# Project Goals

- examine various recomendation systems with and without user ratings.

# Imports & Getting the data

In [594]:
```
from random import gauss as gs, uniform as uni, seed
import numpy as np
import pandas as pd
from datetime import datetime

from matplotlib import pyplot as plt
import seaborn as sns
import plotly.express as px

import matplotlib.pyplot as plt
```

In [595]:
```
movies = pd.read_csv('Data/movies.csv')
ratings = pd.read_csv('Data/ratings.csv')
tags = pd.read_csv('Data/tags.csv')
```

## Examining & Cleaning the Data!

The movie and rating CSVs are fairly self-explainatory (providing movies, reviews, and users), so we'll start there.

In [596]:
```
def get_movie_name(movie_id):
    try:
        return movies.loc[movie_id, 'title']
    except KeyError:
        return "Movie not found"

def get_movie_genres(movie_id):
    try:
        return movies.loc[movie_id, 'genres']
    except KeyError:
        return "Movie not found"

def get_movie_year(movie_id):
    try:
        return movies.loc[movie_id, 'year']
    except KeyError:
        return "Movie not found"

def get_movie_rating(movie_id):
    try:
        rating = movies.loc[movie_id, 'avg_rating']
        return f'{round(rating,2)}'
    except KeyError:
        return "Movie rating not found"

def get_movie_num_reviews(movie_id):
    try:
        num = movies.loc[movie_id, 'num_reviews']
        return f'{int(num)} users'
    except KeyError:
        return " "

def movie_pretty_string(movie_id):
    try:
        string = f'{get_movie_name(movie_id)}, {get_movie_year(movi
e_id)}   | {get_movie_rating(movie_id)} by {get_movie_num_reviews(m
ovie_id)} users |  {get_movie_genres(movie_id)}'
        return string
    except KeyError:
        return "Movie not found"
```

## Movies

In [597]:
```
#Problems: Year is tied into the title, no reviews, genres are a pi
pe separated list, ids are present, but not the indexes
movies.set_index('movieId', inplace=True)
movies.sample(5)
```

Out[597]:

| movieId | title | genres |
|---|---|---|
| 3991 | 102 Dalmatians (2000) | Children\|Comedy |
| 48516 | Departed, The (2006) | Crime\|Drama\|Thriller |
| 142196 | Cornered! (2009) | Comedy\|Horror |
| 102033 | Pain & Gain (2013) | Action\|Comedy\|Crime |
| 115216 | Salvation, The (2014) | Drama\|Western |

In [598]:
```
#Get the year from title
movies['year'] = movies.title.str.extract("\((\d{4})\)", expand=Tru
e).astype(str)
movies['title'] = movies.title.str[:-7]
```

In [599]:
```
#Add the mean rating for each movie
ratings_movie_mean = ratings.groupby('movieId').mean()
mean_rating = ratings.groupby('movieId').mean()['rating']
movies['avg_rating'] = mean_rating
movies['num_reviews'] = ratings.groupby('movieId').count()['ratin
g']
```

In [600]:
```
fig = px.bar(movies, x='year', y='avg_rating', hover_data=['title',
'num_reviews'])

fig.update_layout(
    title_text='Movies', # title of plot
    xaxis_title_text='Year', # xaxis label
    yaxis_title_text='Number of Movies Released', # yaxis label
    bargap=0.2, # gap between bars of adjacent location coordinates
    bargroupgap=0.1, # gap between bars of the same location coordi
nates
)

fig.show()
```

In [601]:
```
df_temp = movies
df_temp['decade'] = df_temp['year'].str[:3]+'0s'
movies['decade'] = df_temp['year'].str[:3]+'0s'
df_temp = df_temp.groupby(['decade'])['title'].count()
df_temp

fig = px.bar(df_temp, text_auto=True)

fig.update_layout(
    title_text='Movies released per decade', # title of plot
    xaxis_title_text='Decade', # xaxis label
    yaxis_title_text='Number of Movies Released', # yaxis label
    bargap=0.2, # gap between bars of adjacent location coordinates
    bargroupgap=0.1, # gap between bars of the same location coordi
nates
)

fig.show()
```

In [602]:
```
#make the genres a csv
def unpack_genres(string):
    s = string.split('|')
    return s

movies['genres'] = movies['genres'].map(unpack_genres)
```

In [603]:
```
unique_genres = [ ]
for row in movies['genres']:
    for genre in row:
        if genre not in unique_genres:
            unique_genres.append(genre)

unique_genres
```

Out[603]:
```
['Adventure',
 'Animation',
 'Children',
 'Comedy',
 'Fantasy',
 'Romance',
 'Drama',
 'Action',
 'Crime',
 'Thriller',
 'Horror',
 'Mystery',
 'Sci-Fi',
 'War',
 'Musical',
 'Documentary',
 'IMAX',
 'Western',
 'Film-Noir',
 '(no genres listed)']
```

## Creating a sparse matrix of Genres

In [604]:
```
movie_genre_matrix = pd.DataFrame()
movie_genre_matrix['movieId'] = movies.index

for genre in unique_genres:
    movie_genre_matrix[genre] = movies['genres'].apply(lambda x: 1
if genre in x else 0)

for col in movie_genre_matrix.columns:
    summ = movie_genre_matrix[col].sum()
    print(f'{col} has {summ} movies')
```

```
movieId has 411115845 movies
Adventure has 627.0 movies
Animation has 172.0 movies
Children has 361.0 movies
Comedy has 2047.0 movies
Fantasy has 336.0 movies
Romance has 995.0 movies
Drama has 2556.0 movies
Action has 873.0 movies
Crime has 656.0 movies
Thriller has 995.0 movies
Horror has 513.0 movies
Mystery has 310.0 movies
Sci-Fi has 472.0 movies
War has 231.0 movies
Musical has 229.0 movies
Documentary has 167.0 movies
IMAX has 19.0 movies
Western has 110.0 movies
Film-Noir has 68.0 movies
(no genres listed) has 0.0 movies
```

