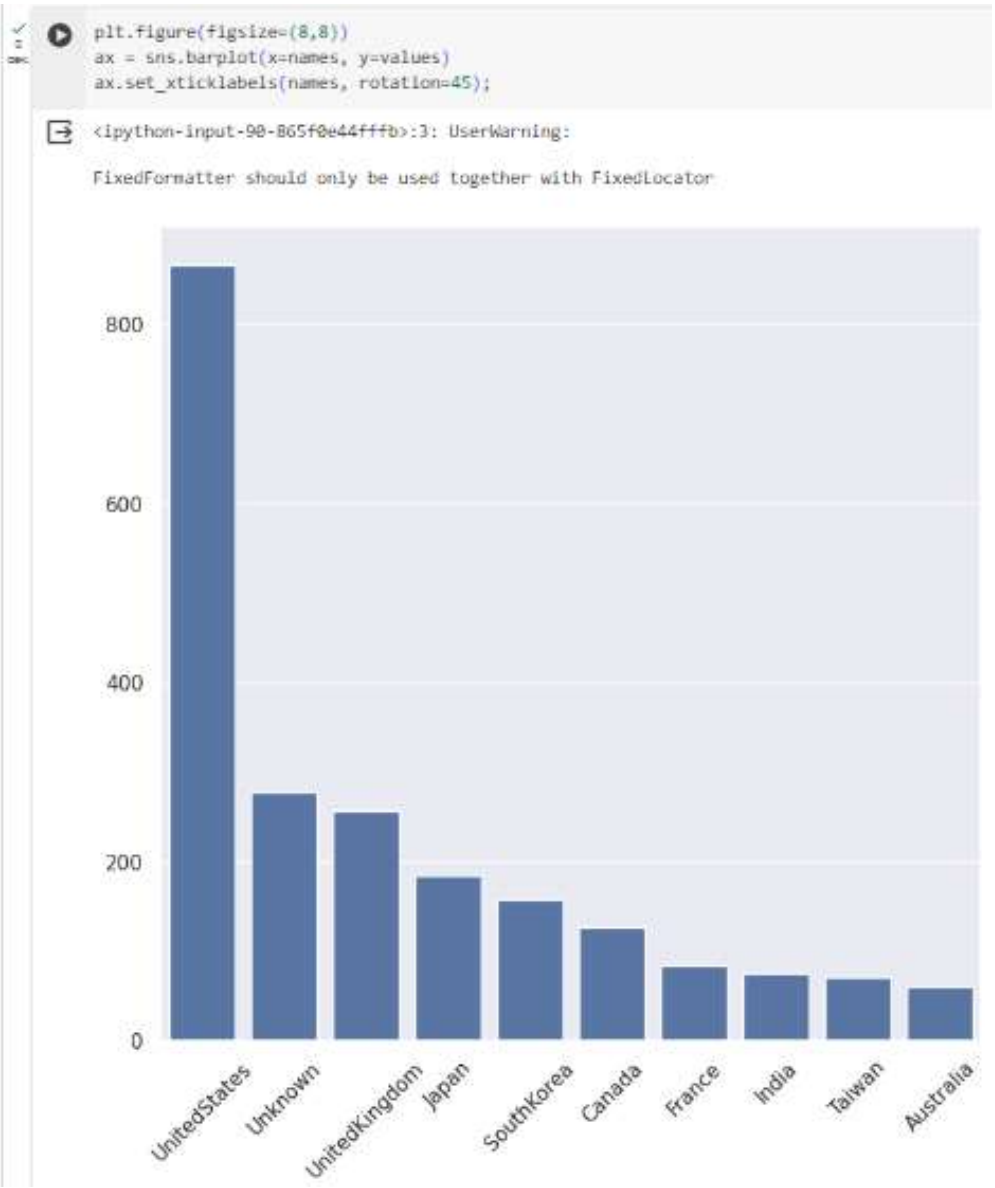



Рисунок 14 – Анализ стран, выпускающих сериалы

Выведем на экран сериалы с большим количеством сезонов (см. рис.15).

