Capstone Project -

Netflix Movie Recommender

Netflix Movie Recommender -

The Billion Dollar Algorithm

- The goal of this project is to build a recommender system to help users find movies and shows that fit their preferences and tastes so that they will be more likely to maintain a subscription with Netflix.

- Due to user privacy issues and the unavailability of datasets containing both movie content details and user ratings, we would be using *two* separate datasets to build *two* distinct recommender systems - **Content-Based Filtering Model** and **Collaborative Filtering Model**.

- This has been a core focus for Netflix in recent years. Based on an academic paper penned by Gomez-Uribe and Netflix's Chief Product Officer Neil Hunt, they assert that 'the combined effect of personalization and recommendations is estimated to save them more than $1 billion per year.' This would be a *significant* area of cost savings for the tech giant as it is expected to spend more than $17 billion on global content for the current financial year.

90 Seconds of Truth

- Based on findings from Netflix - “Consumer research suggests that a typical Netflix member loses interest after perhaps 60 to 90 seconds of choosing, having reviewed 10 to 20 titles (perhaps 3 in detail) on one or two screens. The user either finds something of interest or the risk of the user abandoning our service increases substantially.”
- Furthermore, Netflix estimates that only 20% of its subscriber video choices come from search, with the other 80% coming from recommendations. So it’s crucial that Netflix gets its recommendation system right.
- The end goal of the engine is “moments of truth,” when “a member starts a session and we help that member find something engaging within a few seconds, preventing abandonment of our service for an alternative entertainment option.”

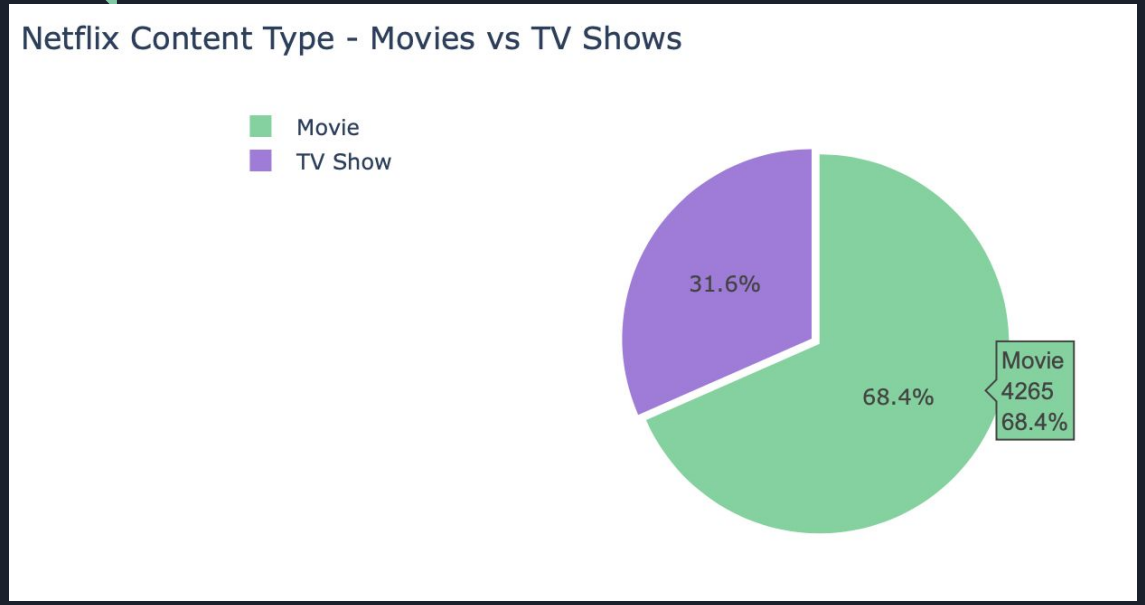
Description of Dataset

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	81145628	Movie	Norm of the North: King Sized Adventure	Richard Finn, Tim Maltby	Alan Marriott, Andrew Toth, Brian Dobson, Cole Howard, Jennifer Cameron, Jonathan Holmes, Lee Tockar, Lisa Durupt, Maya Kay, Michael Dobson	United States, India, South Korea, China	September 9, 2019	2019	TV-PG	90 min	Children & Family Movies, Comedies	Before planning an awesome wedding for his grandfather, a polar bear king must take back a stolen artifact from an evil archaeologist first.
1	80117401	Movie	Jandino: Whatever it Takes	NaN	Jandino Asporaat	United Kingdom	September 9, 2016	2016	TV-MA	94 min	Stand-Up Comedy	Jandino Asporaat riffs on the challenges of raising kids and serenades the audience with a rousing rendition of "Sex on Fire" in his comedy show.
2	70234439	TV Show	Transformers Prime	NaN	Peter Cullen, Sumalee Montano, Frank Welker, Jeffrey Combs, Kevin Michael Richardson, Tania Gunadi, Josh Keaton, Steve Blum, Andy Pessoa, Ernie Hudson, Daran Norris, Will Friedle	United States	September 8, 2018	2013	TV-Y7-FV	1 Season	Kids' TV	With the help of three human allies, the Autobots once again protect Earth from the onslaught of the Decepticons and their leader, Megatron.

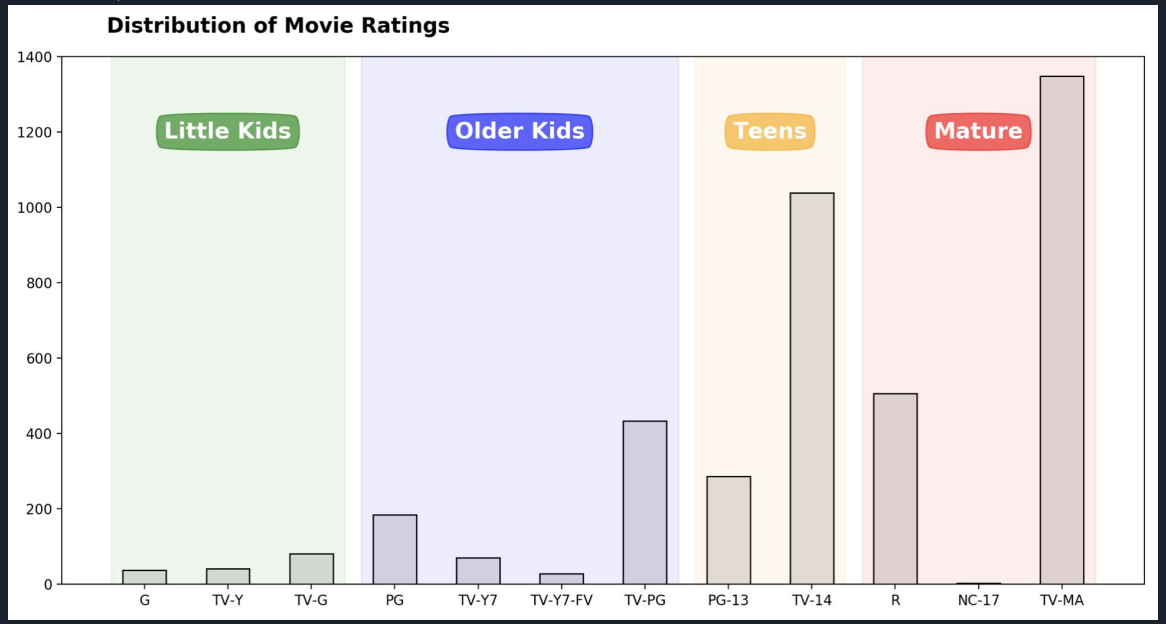
The columns of the dataset consists of:

- show_id - Unique ID for every Movie / Tv Show
- typeIdentifier - 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
- listed_in - Genre
- description - The summary description