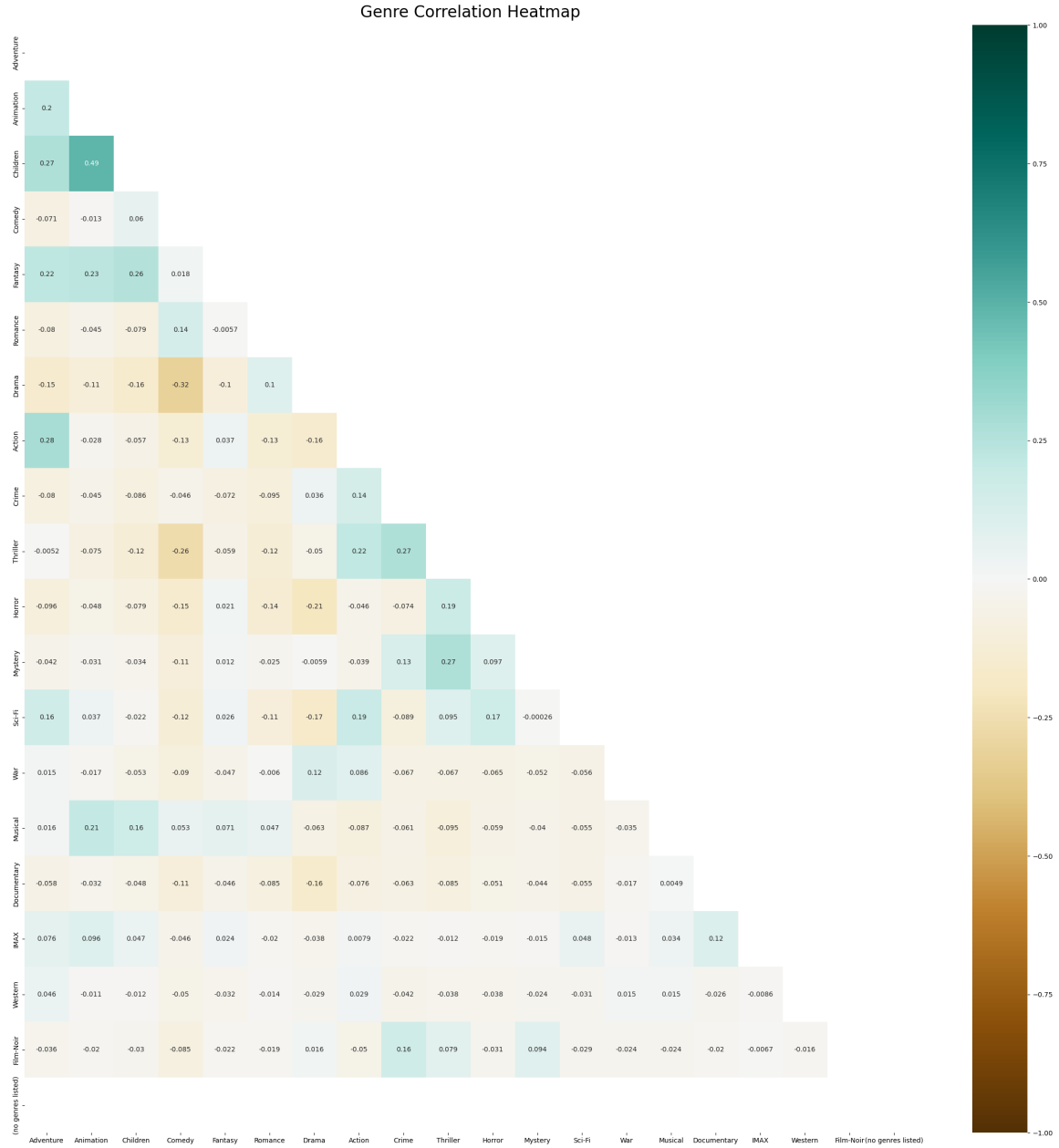
In [605]:
```
df_matrix = movie_genre_matrix.drop(columns='movieId')

corr_matrix = df_matrix.corr()

plt.figure(figsize=(30, 30))

#create a mask to remove the duplicate upper half
mask = np.triu(np.ones_like(corr_matrix, dtype=np.bool))

heatmap = sns.heatmap(corr_matrix, vmin=-1, mask=mask, vmax=1, annot=True, cmap='BrBG')
heatmap.set_title('Genre Correlation Heatmap', fontdict={'fontsize':24}, pad=12);
```



In [606]:
```
#looks good!
movies
```

Out[606]:

| movieId | title | genres | year | avg_rating | num_reviews | decade |
|---|---|---|---|---|---|---|
| **1** | Toy Story | [Adventure, Animation, Children, Comedy, Fantasy] | 1995 | 3.920930 | 215.0 | 1990s |
| **2** | Jumanji | [Adventure, Children, Fantasy] | 1995 | 3.431818 | 110.0 | 1990s |
| **3** | Grumpier Old Men | [Comedy, Romance] | 1995 | 3.259615 | 52.0 | 1990s |
| **4** | Waiting to Exhale | [Comedy, Drama, Romance] | 1995 | 2.357143 | 7.0 | 1990s |
| **5** | Father of the Bride Part II | [Comedy] | 1995 | 3.071429 | 49.0 | 1990s |
| **...** | ... | ... | ... | ... | ... | ... |
| **193581** | Black Butler: Book of the Atlantic | [Action, Animation, Comedy, Fantasy] | 2017 | 4.000000 | 1.0 | 2010s |
| **193583** | No Game No Life: Zero | [Animation, Comedy, Fantasy] | 2017 | 3.500000 | 1.0 | 2010s |
| **193585** | Flint | [Drama] | 2017 | 3.500000 | 1.0 | 2010s |
| **193587** | Bungo Stray Dogs: Dead Apple | [Action, Animation] | 2018 | 3.500000 | 1.0 | 2010s |
| **193609** | Andrew Dice Clay: Dice Rules | [Comedy] | 1991 | 4.000000 | 1.0 | 1990s |

9742 rows × 6 columns

## Ratings

In [607]:
```
#problem: timestamp not human readable, nice to have real names as well
ratings.sample(5)
```