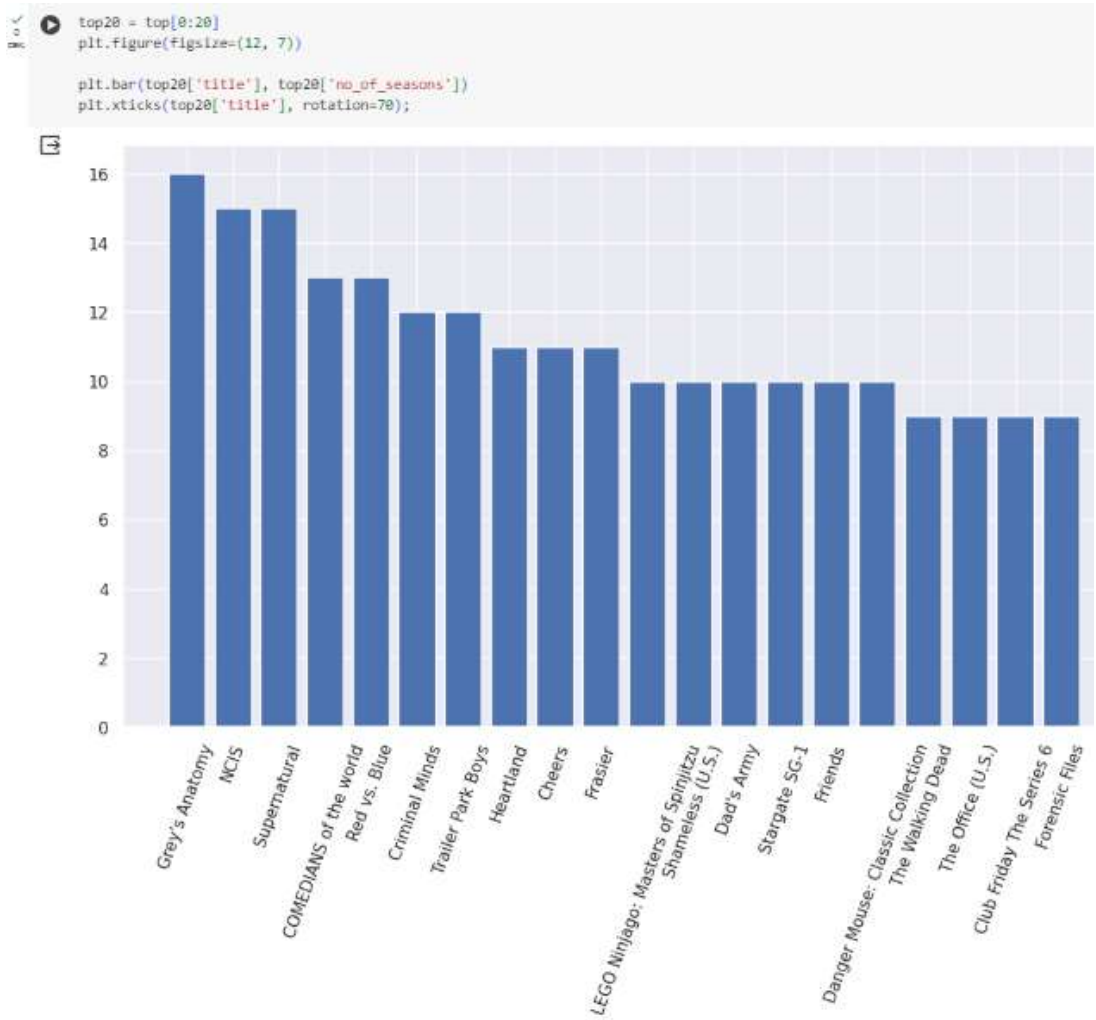



Рисунок 15 – Сериалы с большим количеством сезонов

А теперь отобразим сериалы с наименьшим количеством серий (см. рис. 16).

Рисунок 16 – Сериалы с наименьшим количеством серий

Визуализируем, какой жанр сериалов популярнее (см. рис.16). По результатам можно сделать вывод, что Интернациональные сериалы, драмы и комедии - топовые жанры.