Netflix Content Type



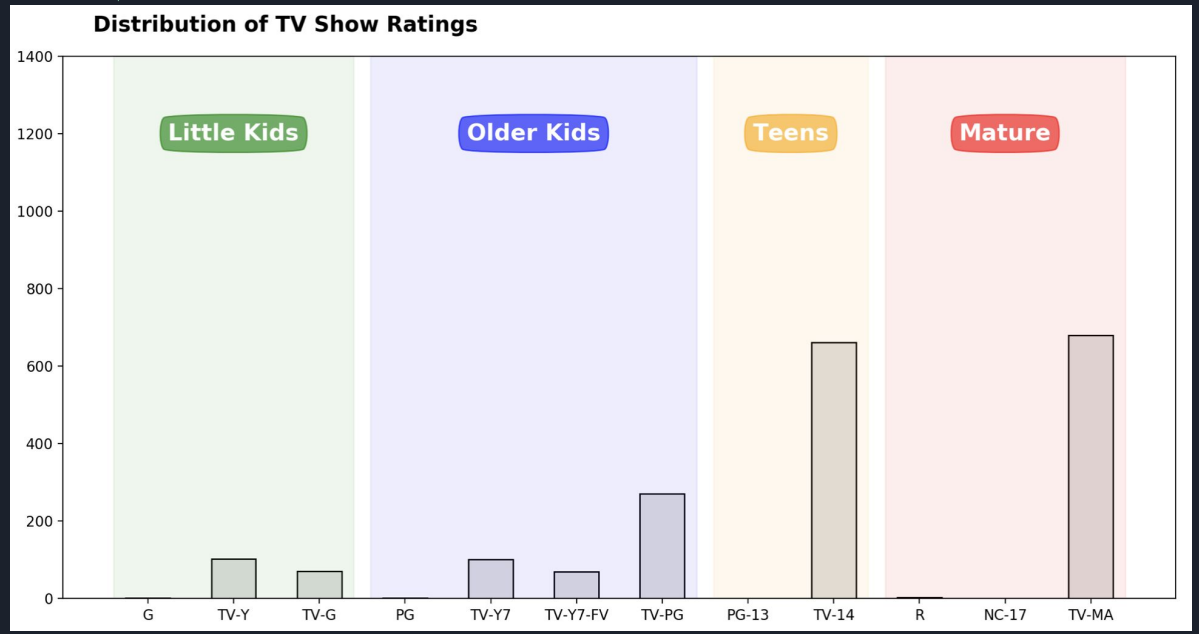
Distribution of Movie Ratings



USA Movie Ratings

- G – *General Audiences*
- PG – *Parental Guidance Suggested*
- PG-13 – *Parents Strongly Cautioned*
- R – *Restricted*
- NC-17 – *Adults Only*

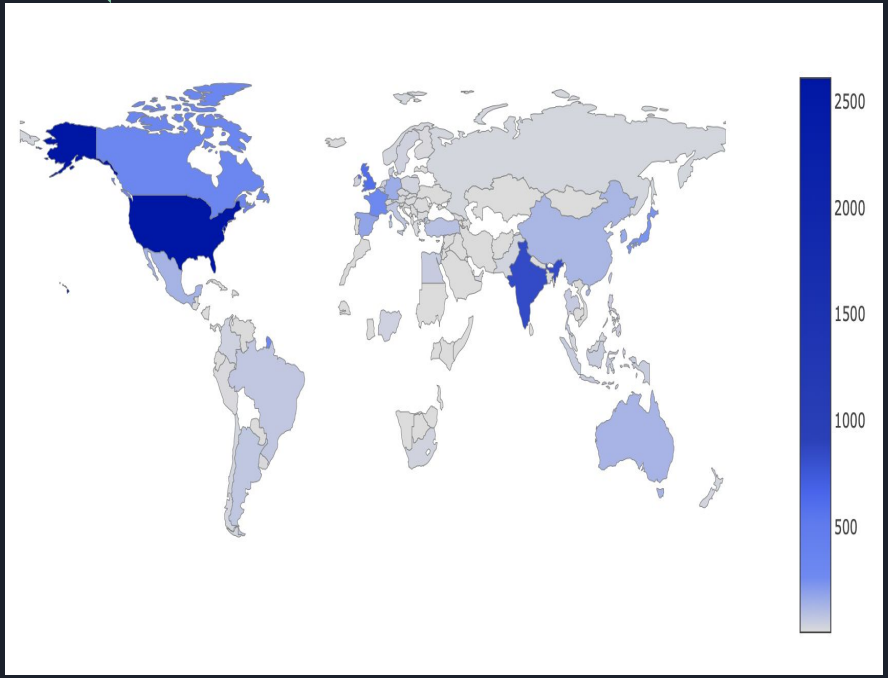
Distribution of TV Show Ratings

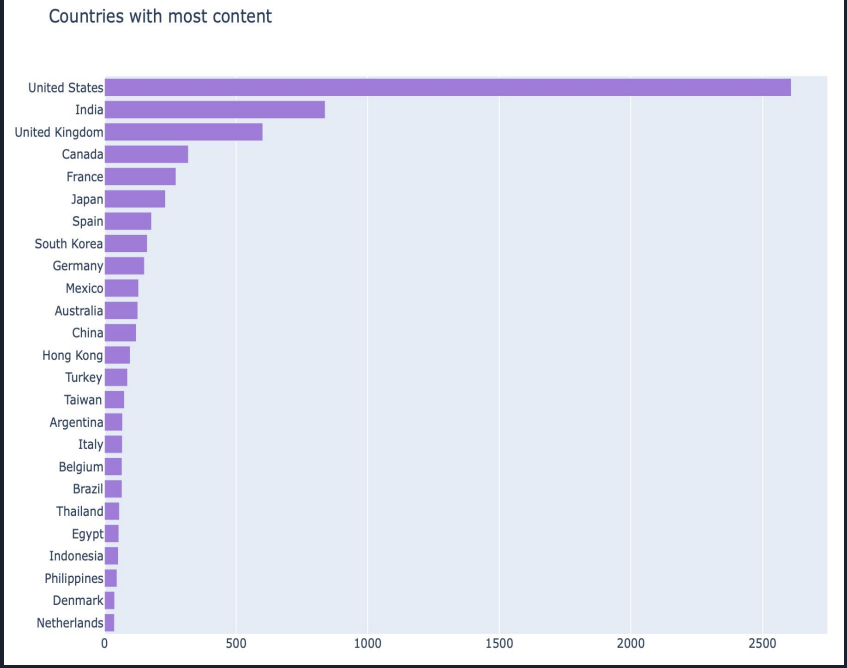


USA TV Ratings

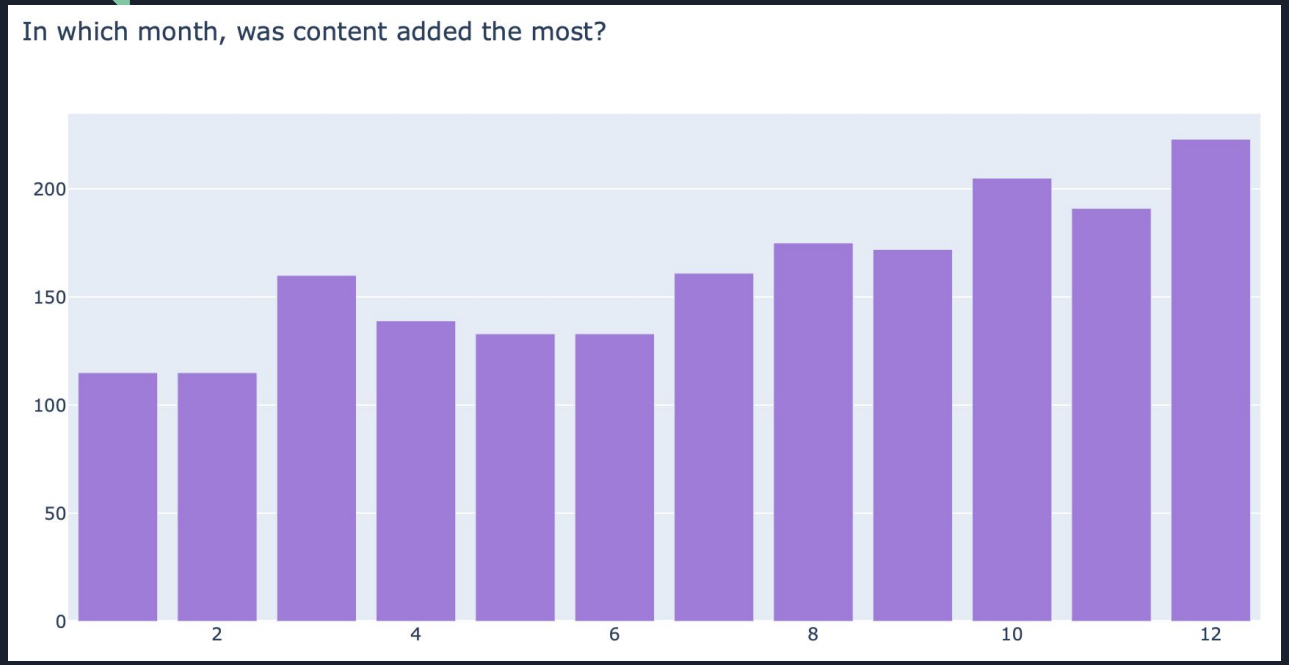
- TV-Y
- TV-Y7
- TV-Y7-FV
- TV-G
- TV-PG
- TV-14
- TV-MA