Out[607]:

| | userId | movieId | rating | timestamp |
|---|---|---|---|---|
| **47719** | 308 | 161 | 1.0 | 1421374403 |
| **24619** | 170 | 376 | 3.0 | 840473290 |
| **50341** | 325 | 2302 | 3.0 | 1039395597 |
| **30876** | 216 | 4002 | 3.0 | 975212110 |
| **42744** | 288 | 3210 | 3.0 | 976120587 |

In [608]:
```
#times now datetime
ratings['timestamp'] = ratings['timestamp'].map(datetime.fromtimestamp)
```

In [609]:
```
ratings['title'] = ratings['movieId'].map(get_movie_name)
```

In [610]:
```
ratings.describe()
```

Out[610]:

| | userId | movieId | rating | timestamp |
|---|---|---|---|---|
| **count** | 100836.000000 | 100836.000000 | 100836.000000 | 100836 |
| **mean** | 326.127564 | 19435.295718 | 3.501557 | 2008-03-19 12:38:29.931839488 |
| **min** | 1.000000 | 1.000000 | 0.500000 | 1996-03-29 13:36:55 |
| **25%** | 177.000000 | 1199.000000 | 3.000000 | 2002-04-18 05:57:46 |
| **50%** | 325.000000 | 2991.000000 | 3.500000 | 2007-08-02 16:31:02 |
| **75%** | 477.000000 | 8122.000000 | 4.000000 | 2015-07-04 03:15:44.500000 |
| **max** | 610.000000 | 193609.000000 | 5.000000 | 2018-09-24 10:27:30 |
| **std** | 182.618491 | 35530.987199 | 1.042529 | NaN |

# Visualizing the Data

In [611]:
```
temp_year = movies.sort_values(by=['year'])
fig = px.scatter(temp_year, x='year', y='avg_rating', hover_data=
['title', 'num_reviews'])

fig.update_layout(
    title_text='Movies ratings by Year', # title of plot
    xaxis_title_text='Year', # xaxis label
    yaxis_title_text='Rating', # yaxis label
    bargap=0.2, # gap between bars of adjacent location coordinates
    bargroupgap=0.1, # gap between bars of the same location coordi
nates
)

fig.show()
```

In [612]:
```
df_temp = movies.groupby('decade').mean(numeric_only=True)

fig = px.bar(df_temp, y='avg_rating', text_auto=True)

fig.update_layout(
    title_text='Average Reviews of movies per decade', # title of p
lot
    xaxis_title_text='Decade', # xaxis label
    yaxis_title_text='Average Rating', # yaxis label
    bargap=0.2, # gap between bars of adjacent location coordinates
    bargroupgap=0.1, # gap between bars of the same location coordi
nates
)

fig.show()
```

In [613]:
```
fig = px.histogram(ratings, x='rating',
                           text_auto=True,
                           labels={'x':'rating', 'y':'count'},
                           color='rating',
                           color_discrete_sequence=["SandyBrown",
"Gold", 'Coral', 'Salmon', "OrangeRed", 'SandyBrown', "Coral", 'Sal
mon', "red", 'OrangeRed'],
                           )

fig.update_layout(
    title_text='Number of Ratings', # title of plot
    xaxis_title_text='Rating', # xaxis label
    yaxis_title_text='Count', # yaxis label
    bargap=0.2, # gap between bars of adjacent location coordinates
    bargroupgap=0.1, # gap between bars of the same location coordi
nates
)

fig.show()
```

In [614]:
```
temp_df = ratings.groupby(['title', 'movieId']).count().sort_values
(by='rating',ascending=False).reset_index().iloc[0:10,:]

temp_df['average rating'] = temp_df['movieId'].map(get_movie_ratin
g)

fig = px.bar(temp_df, x = 'title', y = 'userId',text_auto=True, hov
er_data='average rating')

fig.update_layout(
    title_text='Most frequently rated movies', # title of plot
    xaxis_title_text='Movie', # xaxis label
    yaxis_title_text='Number of Ratings', # yaxis label
    bargap=0.2, # gap between bars of adjacent location coordinates
    bargroupgap=0.1, # gap between bars of the same location coordi
nates
)

fig.show()
```

In [615]:
```
#The average movie has 14 reviews
rated_movies = movies.loc[movies['num_reviews'] >= 14]

# Find Highest rated movies:
top_10 = rated_movies.sort_values(by=['avg_rating','num_reviews'],a
scending=False)[:10]


fig = px.bar(top_10, x = 'title', y = 'avg_rating',text_auto=True)
fig.update_layout(
    title_text='Higest rated movies with more than 14 reviews', # t
itle of plot
    xaxis_title_text='Movie', # xaxis label
    yaxis_title_text='Number of Ratings', # yaxis label
    bargap=0.2, # gap between bars of adjacent location coordinates
    bargroupgap=0.1, # gap between bars of the same location coordi
nates
)
fig.update_yaxes(range=[0, 5])
fig.show()

# Lowest rated movies
lowest_10 = rated_movies.sort_values(by=['avg_rating','num_review
s'],ascending=True)[:10]

fig = px.bar(lowest_10, x = 'title', y = 'avg_rating',text_auto=Tru
e)
fig.update_layout(
    title_text='Lowest rated movies with more than 14 reviews', # t
itle of plot
    xaxis_title_text='Movie', # xaxis label
    yaxis_title_text='Number of Ratings', # yaxis label
    bargap=0.2, # gap between bars of adjacent location coordinates
    bargroupgap=0.1, # gap between bars of the same location coordi
nates
)
fig.update_yaxes(range=[0, 5])

fig.show()
```

# Cosine Similarity for Movie Genres (Item Based Recomendation, No Users)

The cosine similarity is a metric to determine how similar different things are.

