```
import pandas as pd

movies = pd.read_csv("/content/movies.csv")

```

```
movies.head()
```

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

```
import re

def clean_title(title):
    title = re.sub("[^a-zA-Z0-9 ]", "", title)
    return title
```

```
movies["clean_title"] = movies["title"].apply(clean_title)
```

```
movies
```

	movieId	title	genres	clean_title
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	Toy Story 1995
1	2	Jumanji (1995)	Adventure|Children|Fantasy	Jumanji 1995
2	3	Grumpier Old Men (1995)	Comedy|Romance	Grumpier Old Men 1995
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance	Waiting to Exhale 1995
4	5	Father of the Bride Part II (1995)	Comedy	Father of the Bride Part II 1995
...	...	...	...	...
62418	209157	We (2018)	Drama	We 2018
62419	209159	Window of the Soul (2001)	Documentary	Window of the Soul 2001
62420	209163	Bad Poems (2018)	Comedy|Drama	Bad Poems 2018
62421	209169	A Girl Thing (2001)	(no genres listed)	A Girl Thing 2001
62422	209171	Women of Devil's Island (1962)	Action|Adventure|Drama	Women of Devils Island 1962

62423 rows × 4 columns

```
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(ngram_range=(1,2))

tfidf = vectorizer.fit_transform(movies["clean_title"])
```

```
from sklearn.metrics.pairwise import cosine_similarity
import numpy as np

def search(title):
    title = clean_title(title)
    query_vec = vectorizer.transform([title])
    similarity = cosine_similarity(query_vec, tfidf).flatten()
    indices = np.argpartition(similarity, -5)[-5:]
    results = movies.iloc[indices].iloc[::-1]

    return results
```

```
import ipywidgets as widgets
from IPython.display import display

movie_input = widgets.Text(
    value='Toy Story',
    description='Movie Title:',
    disabled=False
)
```

```
movie_list = widgets.Output()

def on_type(data):
    with movie_list:
        movie_list.clear_output()
        title = data["new"]
        if len(title) > 5:
            display(search(title))

movie_input.observe(on_type, names='value')


display(movie_input, movie_list)
```

Movie Title: Toy Story

```
movie_id = 1

movie = movies[movies["movieId"] == movie_id]
```

```
ratings = pd.read_csv("/content/ratings.csv")
```

```
ratings.dtypes
```

	0
userId	int64
movieId	int64
rating	float64
timestamp	int64

dtype: object

```
similar_users = ratings[(ratings["movieId"] == movie_id) & (ratings["rating"] > 4)]["userId"].unique()
```

```
similar_user_recs = ratings[(ratings["userId"].isin(similar_users)) & (ratings["rating"] > 4)]["movieId"]
```

```
similar_user_recs = similar_user_recs.value_counts() / len(similar_users)

similar_user_recs = similar_user_recs[similar_user_recs > .10]
```

```
all_users = ratings[(ratings["movieId"].isin(similar_user_recs.index)) & (ratings["rating"] > 4)]
```

```
all_user_recs = all_users["movieId"].value_counts() / len(all_users["userId"].unique())
```

```
rec_percentages = pd.concat([similar_user_recs, all_user_recs], axis=1)
rec_percentages.columns = ["similar", "all"]
```

```
rec_percentages
```

movieId	similar	all
1	1.000000	0.124861
318	0.444989	0.342147
260	0.400146	0.221404
356	0.369742	0.235092
296	0.367358	0.285047
...	...	...
59315	0.103862	0.053784
778	0.103266	0.074634
953	0.103067	0.046076
551	0.101146	0.041262
1225	0.100351	0.057175

111 rows × 2 columns

```
rec_percentages["score"] = rec_percentages["similar"] / rec_percentages["all"]
```

```
rec_percentages = rec_percentages.sort_values("score", ascending=False)
```

```
rec_percentages.head(10).merge(movies, left_index=True, right_on="movieId")
```

	similar	all	score	movieId	title	genres	clean_title
0	1.000000	0.124861	8.008876	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy	Toy Story 1995
3021	0.281115	0.053825	5.222752	3114	Toy Story 2 (1999)	Adventure|Animation|Children|Comedy|Fantasy	Toy Story 2 1999
2264	0.111413	0.025316	4.400826	2355	Bug's Life, A (1998)	Adventure|Animation|Children|Comedy	Bugs Life A 1998
14813	0.153011	0.035150	4.353060	78499	Toy Story 3 (2010)	Adventure|Animation|Children|Comedy|Fantasy|IMAX	Toy Story 3 2010
4780	0.235212	0.070532	3.334840	4886	Monsters, Inc. (2001)	Adventure|Animation|Children|Comedy|Fantasy	Monsters Inc 2001
6258	0.227065	0.071665	3.168428	6377	Finding Nemo (2003)	Adventure|Animation|Children|Comedy	Finding Nemo 2003
580	0.214016	0.067554	3.168056	588	Aladdin (1992)	Adventure|Animation|Children|Comedy|Musical	Aladdin 1992
					Beauty and the		Beauty and the

```
def find_similar_movies(movie_id):
    similar_users = ratings[(ratings["movieId"] == movie_id) & (ratings["rating"] > 4)]["userId"].unique()
    similar_user_recs = ratings[(ratings["userId"].isin(similar_users)) & (ratings["rating"] > 4)]["movieId"]
    similar_user_recs = similar_user_recs.value_counts() / len(similar_users)

    similar_user_recs = similar_user_recs[similar_user_recs > .10]
    all_users = ratings[(ratings["movieId"].isin(similar_user_recs.index)) & (ratings["rating"] > 4)]
    all_user_recs = all_users["movieId"].value_counts() / len(all_users["userId"].unique())
    rec_percentages = pd.concat([similar_user_recs, all_user_recs], axis=1)
    rec_percentages.columns = ["similar", "all"]

    rec_percentages["score"] = rec_percentages["similar"] / rec_percentages["all"]
    rec_percentages = rec_percentages.sort_values("score", ascending=False)
    return rec_percentages.head(10).merge(movies, left_index=True, right_on="movieId")[["score", "title", "genres"]]
```

```
import ipywidgets as widgets
from IPython.display import display

movie_name_input = widgets.Text(
    value='Toy Story',
    description='Movie Title:',
    disabled=False
)
recommendation_list = widgets.Output()

def on_type(data):
    with recommendation_list:
        recommendation_list.clear_output()
        title = data["new"]
```

```
        if len(title) > 5:
            results = search(title)
            movie_id = results.iloc[0]["movieId"]
            display(find_similar_movies(movie_id))

movie_name_input.observe(on_type, names='value')

display(movie_name_input, recommendation_list)
```

Movie Title: Toy Story

	score	title	genres
3021	18.791795	Toy Story 2 (1999)	Adventure|Animation|Children|Comedy|Fantasy
2264	8.238676	Bug's Life, A (1998)	Adventure|Animation|Children|Comedy
2669	6.987077	Iron Giant, The (1999)	Adventure|Animation|Children|Drama|Sci-Fi
14813	6.508593	Toy Story 3 (2010)	Adventure|Animation|Children|Comedy|Fantasy|IMAX
3650	6.345845	Chicken Run (2000)	Animation|Children|Comedy
1992	5.625199	Little Mermaid, The (1989)	Animation|Children|Comedy|Musical|Romance
2895	5.446722	Who Framed Roger Rabbit? (1988)	Adventure|Animation|Children|Comedy|Crime|Fant...
3082	5.361188	Galaxy Quest (1999)	Adventure|Comedy|Sci-Fi
1818	5.353431	Mulan (1998)	Adventure|Animation|Children|Comedy|Drama|Musi...