Countries with the Most Content

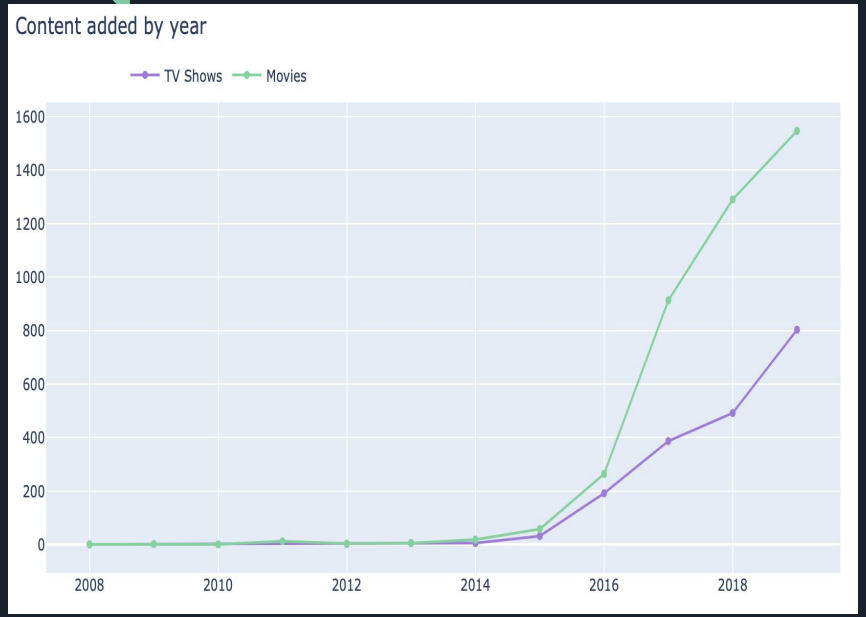


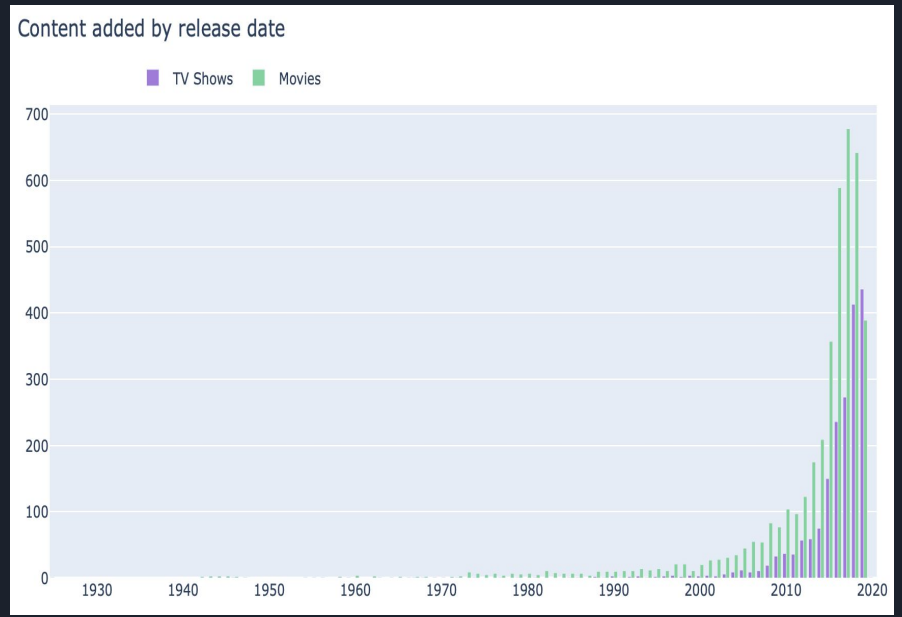


Content Added by Month

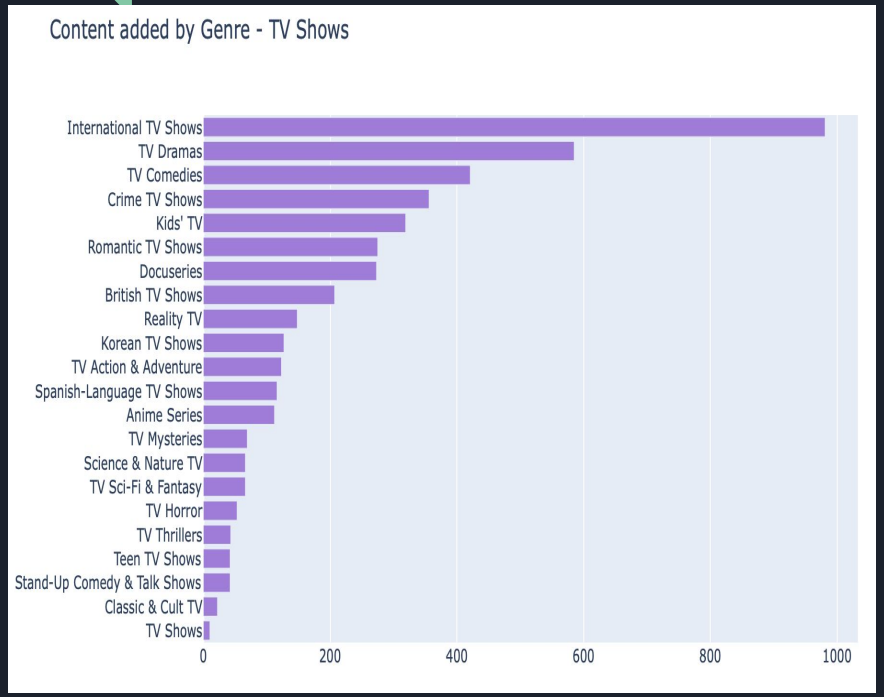


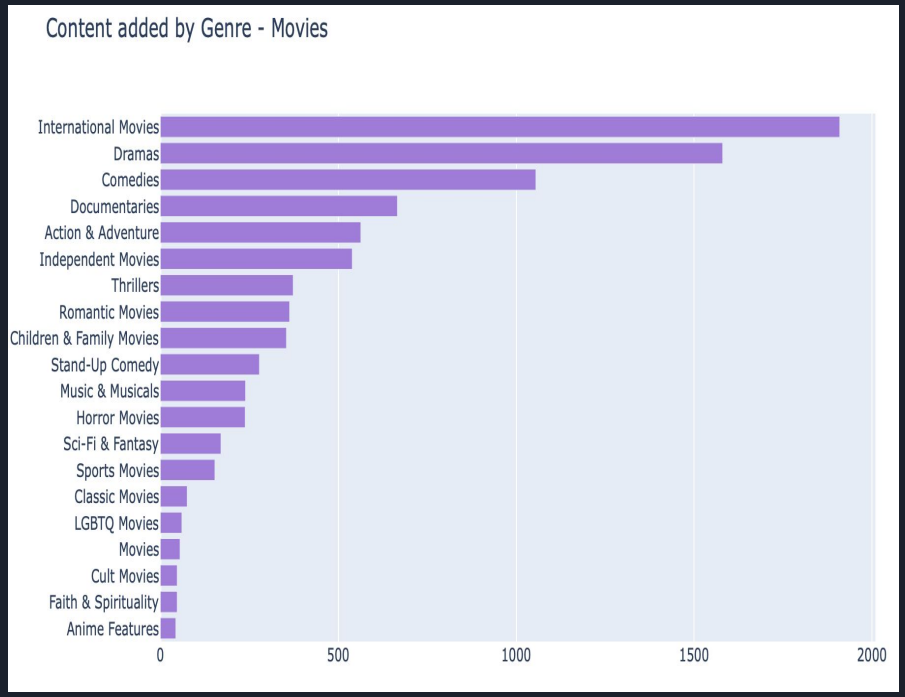
Content Added by Year/Release Date





Content Added by Genre

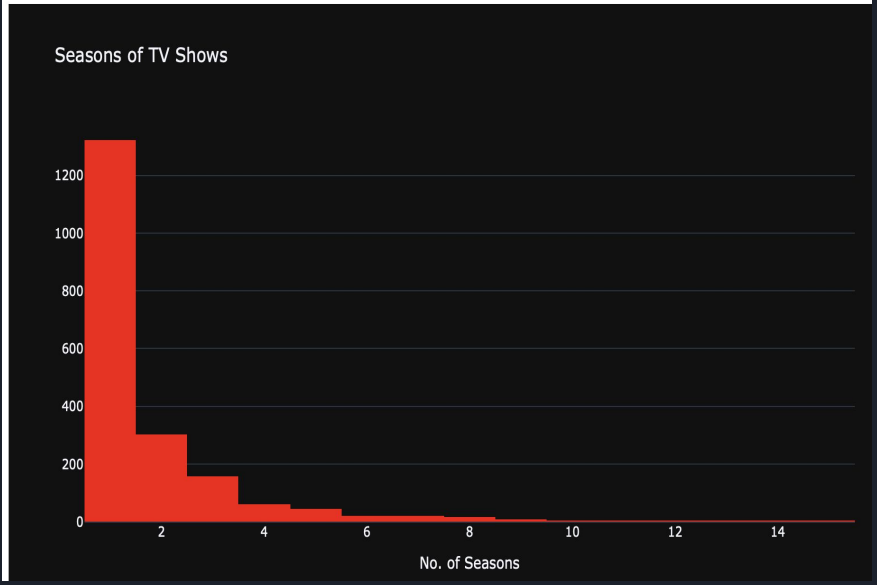




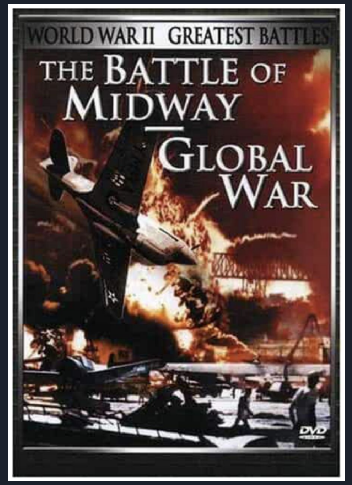
TV Shows with Highest No. of Seasons

GREY'S ANATOMY AND NCIS HAVE HIGHEST NO. OF SEASON ON NETFLIX

	title	director	cast	country	release_year
5908	Grey's Anatomy	NaN	Ellen Pompeo, Sandra Oh, Katherine Heigl, Justin Chambers, Patrick Dempsey, Chandra Wilson, T.R. Knight, James Pickens Jr., Isaiah Washington, Sara Ramirez, Eric Dane, Chyler Leigh, Kate Walsh, Kevin McKidd, Jessica Capshaw	United States	2018
5974	NCIS	NaN	Mark Harmon, Michael Weatherly, Pauley Perrette, David McCallum, Sean Murray, Cote de Pablo, Brian Dietzen, Rocky Carroll, Lauren Holly, Sasha Alexander, Joe Spano, Alan Dale, Emily Wickersham, Pancho Demmings	United States	2017



Oldest Movies on Netflix



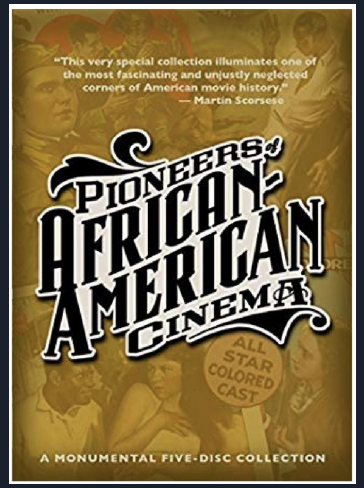
OLDEST MOVIES ON NETFLIX

	title	release_year	listed_in	country
2011	Prelude to War	1942	Classic Movies, Documentaries	United States
2013	The Battle of Midway	1942	Classic Movies, Documentaries	United States
2022	Undercover: How to Operate Behind Enemy Lines	1943	Classic Movies, Documentaries	United States
2023	Why We Fight: The Battle of Russia	1943	Documentaries	United States