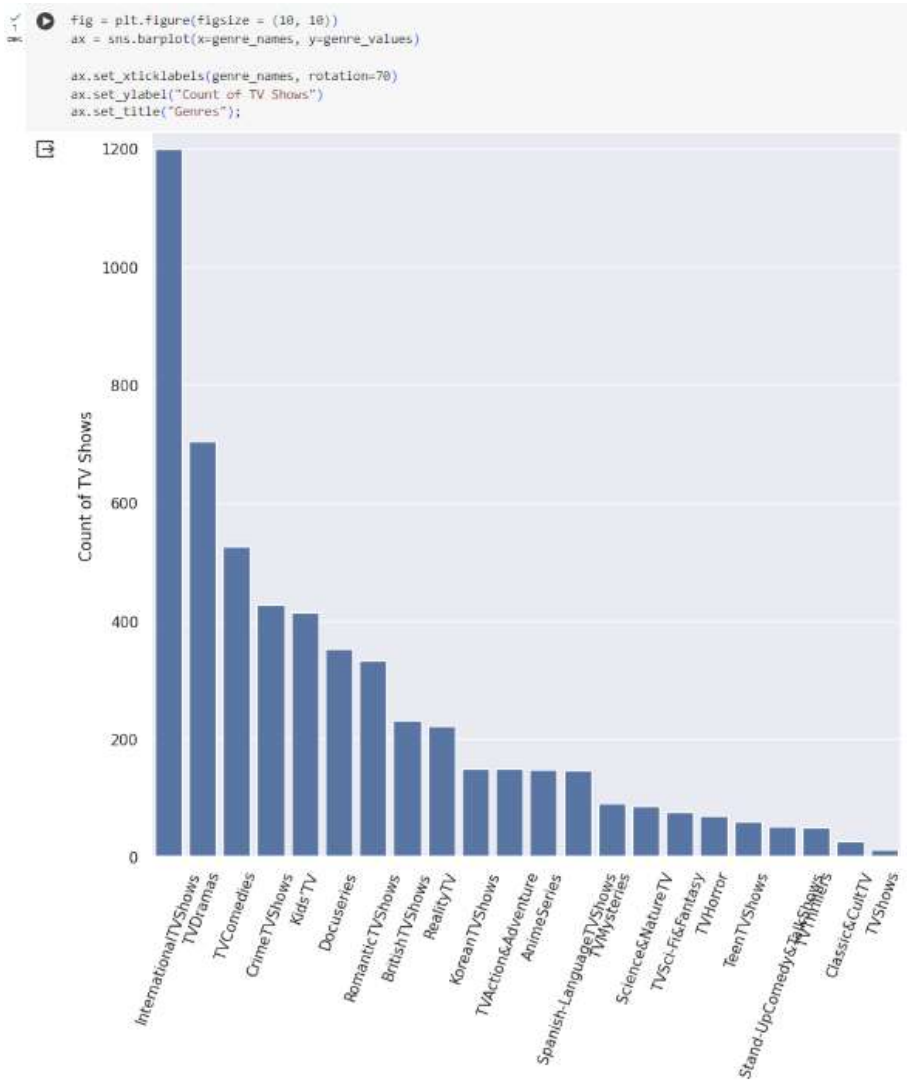



Рисунок 16 – Жанры сериалов

Далее проанализируем продолжительность сериалов (см. рис.17). Самый частотное значение – это 1 сезон.

```
sns.set(style="darkgrid")
plt.figure(figsize=(10, 8))
sns.kdeplot(data=durations['no_of_seasons'], shade=True)
plt.xticks(sorted(durations['no_of_seasons'].unique()));
```

```
<ipython-input-63-b8a2a24c5a12>:3: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.
```