In [616]:
```
from sklearn.metrics.pairwise import cosine_similarity
```

In [617]:
```
#get a random movie
test_movie = np.random.choice(movies.shape[0])
get_movie_name(test_movie)
```

Out[617]:
```
"We Don't Live Here Anymore"
```

In [618]:
```
def top_k_items(item_id, top_k, corr_mat, map_name):

    # sort correlation value ascendingly and select top_k item_id
    top_items = corr_mat[item_id,:].argsort()[-top_k:]
    top_items = [map_name[e] for e in top_items]

    return top_items


# compute similarity matix
corr_mat = cosine_similarity(movie_genre_matrix)

# get top-k similar items
ind2name = {ind:name for ind,name in enumerate(movie_genre_matrix.i
ndex)}
name2ind = {v:k for k,v in ind2name.items()}
similar_items = top_k_items(name2ind[test_movie],
                            top_k = 3,
                            corr_mat = corr_mat,
                            map_name = ind2name)

# display result
print(f'The top-3 similar movie to: {get_movie_name(test_movie)}')
for movie in similar_items:
    print(f'Movie: {get_movie_name(movie)}')
    print(f'        Correlation: {corr_mat[movie]}')
    print(f'        Genres: {get_movie_genres(movie)}')
```

```
---------------------------------------------------------------------------
-------
ValueError                                Traceback (most recent cal
l last)
Cell In[618], line 11
      7     return top_items
     10 # compute similarity matix
---> 11 corr_mat = cosine_similarity(movie_genre_matrix)
     13 # get top-k similar items
     14 ind2name = {ind:name for ind,name in enumerate(movie_genre_m
atrix.index)}

File /opt/anaconda3/envs/learn-env/lib/python3.9/site-packages/sklea
rn/utils/_param_validation.py:213, in validate_params.<locals>.decor
ator.<locals>.wrapper(*args, **kwargs)
    207 try:
    208     with config_context(
    209         skip_parameter_validation=(
    210             prefer_skip_nested_validation or global_skip_val
idation
    211         )
    212     ):
--> 213         return func(*args, **kwargs)
    214 except InvalidParameterError as e:
    215     # When the function is just a wrapper around an estimato
r, we allow
    216     # the function to delegate validation to the estimator,
but we replace
    217     # the name of the estimator by the name of the function
in the error
    218     # message to avoid confusion.
    219     msg = re.sub(
    220         r"parameter of \w+ must be",
    221         f"parameter of {func.__qualname__} must be",
    222         str(e),
    223     )

File /opt/anaconda3/envs/learn-env/lib/python3.9/site-packages/sklea
rn/metrics/pairwise.py:1679, in cosine_similarity(X, Y, dense_outpu
t)
   1635 """Compute cosine similarity between samples in X and Y.
   1636
   1637 Cosine similarity, or the cosine kernel, computes similarity
as the
   (...)
   1675         [0.57..., 0.81...]])
   1676 """
   1677 # to avoid recursive import
-> 1679 X, Y = check_pairwise_arrays(X, Y)
   1681 X_normalized = normalize(X, copy=True)
   1682 if X is Y:

File /opt/anaconda3/envs/learn-env/lib/python3.9/site-packages/sklea
rn/metrics/pairwise.py:175, in check_pairwise_arrays(X, Y, precomput
ed, dtype, accept_sparse, force_all_finite, ensure_2d, copy)
    172     dtype = dtype_float
```

```
    174 if Y is X or Y is None:
--> 175     X = Y = check_array(
    176         X,
    177         accept_sparse=accept_sparse,
    178         dtype=dtype,
    179         copy=copy,
    180         force_all_finite=force_all_finite,
    181         estimator=estimator,
    182         ensure_2d=ensure_2d,
    183     )
    184 else:
    185     X = check_array(
    186         X,
    187         accept_sparse=accept_sparse,
   (...)
    192         ensure_2d=ensure_2d,
    193     )
```

File /opt/anaconda3/envs/learn-env/lib/python3.9/site-packages/sklearn/utils/validation.py:1064, in check_array(array, accept_sparse, accept_large_sparse, dtype, order, copy, force_writeable, force_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, estimator, input_name)

```
   1058     raise ValueError(
   1059         "Found array with dim %d. %s expected <= 2."
   1060         % (array.ndim, estimator_name)
   1061     )
   1063 if force_all_finite:
-> 1064     _assert_all_finite(
   1065         array,
   1066         input_name=input_name,
   1067         estimator_name=estimator_name,
   1068         allow_nan=force_all_finite == "allow-nan",
   1069     )
   1071 if copy:
   1072     if _is_numpy_namespace(xp):
   1073         # only make a copy if `array` and `array_orig` may share memory`
```

File /opt/anaconda3/envs/learn-env/lib/python3.9/site-packages/sklearn/utils/validation.py:123, in _assert_all_finite(X, allow_nan, msg_dtype, estimator_name, input_name)

```
    120 if first_pass_isfinite:
    121     return
--> 123 _assert_all_finite_element_wise(
    124     X,
    125     xp=xp,
    126     allow_nan=allow_nan,
    127     msg_dtype=msg_dtype,
    128     estimator_name=estimator_name,
    129     input_name=input_name,
    130 )
```

File /opt/anaconda3/envs/learn-env/lib/python3.9/site-packages/sklearn/utils/validation.py:172, in _assert_all_finite_element_wise(X, xp, allow_nan, msg_dtype, estimator_name, input_name)

```
    155 if estimator_name and input_name == "X" and has_nan_error:
    156     # Improve the error message on how to handle missing val
ues in
    157     # scikit-learn.
    158     msg_err += (
    159         f"\n{estimator_name} does not accept missing values"
    160         " encoded as NaN natively. For supervised learning,
you might want"
   (...)
    170         "#estimators-that-handle-nan-values"
    171     )
--> 172 raise ValueError(msg_err)

ValueError: Input contains NaN.
```

This provides us a general reccomendation for movies based on shared genres.