2026	WWII: Report from the Aleutians	1943	Documentaries	United States
2017	The Memphis Belle: A Story of a\nFlying Fortress	1944	Classic Movies, Documentaries	United States
2019	The Negro Soldier	1944	Classic Movies, Documentaries	United States
2021	Tunisian Victory	1944	Classic Movies, Documentaries	United States, United Kingdom
2012	San Pietro	1945	Classic Movies, Documentaries	United States
2009	Nazi Concentration Camps	1945	Classic Movies, Documentaries	United States

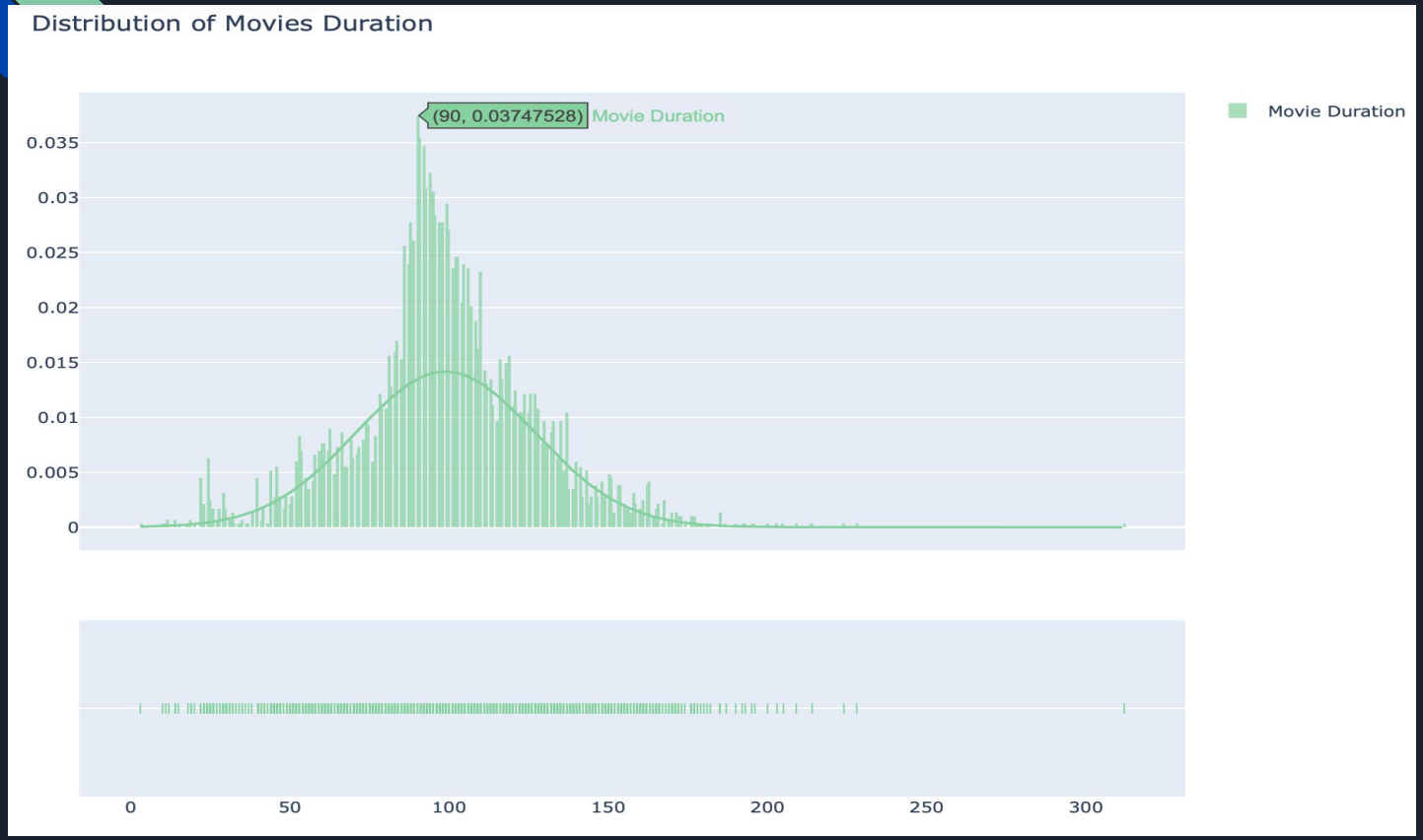
Oldest TV Shows on Netflix

OLDEST TV SHOW ON NETFLIX

	title	release_year	listed_in	country
4292	Pioneers: First Women Filmmakers*	1925	TV Shows	NaN
4079	Pioneers of African-American Cinema	1946	TV Shows	United States
5981	The Twilight Zone (Original Series)	1963	Classic & Cult TV, TV Sci-Fi & Fantasy	United States
5980	The Andy Griffith Show	1967	Classic & Cult TV, TV Comedies	United States
5704	Star Trek	1968	Classic & Cult TV, TV Action & Adventure, TV Sci-Fi & Fantasy	United States
614	Monty Python's Fliegender Zirkus	1972	International TV Shows, TV Comedies	United Kingdom, West Germany
5669	Monty Python's Flying Circus	1974	British TV Shows, Classic & Cult TV, International TV Shows	United Kingdom
5830	Dad's Army	1977	British TV Shows, Classic & Cult TV, TV Comedies	United Kingdom
4273	El Chavo	1979	Classic & Cult TV, Kids' TV, Spanish-Language TV Shows	Mexico
4369	Ninja Hattori	1981	Anime Series, Kids' TV	Japan