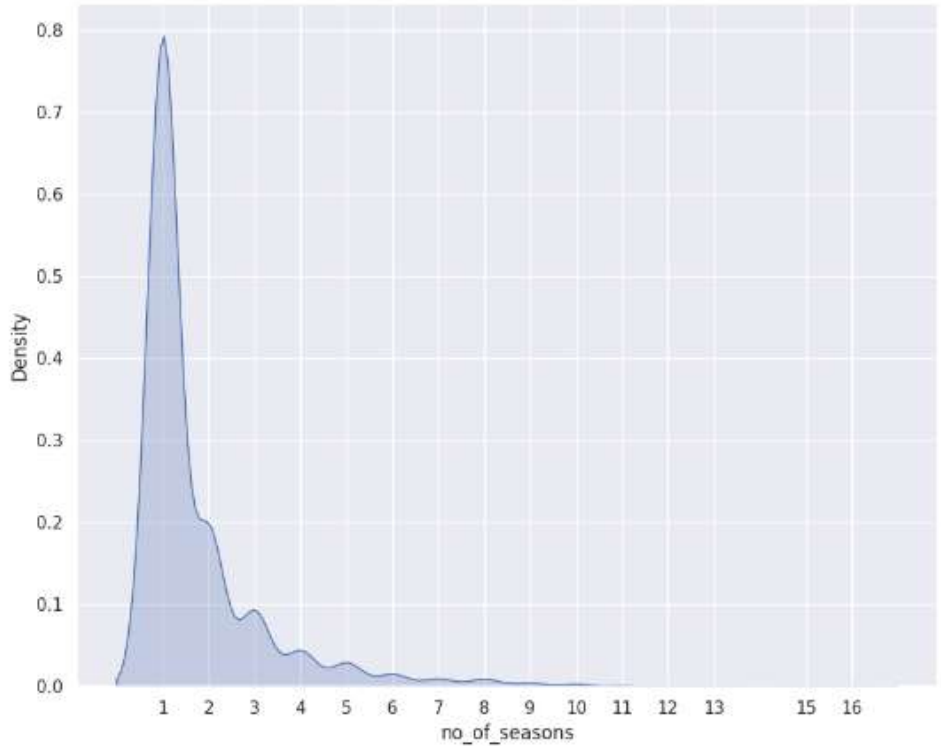



Рисунок 17 – Продолжительность сериалов

Анализ фильмов и сериалов окончен, приступим к системе рекомендаций. Система рекомендаций будет основана на контенте и его описание, который просматривает пользователь. Для этого выведем описание фильмов и сериалов (см. рис. 18).

```
netflix_movies['description']
```

```
1       After a devastating earthquake hits Mexico Cit...
2       When an army recruit is found dead, his fellow...
3       In a postapocalyptic world, rag-doll robots hi...
4       A brilliant group of students become card-coun...
6       After an awful accident, a couple admitted to ...
                              ...
7781    Dragged from civilian life, a former superhero...
7782    When Lebanon's Civil War deprives Zozo of his ...
7783    A scrappy but poor boy worms his way into a ty...
7784    In this documentary, South African rapper Nast...
7786    This documentary delves into the mystique behi...
Name: description, Length: 5377, dtype: object
```

Рисунок 18 – Описание фильмов и сериалов

Для дальнейших действий нам поможет способ «Мешок слов». Мешок слов — упрощенное представление текста, которое используется в обработке естественных языков и информационном поиске. Для

осуществления используем библиотеку sklearn, заполним пропуски в описание и исключим слова, которые не несут никакого смысла, это предлоги, местоимения и артикли (см. рис.19).

```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
```

```
netflix_movies['description'].isna().sum()
```

```
0
```

```
[67] # netflix_movies['description'] = netflix_movies['description'].fillna('')
     # netflix_movies['description'].isna().sum()
```

```
[68] netflix_movies['description'].head()
```

```
1    After a devastating earthquake hits Mexico Cit...
2    When an army recruit is found dead, his fellow...
3    In a postapocalyptic world, rag-doll robots hi...
4    A brilliant group of students become card-coun...
6    After an awful accident, a couple admitted to ...
Name: description, dtype: object
```

```
[69] tfidf = TfidfVectorizer(stop_words='english')
     tfidf_matrix = tfidf.fit_transform(netflix_movies['description'])

     tfidf_matrix.shape
```

```
(5377, 14601)
```

```
[70] tfidf_matrix
```

```
<5377x14601 sparse matrix of type '<class 'numpy.float64'>'
        with 73930 stored elements in Compressed Sparse Row format>
```

Здесь 14601 слов, которые описывают 5377 фильмов.

Рисунок 19 – Метод «Мешок слов»

Далее используем метод «Косинусная похожесть» при помощи библиотеки sklearn (см. рис.20). Косинусное сходство — это показатель, используемый для измерения того, насколько похожи два элемента. Математически он измеряет косинус угла между двумя векторами, проецируемыми в многомерное пространство. Выходное значение находится в диапазоне 0–1. 0 означает отсутствие сходства, а 1 означает, что оба элемента на 100% похожи.

```
from sklearn.metrics.pairwise import cosine_similarity

cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)
cosine_sim
```

```
array([[1.        , 0.        , 0.        , ..., 0.        , 0.09305242,
        0.        ],
       [0.        , 1.        , 0.        , ..., 0.        , 0.        ,
        0.        ],
       [0.        , 0.        , 1.        , ..., 0.07593931, 0.        ,
        0.        ],
       ...,
       [0.        , 0.        , 0.07593931, ..., 1.        , 0.        ,
        0.        ],
       [0.09305242, 0.        , 0.        , ..., 0.        , 1.        ,
        0.02580113],
       [0.        , 0.        , 0.        , ..., 0.        , 0.02580113,
        1.        ]])
```

Рисунок 20 – Метод «Косинусная похожесть»

Преобразуем сводную таблицу, где по индексам и столбца наименование фильмов, а по пересечению степень их похожести (см. рис.21).