# K-Nearest Neighbors (Item Based Reccomendation, Collaborative)

Note: This is not a classifier or regression, so there is no train test split or validation. It is another distance comparison

In [619]:
```
from scipy.sparse import csr_matrix
from sklearn.neighbors import NearestNeighbors
from sklearn.model_selection import train_test_split
```

In [620]:
```
#make a pivot table containing movies, ratings, and user Ids
data = pd.merge(movies,ratings)
#fill unrated movies with 0s because KNN does not like nas
user_movie_table = data.pivot_table(index = ["title"],columns = ["u
serId"],values = "rating").fillna(0)
user_movie_table.sample(10)
```

Out[620]:

| userId | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | ... | 601 | 602 | 603 | 604 | 605 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| title | | | | | | | | | | | | | | | | |
| Dampfnudelblues | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Town, The | 0.0 | 4.5 | 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 |
| Nerve | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Evangelion: 2.0 You Can (Not) Advance (Evangerion shin gekijôban: Ha) | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Eraser | 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 | 0.0 |
| Two for the Money | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 11:14 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| My Big Fat Greek Wedding 2 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Buddy Boy | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| À nous la liberté (Freedom for Us) | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | ... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |

10 rows × 610 columns

In [621]:
```
#the test movie will be different because the indexes changed in th
e pivot table
print("Choosen Movie is: ",user_movie_table.index[test_movie])
```

```
Choosen Movie is:  Unfinished Life, An
```

In [622]:
```
#create a real sparse matrix
movie_table_matrix = csr_matrix(user_movie_table.values)
#use cosine distance like earlier
model_knn = NearestNeighbors(metric = 'cosine')
model_knn.fit(movie_table_matrix)
distances, indices = model_knn.kneighbors(user_movie_table.iloc[tes
t_movie,:].values.reshape(1,-1), n_neighbors = 6)
```

In [623]:
```
movie = []
distance = []

for i in range(0, len(distances.flatten())):
    if i != 0:
        movie.append(user_movie_table.index[indices.flatten()[i]])
        distance.append(distances.flatten()[i])

m_series = pd.Series(movie,name='movie')
d_series = pd.Series(distance,name='distance')
recommended = pd.concat([m_series, d_series], axis=1)
recommended = recommended.sort_values('distance',ascending=False)

print('Recommendations for {0}:\n'.format(user_movie_table.index[te
st_movie]))
for i in range(0, recommended.shape[0]):
    print(f'{recommended["movie"].iloc[i]}, with distance of {recom
mended["distance"].iloc[i]}')
```

```
Recommendations for Unfinished Life, An:

Two for the Money, with distance of 0.0
My Name Is Bruce, with distance of 0.0
Bless the Child, with distance of 0.0
Helter Skelter, with distance of 0.0
Many Adventures of Winnie the Pooh, The, with distance of 0.0
```

# Using Surprize! (User-based, Collaborative)

In [16]:
```
!pip install surprise
```

```
Collecting surprise
  Downloading surprise-0.1-py2.py3-none-any.whl.metadata (327 bytes)
Collecting scikit-surprise (from surprise)
  Downloading scikit_surprise-1.1.4.tar.gz (154 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 154.4/154.4 kB 4.4 MB/
s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/pytho
n3.10/dist-packages (from scikit-surprise->surprise) (1.4.2)
Requirement already satisfied: numpy>=1.19.5 in /usr/local/lib/pytho
n3.10/dist-packages (from scikit-surprise->surprise) (1.26.4)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python
3.10/dist-packages (from scikit-surprise->surprise) (1.13.1)
Downloading surprise-0.1-py2.py3-none-any.whl (1.8 kB)
Building wheels for collected packages: scikit-surprise
  Building wheel for scikit-surprise (pyproject.toml) ... done
  Created wheel for scikit-surprise: filename=scikit_surprise-1.1.4-
cp310-cp310-linux_x86_64.whl size=2357285 sha256=45fa9b529d8326256b4
3afcdc6f29e53528aa863a1abbe4bbd4b56a30662590f
  Stored in directory: /root/.cache/pip/wheels/4b/3f/df/6acbf0a40397
d9bf3ff97f582cc22fb9ce66adde75bc71fd54
Successfully built scikit-surprise
Installing collected packages: scikit-surprise, surprise
Successfully installed scikit-surprise-1.1.4 surprise-0.1
```

In [17]:
```
from surprise import SVD, accuracy
from surprise import Dataset, Reader
from surprise.model_selection import cross_validate
from surprise.model_selection.split import train_test_split
from collections import defaultdict
```

In [18]:
```
# preprocessing the data
reader = Reader(rating_scale=(1,5))
data = Dataset.load_from_df(ratings[['userId','movieId','rating']],
reader)
train, test = train_test_split(data, test_size=.2, random_state=42)

# initial model
algo = SVD(random_state = 42)
algo.fit(train)
predictions = algo.test(test)

# evaluate the rmse result of the prediction and ground thuth
accuracy.rmse(predictions)
```

```
RMSE: 0.8807
```