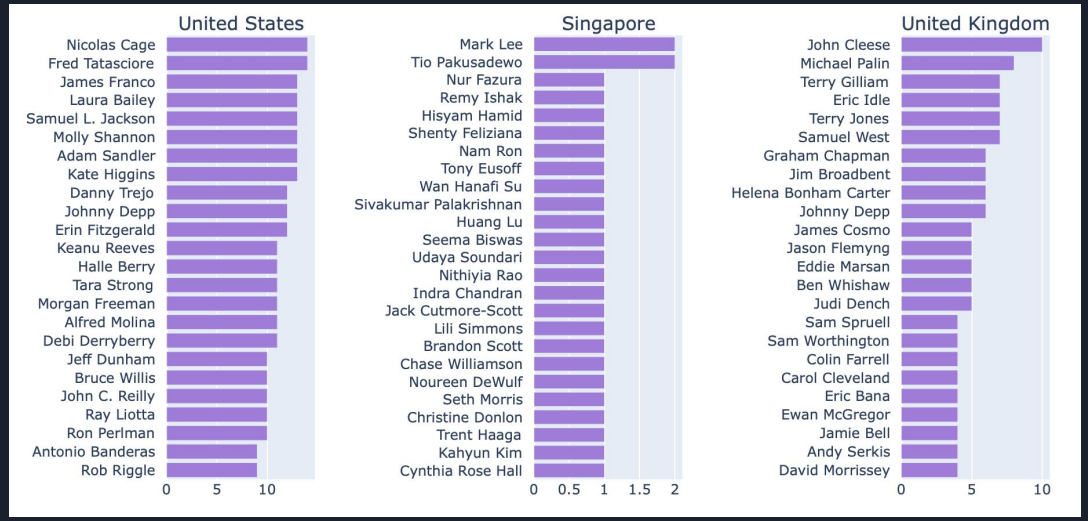


Distribution of Movies Duration



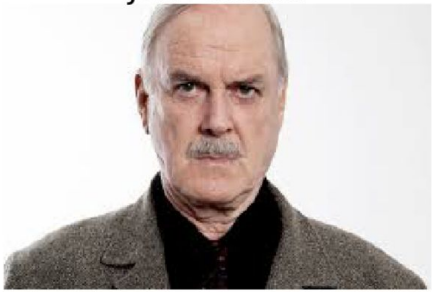
Most Popular Actors by Country - Movies

Nicolas Cage



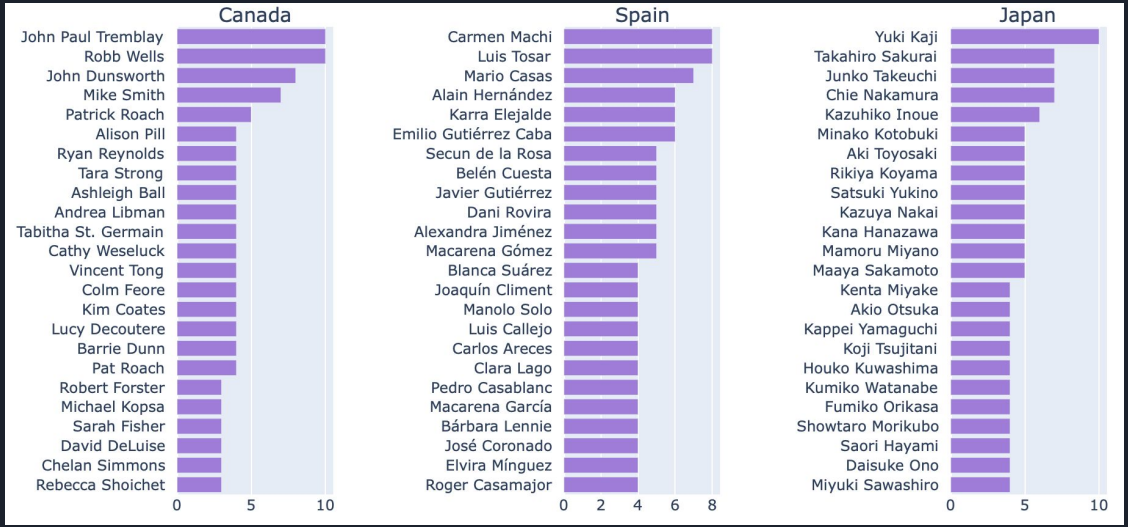
Mark Lee

John Cleese



Most Popular Actors by Country - Movies

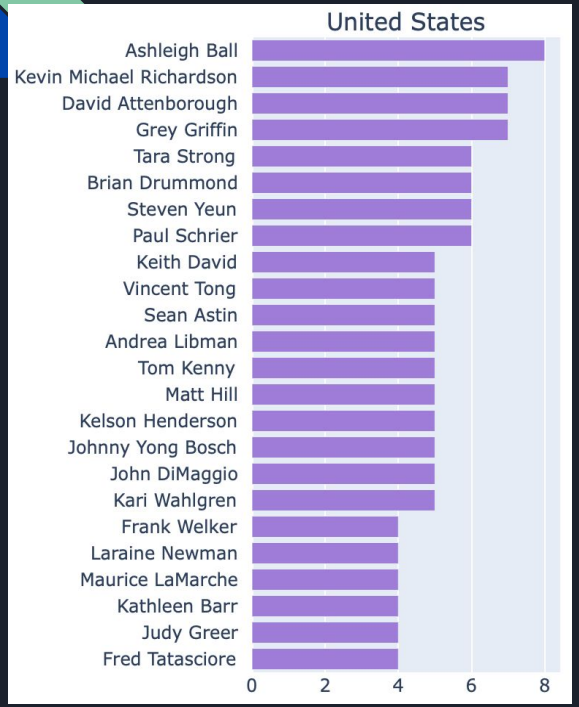
John Paul Tremblay

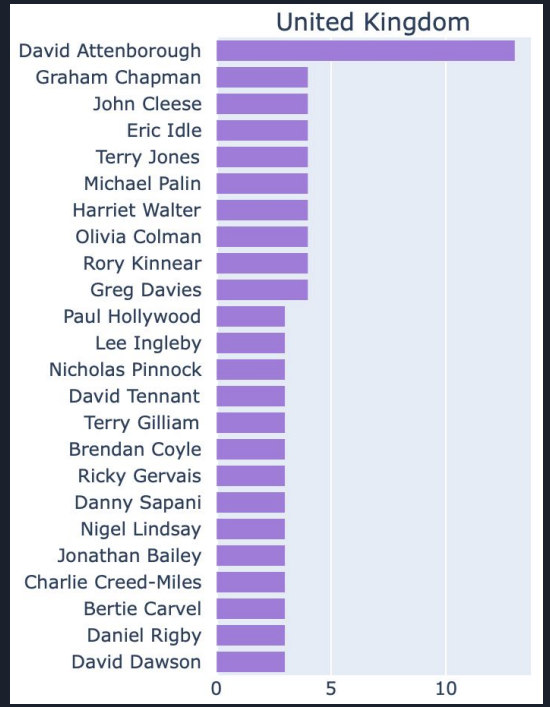


Carmen Machi

Yuki Kaji

Most Popular Actors by Country - TV Shows



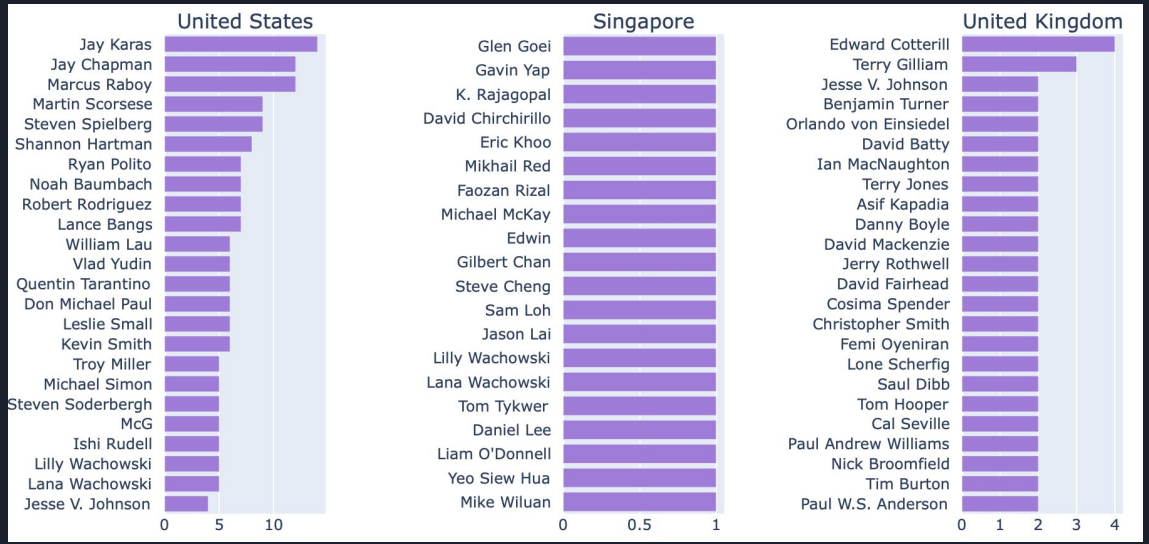


Ashleigh Ball

David Attenborough

Most Popular Directors by Country - Movies

Jay Karas

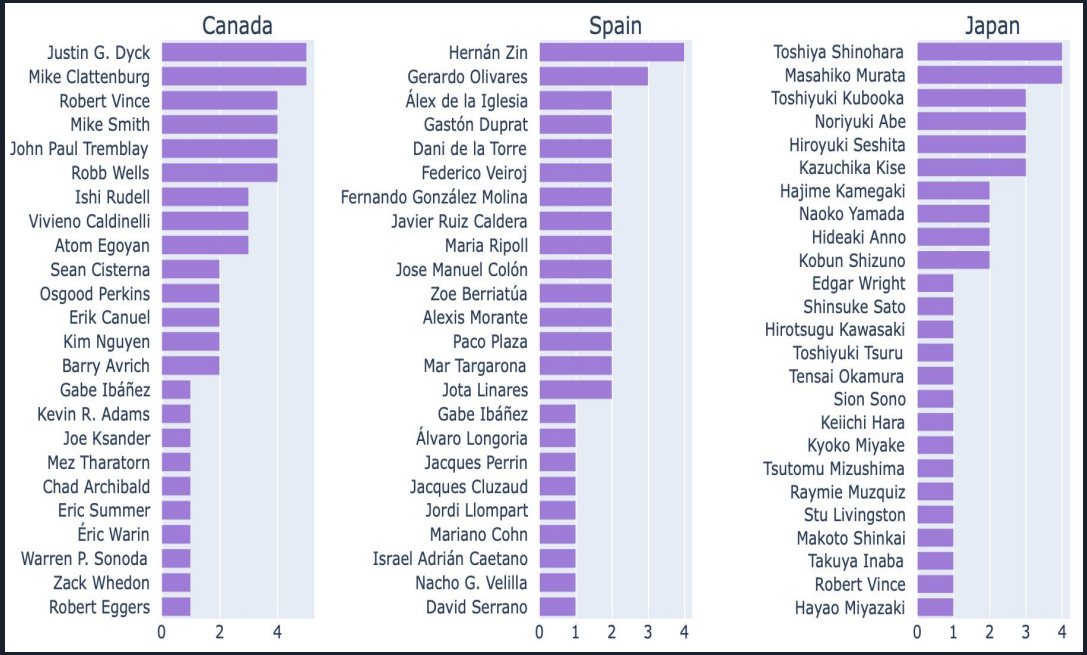


Glen Goei

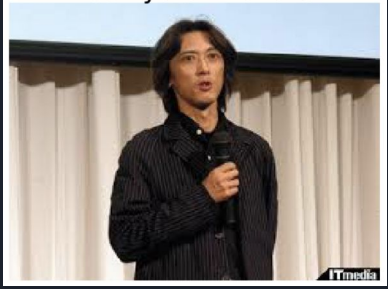
Edward Cotterill