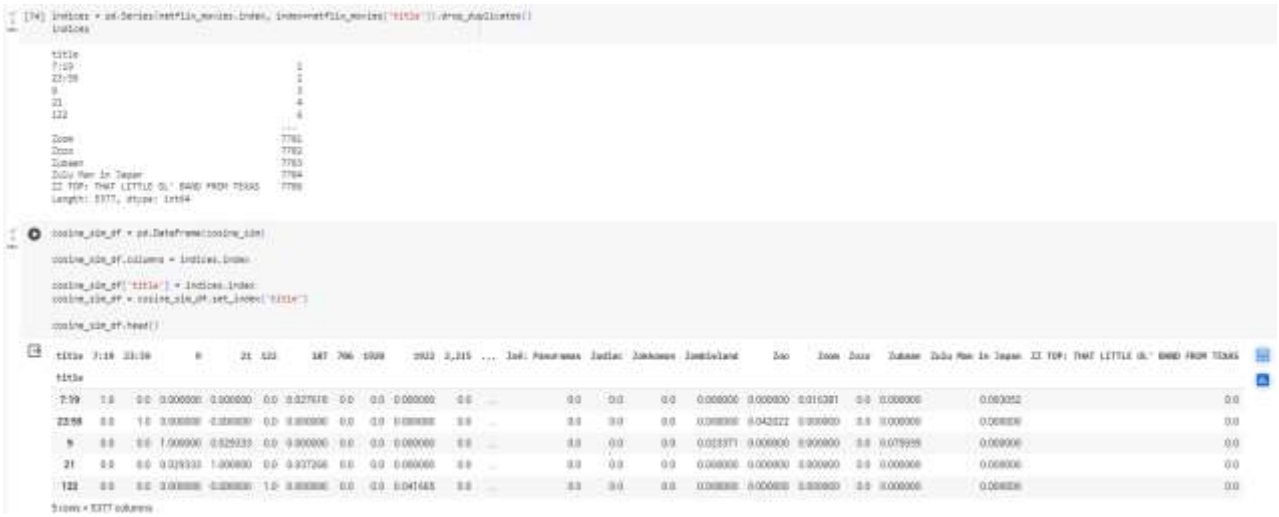


Рисунок 21 – Сводная таблица похожести фильмов и сериалов

Основываясь на этих похожестях, сформируем рекомендации (см. рис.22)

```
def get_recommendations(title, cosine_sim=cosine_sim_df):
    idx = indices[title]
    # Получаем похожести для этого фильма
    sim_scores = list(enumerate(cosine_sim_df.loc[title]))

    # Сортируем фильмы, основываясь на пох
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:11]
    # Получаем индексы фильма
    movie_indices = [i[0] for i in sim_scores]

    return netflix_movies['title'].iloc[movie_indices]
```

Loading...

Рисунок 21 – Формирование рекомендаций

Далее проверим работу системы организаций (см. рис.22). Для примера возьмем название фильма «Last Breath» и выведем описание фильма. Фильм о профессиональном дайвере, который оказывается в ловушке на дне океана с истощающимся запасом кислорода и малой надеждой на своевременное спасение, поэтому он пытается спастись сам.

```
netflix_movies[netflix_movies['title'] == 'Last Breath'].description.iloc[0]
```

```
'A commercial diver becomes trapped on the ocean floor with dwindling oxygen and little hope of a timely rescue, so he tries to save himself.'
```

```
get_recommendations('Last Breath')
```

```
2972                                    In The Deep
4111                                         Mine 9
1325                                  Chicken Little
4537                        NOVA: Thai Cave Rescue
4332    My Little Pony Equestria Girls: Legend of Ever...
985                                   Big Miracle
4916                                        Posesif
5411                                  Savage Raghda
1846                          Dragon Quest Your Story
5289                                  Rowdy Rathore
Name: title, dtype: object
```

Рисунок 22 – рекомендации фильму, который взят для примера

Проверим описание фильма, который первый в списке рекомендаций – «In The Deep» (см. рис.23). Описание фильма – «С небольшим количеством кислорода в баллонах для подводного плавания две сестры оказались запертыми в клетке с акулами на дне океана, в то время как большие белые акулы кружат поблизости».


Рисунок 23 – Описание фильма

Можно сделать вывод, что рекомендованный фильм действительно похож на заданный фильм, ведь в обоих фильмах присутствие тема океана и трагедии.

**Библиографический список**

1. Федоренко В. И., Киреев В. С. Анализ подходов к построению гибридных рекомендательных систем в задаче рекомендации фильмов //Теория. Практика. Инновации. 2017. №. 6. С. 44-50.
2. Викторенко А. Г., Казаковцева Е. В. Разработка системы рекомендаций фильмов на python //Прикладная математика: современные проблемы математики, информатики и моделирования. 2022. С. 301-305.
3. Федоренко В. И., Киреев В. С. Использование методов векторизации текстов на естественном языке для повышения качества контентных рекомендаций фильмов //Современные наукоемкие технологии. 2018. №. 3. С. 102-106.
4. Жэнь Ш. Система рекомендаций фильмов на основе DeepFM: магистерская диссертация по направлению подготовки: 01.04.02-Прикладная математика и информатика. 2023.