Out[18]:
```
0.8807462819979623
```

In [21]:
```
def get_top_picks(predictions, num_recs = 10):
    """Return the top-N recommendation for each user from a set of
predictions.

    predictions(list of Prediction objects): The list of prediction
s, as returned by suprise's algorithm.
    num_recs (int): The number of recommendation to output for each
user. Default is 10.

    Returns:
    A dict where keys are user (raw) ids and values are lists of tu
ples:
        [(raw item id, rating estimation), ...] of size num_recs.
    """

    # First map the predictions to each user.
    top_n = defaultdict(list)
    for uid, iid, true_r, est, _ in predictions:
        top_n[uid].append((iid, est))

    # Then sort the predictions for each user and retrieve the k hi
ghest ones.
    for uid, user_ratings in top_n.items():
        user_ratings.sort(key=lambda x: x[1], reverse=True)
        top_n[uid] = user_ratings[:num_recs]

    return top_n

top_n = get_top_picks(predictions, num_recs=10)

# Print the recommended items for each user
for uid, user_ratings in top_n.items():
    print(uid, [iid for (iid, _) in user_ratings])
```

```
140 [2947, 953, 1036, 5995, 6787, 2067, 1408, 48516, 1234, 3578]
603 [1221, 290, 2858, 1172, 1193, 85, 2692, 36, 1248, 912]
438 [4011, 4223, 34405, 4993, 7451, 4995, 5902, 1732, 8949, 1196]
433 [608, 1089, 296, 164179]
474 [1283, 1172, 5618, 720, 260, 914, 17, 48516, 3083, 3451]
304 [318, 1035, 1704, 2502, 593, 457, 356, 60, 733, 474]
298 [260, 5618, 1210, 2858, 1625, 54503, 1285, 48516, 3578, 1222]
131 [1213, 1193, 1228, 293, 593, 1200, 1617, 4226, 1136, 1288]
288 [1258, 8874, 908, 7153, 1250, 2571, 2959, 1261, 2028, 3507]
448 [296, 32, 112552, 1198, 48516, 1200, 109374, 778, 1214, 1704]
284 [296, 318, 356, 590, 342, 357, 104, 380, 112, 433]
331 [79132, 58559, 134130, 55765, 33794, 54997, 8636, 48394, 34, 693
6]
325 [969, 356, 2186, 1682, 1090, 1304, 919, 3267, 1280, 1610]
504 [4027, 296, 5060, 4995, 3897, 4306, 3556, 5445, 5673, 2797]
286 [2329, 4973, 2028, 4848, 4571, 364, 6377, 2692, 150, 5989]
232 [527, 5418, 78499, 1198, 745, 58559, 68157, 44191, 7143, 44199]
6 [50, 475, 589, 553, 293, 891, 981, 151, 296, 302]
54 [593, 588, 21, 161, 339, 454, 592, 288]
264 [1035, 4995, 4886, 1196, 480, 4896, 7303, 2085, 41566, 253]
45 [2028, 3578, 1196, 3972, 903, 858, 1961, 296, 1036, 3897]
525 [356, 4973, 2324, 527, 1196, 54881, 81834, 46578, 5971, 96821]
577 [318, 260, 1198, 1210, 858, 1213, 2028, 1304, 1303, 919]
226 [4993, 2858, 1204, 7153, 1197, 1222, 6377, 2398, 3671, 608]
597 [1221, 1079, 1263, 2692, 1394, 441, 541, 1207, 1732, 745]
20 [5014, 3535, 1073, 3189, 5952, 3396, 2804, 4306, 2083, 5630]
386 [110, 293, 608, 349, 780, 733, 161, 34, 590, 198]
405 [4226, 1673, 1732, 68157, 16, 4011, 32, 1209, 110, 34405]
136 [280, 457, 333, 225, 235, 551, 165, 590, 266, 440]
275 [858, 908, 356, 4973, 3996, 50, 1199, 910, 951, 1719]
177 [2502, 1089, 2959, 1244, 16, 1193, 2194, 3175, 1923, 1206]
260 [904, 2959, 2329, 1089, 910, 3897, 1148, 5902, 2186, 6620]
599 [1247, 356, 260, 58559, 1219, 8874, 1291, 904, 1921, 6874]
57 [1196, 296, 356, 858, 2858, 1249, 969, 260, 1221, 593]
534 [89745, 49272, 1215, 4306, 8368, 593, 356, 111759, 6377, 2028]
274 [1221, 1704, 3703, 1276, 1201, 1210, 4993, 1261, 608, 904]
565 [318, 34, 509, 592, 344, 231, 153, 420]
451 [593, 1073, 6, 1356, 661, 736]
368 [904, 3703, 2028, 608, 1732, 1090, 1220, 3505, 589, 2949]
198 [318, 356, 4973, 2329, 1197, 1258, 2762, 608, 1250, 5013]
269 [494, 780, 81, 63, 79]
68 [7153, 2248, 1197, 912, 4878, 3996, 1617, 30749, 79702, 1653]
228 [2571, 50, 1801, 362, 3005, 1091]
561 [2959, 1196, 1732, 1291, 1250, 913, 1210, 8874, 1240, 110]
201 [1198, 260, 296, 2858, 1240, 931, 1307, 1584, 2762, 1923]
351 [7153, 79132, 356, 58559, 2762, 2231, 4720, 86781, 85342, 8914]
200 [51255, 4963, 2858, 8949, 8874, 49530, 6874, 8958, 2918, 2571]
591 [356, 2762, 2541, 2763, 2628, 2394, 2712, 24, 2722, 3409]
482 [7153, 1246, 6539, 1270, 1200, 16, 2792, 2640, 62, 8529]
317 [1208, 1089, 750, 2571, 2692, 7147, 4239, 109487, 30707, 2324]
580 [1221, 5902, 296, 1199, 2571, 1089, 5991, 2329, 38061, 1208]
105 [2324, 1198, 7090, 6016, 608, 79132, 1178, 44191, 68157, 8961]
462 [1276, 2797, 356, 1235, 2959, 2186, 6787, 1089, 55442, 4105]
339 [356, 1203, 4995, 1247, 68157, 91529, 56782, 5989, 1258, 55721]
307 [356, 858, 608, 58559, 1258, 904, 54503, 32, 1, 2502]
18 [50, 1252, 778, 2571, 4011, 5952, 1223, 2329, 5995, 1704]
```

```