Most Popular Directors by Country - Movies

Justin G. Dyck

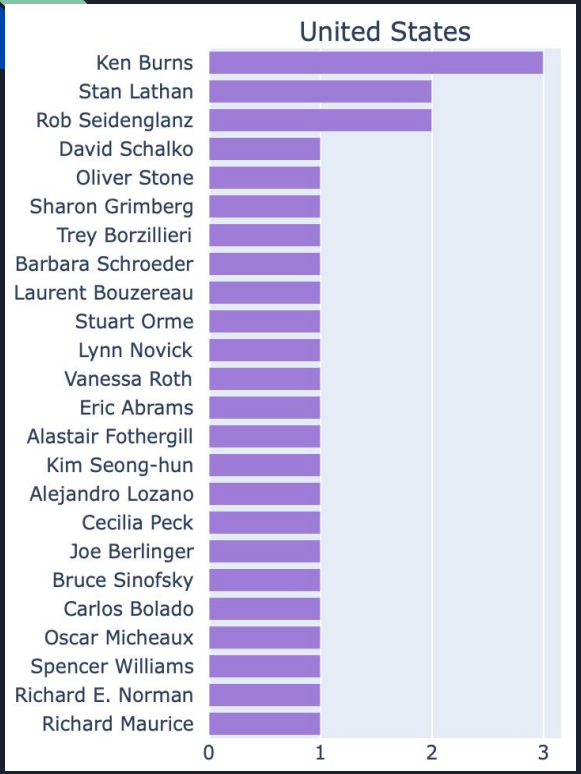


Toshiya Shinohara



Hernan Zin

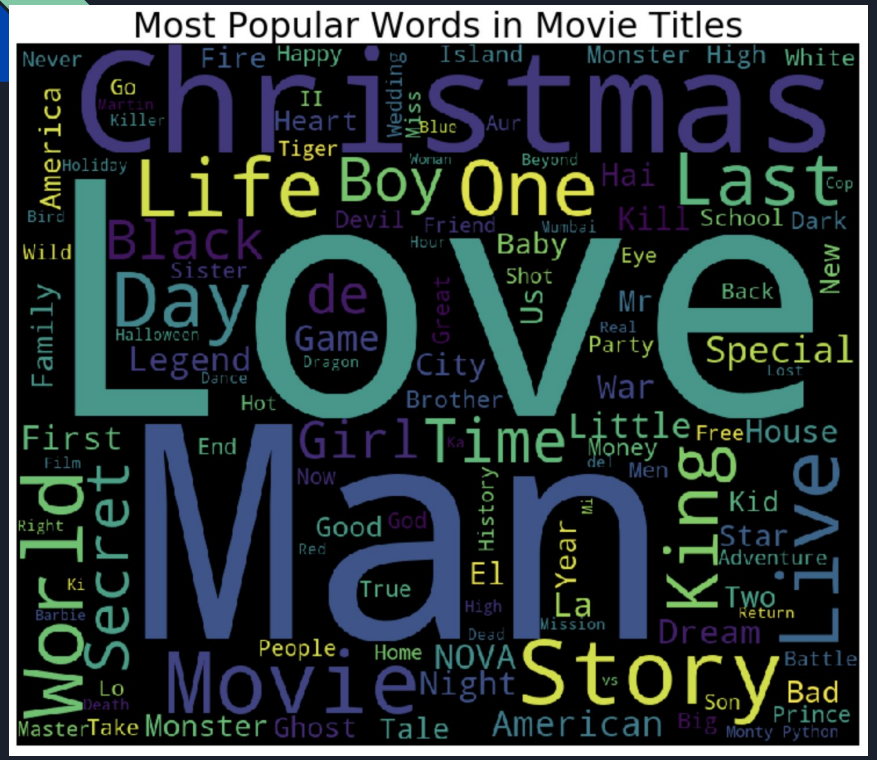
Most Popular Directors by Country - TV Shows

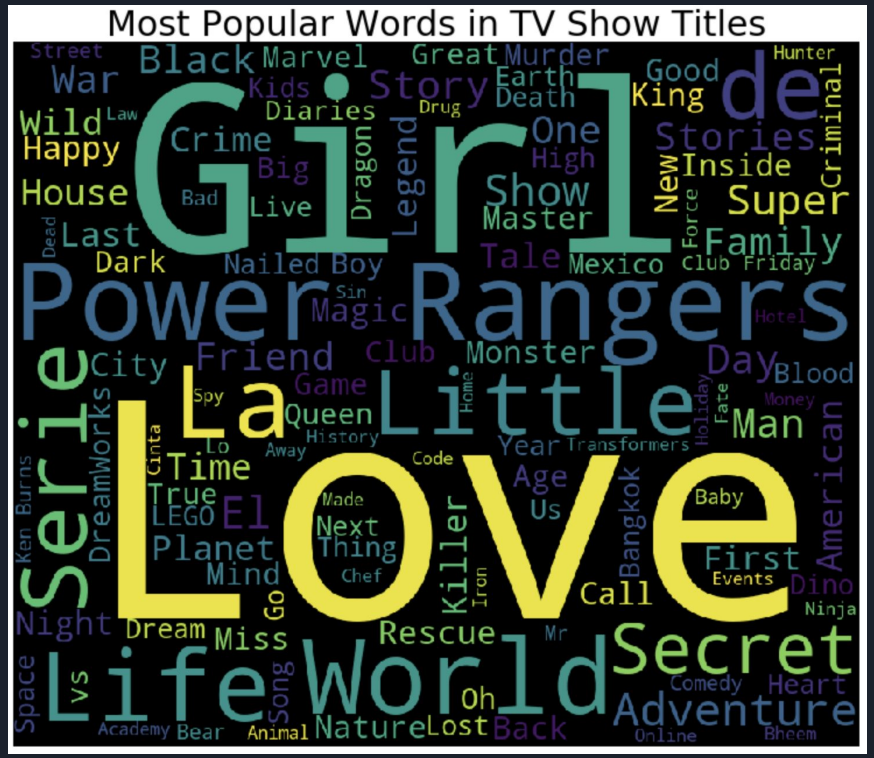


Ken Burns

Alastair Fothergill

Most Popular Words in Title





Data Cleaning

```
df.shape
```

```
(6234, 12)
```

The dataset provided has 6,234 rows and 12 columns.

```
df.isnull().sum()
```

```
show_id         0
type            0
title           0
director        1969
cast            570
country         476
date_added      11
release_year    0
rating          10
duration        0
listed_in       0
description     0
dtype: int64
```

The dataset consists of null values in multiple columns - 1,969 in *director* column, 570 in *cast* column, 476 in *country* column, 11 in _date*added* column and 10 in *rating* column. This could be due to the unavailability of the data on Flixable the third-party Netflix search engine.

Data Cleaning

Keywords Extraction Using Rake

RAKE short for *Rapid Automatic Keyword Extraction* algorithm, is a domain independent keyword extraction algorithm which tries to determine key phrases in a body of text by analyzing the frequency of word appearance and its co-occurance with other words in the text.

For this case, we used RAKE to split the text into a list of words and remove stopwords from that list. This returns us a list of what is known as *content words* which is useful for building our model and analyzing the keywords within the plot description of the show/ movie.

title	director	cast	listed_in	key_words
Norm of the North: King Sized Adventure	[richardfinn, timmaltby]	[alanmarriott, andrewtoth, briandobson, colehoward, jennifercameron, jonathanholmes, leetockar, lisadurupt, mayakay, michaeldobson]	[children & family movies, comedies]	[planning, evil, archaeologist, first, awesome, wedding, polar, bear, king, must, take, back, stolen, artifact, grandfather]
#realityhigh	[fernandolebrija]	[nestacooper, katewalsh, johnmichaelhiggins, keithpowers, aliciasanz, jakeborelli, kidink, youseferakat, rebekahgraf, annewinters, petergilroy, patrickdavis]	[comedies]	[cross, hairs, interest, longtime, crush, ex, nerdy, high, schooler, dani, finally, attracts, lands, social, media, celebrity]
Automata	[gabeibáñez]	[antoniobanderas, dylanmcdermott, melaniegriffith, birgittehjortsørensen, robertforster, christacampbell, timmcinnerny, andynyman, davidryall]	[international movies, sci-fi & fantasy, thrillers]	[dystopian, future, discovers, global, conspiracy, violating, protocol, tech, company, investigates, insurance, adjuster, robot, killed]
Fabrizio Copano: Solo pienso en mi	[rodrigotoro, franciscoschultz]	[fabriziocopano]	[stand-up comedy]	[stand, sperm, banks, set, family, whatsapp, groups, fabrizio, copano, takes, audience, participation, reflecting, next, level]
Good People	[henrikrubengenz]	[jamesfranco, katehudson, tomwilkinson, omarsy, samspruell, annafriel, thomasarnold, oliverdimsdale, dianahardcastle, michaeljibson, diarmaidmurtagh]	[action & adventure, thrillers]	[find, neighbor, apartment, luck, recently, murdered, struggling, couple, believe, stash, money]

RAKE algorithm in one slide

"For search managers, developers & data scientists finding ways to innovate"

~~For~~ **search managers**~~,~~ **developers** ~~&~~ **data scientists finding** ~~ways to~~ **innovate**

	algorithms	bounds	compatibility	components	constraints	constructing	corresponding	criteria	diophantine	equations
algorithms	2						1			
bounds		1								
compatibility			2							
components				1						
constraints					1					
constructing						1				
corresponding	1						1			
criteria								2		
diophantine									1	1
equations									1	1

	algorithms	bounds	compatibility	components	constraints	constructing	corresponding	criteria	diophantine	equations
deg(w)	3	2	2	1	2	1	2	2	3	3
freq(w)	2	1	2	1	1	1	1	2	1	1
deg(w) / freq(w)	1.5	2	1	1	2	1	2	1	3	3

Constructing criteria bounds	= 1 + 1 + 2	= **4**
Corresponding components	= 2 + 1	= **3**
Compatibility algorithms	= 1.5 + 1	= **2.5**

Wolters Kluwer

Content-Based Recommender System

- Bag of Words Model
- TF-IDF Vectorizer

$$\mathrm{Tf}(t) = \frac{\text{Frequency occurence of term t in document}}{\text{Total number of terms in document}}$$

$$\mathrm{Idf}(t) = log_{10}\left(\frac{\text{Total Number of documents}}{\text{Number of documents containing term t}}\right)$$

Content-Based Recommender System

title	bag_of_words
Norm of the North: King Sized Adventure	richardfinn timmaltby alanmarriott andrewtoth briandobson colehoward jennifercameron jonathanholmes leetockar lisadurupt mayakay michaeldobson children & family movies comedies grandfather planning polar bear king must take back stolen artifact awesome wedding evil archaeologist first
#realityhigh	fernandolebrija nestacooper katewalsh johnmichaelhiggins keithpowers aliciasanz jakeborelli kidink youseferakat rebekahgraf annewinters petergilroy patrickdavis comedies lands social media celebrity cross hairs interest longtime crush ex nerdy high schooler dani finally attracts
Automata	gabeibáñez antoniobanderas dylanmcdermott melaniegriffith birgittehjortsørensen robertforster christacampbell timmcinnerny andynyman davidryall international movies sci-fi & fantasy thrillers tech company investigates robot killed violating protocol global conspiracy insurance adjuster dystopian future discovers
Fabrizio Copano: Solo pienso en mi	rodrigotoro franciscoschultz fabriziocopano stand-up comedy family whatsapp groups next level stand fabrizio copano takes audience participation set reflecting sperm banks
Good People	henrikrubengenz jamesfranco katehudson tomwilkinson omarsy samspruell annafriel thomasarnold oliverdimsdale dianahardcastle michaeljibson diarmaidmurtagh action & adventure thrillers apartment recently murdered find believe struggling couple money stash luck neighbor

```python
# instantiating and generating the count matrix
count = TfidfVectorizer()
count_matrix = count.fit_transform(new_df['bag_of_words'])
```

```python
count_matrix
```

```
<3909x35149 sparse matrix of type '<class 'numpy.float64'>'
	with 106124 stored elements in Compressed Sparse Row format>
```

Content-Based Recommender System

- Cosine Similarity
- Euclidean Distance
- Pearson's Correlation

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Content-Based Recommender System

```python
# Function to get the most similar movies
def recommend_cosine(Title):
    id = inddict[Title]
    #Get the pairwise similarity scores of all movies compared to that movies
    #sorting them ang getting top 10
    similarity_scores = list(enumerate(cosine_sim[id]))
    similarity_scores = sorted(similarity_scores, key =lambda x : x[1], reverse = True)
    similarity_scores = similarity_scores[1:11]

    # Get the movies index
    movies_index = [i[0] for i in similarity_scores]

    #Return the top 10 most similar movies using iloc
    return list(new_df.iloc[movies_index].index)
```

```python
recommend_cosine('Rocky')
```

```
['Rocky III',
 'Rocky II',
 'Rocky IV',
 'Rocky V',
 'The Bleeder',
 'Tunisian Victory',
 'The Blue Planet: A Natural History of the Oceans',
 'Ghost Rider',
 'Wheelman',
 'Spy Kids 3: Game Over']
```

Description of Dataset

	Cust_Id	Rating	Movie_Id
1	1488844	3.0	1
4	30878	4.0	1
5	823519	3.0	1
6	893988	3.0	1
8	1248029	3.0	1

Movie_Id	Year	Name
1	2003.0	Dinosaur Planet
2	2004.0	Isle of Man TT 2004 Review
3	1997.0	Character
4	1994.0	Paula Abdul's Get Up & Dance
5	2004.0	The Rise and Fall of ECW
6	1997.0	Sick
7	1992.0	8 Man
8	2004.0	What the #$*! Do We Know!?
9	1991.0	Class of Nuke 'Em High 2
10	2001.0	Fighter