370 [2959, 1197, 1732, 2692, 8961, 5618, 4306, 1288, 1285, 2115]
156 [923, 1208, 50, 1252, 2997, 1136, 1089, 593, 1278, 3578]
48 [5903, 5349, 1584, 4886, 2353, 1356, 5882, 2081, 1485]
362 [2329, 1213, 318, 104879, 1089, 2571, 1208, 6874, 3147, 89745]
98 [79132, 1258, 2324, 7153, 3114, 122916, 858, 1036, 52975, 1219]
249 [527, 260, 32, 2502, 1089, 112552, 57669, 4011, 4226, 1203]
483 [1270, 1210, 5952, 60069, 4226, 914, 8874, 2997, 6016, 1215]
478 [1225, 104, 33493, 266]
486 [110, 457, 21, 733, 380, 377, 112, 10, 494, 780]
314 [1, 497, 296, 6, 1356, 440, 349, 594, 236, 107]
119 [58559, 3275, 6377, 112552, 2571, 68157, 50872, 63082, 142488, 3
64]
592 [356, 480, 353, 377, 163, 551, 2, 597, 780, 367]
294 [1204, 1221, 3424, 1215, 555, 1374, 866, 2871, 2396, 1566]
82 [1210, 5669, 5991, 4246, 364, 2355, 293, 2011, 733, 3793]
361 [150, 5989, 4963, 5444, 52973, 8968, 1732, 1923, 6365, 1374]
8 [318, 593, 296, 457, 590, 364, 380, 236, 586]
159 [1197, 97304, 30707, 72641, 4886, 166643, 6373, 4014, 1784, 8977
4]
414 [858, 1732, 293, 1136, 1198, 5952, 68237, 1945, 5954, 741]
590 [858, 1221, 1203, 898, 3451, 4993, 1303, 953, 1291, 8636]
84 [904, 1208, 916, 1288, 1276, 910, 1197, 527, 1203, 356]
89 [1136, 910, 6936, 6942, 108932, 5218, 918, 3088, 911, 149406]
608 [527, 2028, 1210, 5349, 296, 4878, 8784, 1036, 1089, 858]
316 [6874, 4027, 3751, 4720, 1892, 4641, 4816, 342, 48, 1676]
318 [1673, 913, 68157, 223, 89745, 5617, 2542, 4011, 29, 6502]
246 [6016, 5902, 4973, 5952, 68237, 2791, 62336, 3910, 589, 69481]
387 [858, 912, 8874, 4993, 1207, 2959, 32, 5618, 933, 1201]
220 [912, 1193, 3147, 58559, 1200, 2947, 4886, 1210, 48774, 6539]
337 [318, 733, 104, 553, 17, 107, 494, 802, 141, 135]
343 [260, 296, 51255, 1214, 54503, 3298, 3993]
358 [88163, 56715, 2396, 87028, 52328, 3882]
115 [1617, 3037, 1196, 3703, 1682, 260, 2762, 590, 1907, 480]
600 [2329, 902, 5617, 2571, 1175, 2300, 2160, 1291, 1394, 5225]
150 [32, 805, 1073, 62, 141, 3, 786]
103 [356, 1222, 318, 1228, 48516, 7361, 1278, 81834, 2329, 1198]
509 [96821, 91529, 89745, 109487, 73017, 99114, 50872, 8961, 6377, 2
248]
563 [914, 919, 40815, 46578, 1028, 916, 106920, 68954, 97921, 1968]
160 [593, 1200, 1258, 3147, 1028, 541, 3034, 2716, 2115, 2011]
335 [318, 1639, 2001, 2424, 466]
553 [1221, 608, 7361, 55247, 1466, 2395, 6, 33166, 16, 3424]
403 [2329, 3703, 2420, 3107, 3006, 61465, 61697, 1805, 2724]
529 [260, 736, 661, 788]
234 [364, 1967, 1291, 2000, 1012, 2640, 590, 1573, 4519, 4085]
313 [2762, 1214, 1247, 6, 1235, 2529, 1748, 858, 1212, 1387]
546 [2791, 1673, 1035, 3917, 3868, 3985, 3939, 240, 370, 3409]
345 [2791, 932, 1214, 3489, 924, 1186, 8261, 38164, 33085, 5900]
520 [1197, 4027, 1198, 1278, 356, 1215, 1214, 1234, 1079, 2716]
10 [2959, 6377, 7451, 106489, 2571, 4993, 8961, 73017, 104374, 2762]
464 [5952, 2571, 318, 858, 60069, 2019, 2959, 1210, 1704, 5989]
610 [750, 1208, 50, 2571, 7438, 3578, 1080, 1198, 1240, 1199]
122 [356, 58559, 1732, 1270, 1673, 2502, 6807, 3949, 1090, 31658]
586 [58559, 858, 3147, 2762, 96079, 33794, 1291, 1374, 6333, 122886]
606 [904, 5618, 1244, 1394, 750, 44195, 1136, 27773, 1250, 2160]
365 [356, 96821, 8636, 1704, 58559, 593, 1393, 541, 92420, 69844]
```

```
382 [1204, 60069, 2959, 1201, 260, 912, 58559, 7153, 105504, 37741]
242 [296, 111, 457, 225, 253, 153]
453 [1136, 1221, 3108, 2000, 1210, 4235, 2329, 2502, 5876, 4225]
221 [1136, 1250, 1199, 1172, 2019, 4973, 904, 1252, 2959, 928]
517 [356, 3967, 364, 595, 457, 1945, 5991, 1148, 51255, 912]
1 [3147, 2959, 260, 2502, 608, 923, 1213, 480, 356, 2692]
372 [1246, 1208, 260, 858, 1249, 953, 62, 293, 110, 111]
19 [1732, 1193, 1073, 933, 1610, 1784, 1086, 1200, 2011, 924]
217 [1210, 1945, 1291, 1215, 2231, 2268, 1912, 296, 1079, 1242]
418 [527, 5952, 1036, 296, 1276, 1270, 593, 1246, 1219, 778]
587 [2571, 1235, 933, 1079, 3435, 1101, 3244, 2144, 1077, 3260]
51 [2791, 1090, 5060, 1235, 1249, 3683, 1186, 1200, 1272, 54001]
380 [260, 1193, 1214, 5618, 541, 589, 81834, 1199, 50, 1213]
555 [800, 2248, 1261, 1206, 969, 1200, 110, 1394, 3793, 1266]
312 [111, 1215, 2288, 1175, 1200, 2186, 5989, 480, 1748, 1206]
596 [79132, 8874, 4993, 1196, 96821, 1197, 3147, 589, 2762, 260]
480 [527, 593, 1732, 1208, 1089, 2788, 2502, 8874, 2028, 1198]
7 [750, 593, 1617, 8961, 33794, 920, 1682, 150, 2529, 4886]
469 [541, 2571, 260, 1258, 1244, 1208, 527, 1089, 110, 1285]
551 [7153, 5952, 356, 4993, 1270, 72998, 88125, 589, 5418, 73017]
589 [356, 246, 593, 111, 337, 150, 454, 25, 1173, 481]