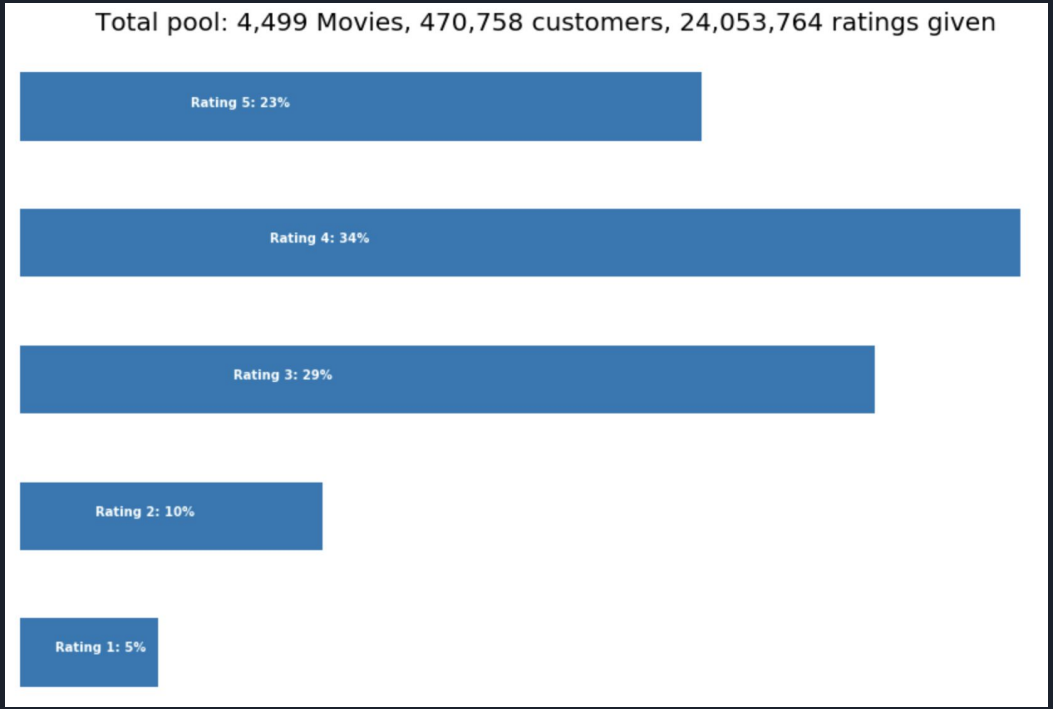
CustomerID,Rating,Date

- MovieIDs range from 1 to 17770 sequentially.
- CustomerIDs range from 1 to 2649429, with gaps. There are 480189 users.
- Ratings are on a five star (integral) scale from 1 to 5.
- Dates have the format YYYY-MM-DD.

MovieID,YearOfRelease,Title

- MovieID do not correspond to actual Netflix movie ids or IMDB movie ids.
- YearOfRelease can range from 1890 to 2005 and may correspond to the release of corresponding DVD, not necessarily its theaterical release.
- Title is the Netflix movie title and may not correspond to titles used on other sites. Titles are in English.

Description of Dataset



Data Cleaning

```
df.shape
```

```
(24058263, 2)
```

```
df.isnull().sum()
```

```
Cust_Id    0
Rating     4499
dtype: int64
```

- Remove Null values
- Remove all users with less than 100 movie ratings
- Pivot table to create a sparse matrix

Movie_Id	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Cust_Id																																
6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	0.0	5.0	0.0	0.0
79	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0
97	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
134	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	0.0	0.0
188	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0
199	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0
265	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	5.0	0.0	0.0
283	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	0.0	0.0
383	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	0.0	0.0

Algorithms Benchmarking

Algorithm	test_rmse	fit_time	test_time
KNNBaseline	1.216238	1.468207	0.319905
SVD	1.219008	0.371294	0.016853
BaselineOnly	1.223488	0.018836	0.031920
SVDpp	1.245075	0.623140	0.018906
KNNBasic	1.249050	1.317096	0.300885
KNNWithZScore	1.266237	1.492937	0.297541
KNNWithMeans	1.266423	1.408333	0.321885
CoClustering	1.271607	0.566804	0.012389
NMF	1.274602	0.632044	0.030305
SlopeOne	1.278264	0.051698	0.015374
NormalPredictor	1.693241	0.007591	0.020985

- Root Mean Squared Error (RMSE)
- Computational Efficiency

KNNBaseline

- 3-Fold Cross Validation

test_rmse	1.205	1.226	1.218
fit_time	1.340	1.312	1.300
test_time	0.302	0.333	0.297

- Final RMSE - **1.205**

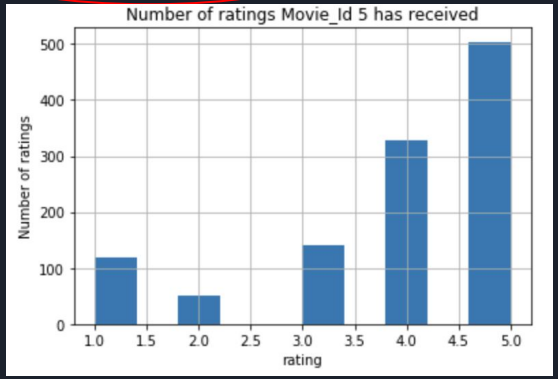
Best/ Worst Predictions

```
best_predictions
```

	uid	iid	rui	est	details	lu	Ui	err
2316	2118461	13	5.0	5.000000	{'actual_k': 3, 'was_impossible': False}	7	41	0.000000
1097	2439493	2	1.0	1.000000	{'actual_k': 17, 'was_impossible': False}	11	46	0.000000
2099	124042	7	2.0	1.994936	{'actual_k': 5, 'was_impossible': False}	1	42	0.005064
614	2564111	8	3.0	3.012996	{'actual_k': 40, 'was_impossible': False}	1	4408	0.012996
1630	404378	8	3.0	3.012996	{'actual_k': 40, 'was_impossible': False}	1	4408	0.012996
844	1052633	10	3.0	3.015038	{'was_impossible': False}	0	96	0.015038
823	651142	10	3.0	3.015038	{'was_impossible': False}	0	96	0.015038
1922	140565	10	3.0	3.015038	{'was_impossible': False}	0	96	0.015038
151	1652323	10	3.0	3.015038	{'was_impossible': False}	0	96	0.015038
96	1072863	10	3.0	3.015038	{'was_impossible': False}	0	96	0.015038

```
worst_predictions
```

	uid	iid	rui	est	details	lu	Ui	err
1472	2100070	5	1.0	3.691271	{'was_impossible': False}	0	391	2.691271
1110	229971	5	1.0	3.691271	{'was_impossible': False}	0	391	2.691271
1912	2520933	5	1.0	3.691271	{'was_impossible': False}	0	391	2.691271
2470	1127519	8	1.0	3.717969	{'actual_k': 6, 'was_impossible': False}	1	4408	2.717969
1392	2291306	6	5.0	2.117204	{'actual_k': 37, 'was_impossible': False}	4	419	2.882796
419	1190391	3	5.0	2.087615	{'actual_k': 18, 'was_impossible': False}	1	821	2.912385
1027	883478	3	1.0	3.953175	{'actual_k': 40, 'was_impossible': False}	2	821	2.953175
1504	2174682	10	5.0	1.955160	{'actual_k': 10, 'was_impossible': False}	1	96	3.044840
2003	491531	5	1.0	4.063403	{'actual_k': 15, 'was_impossible': False}	2	391	3.063403
604	407560	5	5.0	1.290189	{'actual_k': 1, 'was_impossible': False}	1	391	3.709811



Collaborative Filtering Model Recommendations

```
recommend("What the #$*! Do We Know!?", 0)
```

```
For movie (What the #$*! Do We Know!?)
- Top 10 movies recommended based on Pearsons'R correlation -
 PearsonR                                Name  count      mean
 0.038149  Sick                                131.0  3.100740
 0.035427  Character                           279.0  3.553407
 0.016064  Full Frame: Documentary Shorts       20.0  2.776332
 0.011349  Screamers                           120.0  3.046643
 0.011296  My Favorite Brunette                119.0  3.363058
 0.010994  Dinosaur Planet                     68.0  3.416132
 0.010303  Class of Nuke 'Em High 2            19.0  2.516688
 0.010007  Isle of Man TT 2004 Review          10.0  2.386080
 0.008172  Fighter                             27.0  2.885987
 0.006889  Neil Diamond: Greatest Hits Live    31.0  3.009788
```

Conclusion

1. Manage to build two recommendation models - Content-Based & Collaborative Filtering

2. Content-Based Model is limited to recommending items that are of the same type and could be improved to take into account users' behaviour as well

3. Collaborative Filtering Model suffers from a 'cold-start' problem

4. One way to tackle the problem is to build a popularity-based recommendations before building up enough user history to implement the model.