116 [1225, 1193, 1704, 1198, 858, 899, 1183, 2000, 1954, 1387]
543 [2571, 50872, 364, 6377, 63082, 5225, 35836, 2382, 380, 4994]
268 [858, 1213, 1193, 356, 2130, 2973, 1923, 2245, 2952, 1964]
222 [296, 58559, 1249, 2571, 4995, 1213, 80463, 2028, 7143, 68954]
91 [858, 1200, 541, 5618, 1079, 1967, 1208, 1249, 8874, 1201]
111 [356, 68157, 2918, 60069, 4226, 3052, 54995, 6377, 54190, 1884]
280 [81845, 4246, 910, 919, 91077, 4878, 1207, 93510, 1, 61024]
440 [541, 4361, 2359, 1186, 2707, 1093]
109 [296, 527, 553, 354, 290, 348, 18, 524, 154, 230]
319 [2571, 2959, 60069, 6942, 122886, 137857, 1907, 2687]
202 [1208, 3147, 3703, 1247, 2028, 541, 1136, 3552, 1278, 1682]
62 [318, 527, 1198, 593, 71535, 2028, 38061, 122926, 112552, 49272]
210 [1198, 1196, 78499, 70286, 7153, 4993, 180031, 116797, 98124, 62
999]
425 [741, 1221, 7090, 79132, 4011, 2997, 441, 1653, 3114, 2028]
255 [1270, 1394, 3037, 2064, 2716, 1663, 1220, 3392, 34, 1739]
85 [1584, 1446, 538, 1693, 53, 1519, 1643, 1694]
52 [2959, 7153, 8874, 48516, 60069, 48774, 49272, 6874, 54995, 5669]
514 [1197, 1387, 356, 1210, 1196, 296, 106782, 134130, 480, 1266]
473 [318, 2329, 3897, 6281, 2253]
334 [356, 6016, 4720, 1291, 260, 3578, 72226, 6377, 30749, 1258]
434 [2959, 7361, 541, 32, 1291, 47, 1732, 68157, 1208, 2700]
297 [293, 1089, 457, 1610, 628, 1358, 22, 150, 1608, 1407]
290 [50, 527, 898, 1266, 296, 111, 1193, 1198, 2762, 908]
391 [4226, 296, 1221, 1201, 1204, 1199, 593, 4973, 2692, 1200]
247 [50, 527, 6016, 2762, 608, 296, 293, 7153, 112852, 1210]
64 [318, 356, 778, 7361, 111, 50, 1215, 3108, 110, 1276]
59 [2571, 1221, 1233, 2076, 2067, 906, 2944, 2550, 2726, 733]
328 [2959, 99114, 6807, 1028, 1203, 1219, 4011, 111, 80463, 924]
420 [296, 7156, 8874, 899, 1136, 4973, 5902, 32587, 32, 4878]
541 [50, 457, 497, 597, 524, 480, 357, 594, 350, 349]
454 [296, 6874, 101, 82, 4754, 56339, 735, 6629, 7116]
432 [912, 5952, 4886, 4973, 4306, 7153, 3949, 3147, 6947, 69481]
510 [1196, 750, 1265, 1262, 527, 1247, 1248, 2918, 2804, 926]
573 [4011, 2762, 2502, 858, 5952, 1221, 51255, 1036, 1198, 32587]
```

```
233 [318, 1136, 1225, 778, 296, 608, 79132, 7371, 8961, 92259]
104 [7361, 80549, 919, 4226, 5952, 595, 3060, 38038, 8533, 8961]
95 [5618, 1198, 7361, 6711, 1228, 913, 32, 5377, 1748, 4979]
470 [50, 296, 457, 1, 590, 588, 261, 34, 515, 339]
495 [2959, 58998, 1193, 6942, 59369, 4963, 112852, 1259, 74458, 8156
2]
477 [1136, 260, 4226, 2997, 1283, 2858, 318, 593, 1198, 924]
437 [246, 296, 1208, 1247, 1225, 1210, 36, 1393, 733, 1270]
186 [908, 593, 589, 1136, 4027, 3578, 1197, 260, 2064, 1]
193 [318, 924, 2959, 480, 55442, 26116]
76 [1258, 56782, 1206, 5952, 40815, 1214, 1247, 858, 2762, 923]
522 [260, 2571, 1270, 1196, 88125, 2502, 7361, 2028, 1374, 223]
282 [1221, 4226, 6874, 1203, 73017, 1208, 1028, 2064, 1610, 2268]
367 [3088, 1704, 1136, 3360, 1246, 260, 953, 2078, 899, 919]
366 [4993, 91529, 68157, 44191, 6874, 110, 2028, 3578, 8961, 115617]
416 [6874, 1199, 2716, 527, 5902, 5952, 247, 6365, 8360, 327]
230 [3578, 1200, 6502, 4034, 589, 6333, 590, 912, 4022, 1573]
171 [593, 1198, 29, 1270, 111, 471, 36, 1151, 1025, 861]
28 [3508, 2959, 1276, 858, 2542, 38061, 44191, 4011, 1653, 2762]
570 [7153, 903, 1246, 110, 1258, 1291, 1080, 5989, 1198, 2858]
161 [318, 1148, 745, 2336, 1019, 48, 5643, 4980]
182 [858, 720, 7361, 4993, 215, 4226, 1213, 2692, 1214, 4973]
523 [31658, 4973, 318, 2959, 4993, 56367, 74458, 134130, 122904, 821
69]
204 [4973, 4993, 33794, 30749, 48394, 7147, 85354, 54372, 5949, 7126
4]
209 [68157, 7153, 356, 593, 3578, 2571, 1291, 68954, 60684, 64614]
544 [40, 68, 619, 839]
352 [1222, 1200, 527, 4226, 54259, 3083, 4034, 1704, 106782, 54001]
430 [2858, 1210, 1374, 66097, 4022, 34, 2321, 4446, 3745, 2100]
356 [1732, 2959, 1223, 6807, 1884, 46578, 1198, 2997, 3471, 60069]
146 [58803, 51662, 7293, 3257]
461 [356, 318, 1704, 106782, 3578, 539, 91500]
63 [527, 1213, 1208, 2329, 1282, 4011, 27846, 8874, 1291, 91529]
558 [223, 2028, 4306, 3831, 1923, 4308, 5388, 5128, 597, 2568]
263 [5952, 1225, 457, 1136, 4034, 150, 1148, 4886, 110, 3967]
94 [318, 337, 32, 165, 377, 2, 434, 432]
199 [1230, 4973, 1228, 69481, 1233, 2692, 904, 5418, 56782, 910]
566 [110, 541, 265, 150, 597, 165, 345, 261, 588, 357]
21 [3578, 1291, 88125, 168248, 1036, 68954, 50872, 1198, 541, 3948]
271 [1225, 3897, 6502, 2288, 55765, 1127, 1047, 3701, 1805, 4370]
342 [2093, 2987, 2723, 2605, 2433, 3004, 2722, 2827, 2701]
439 [4993, 1291, 589, 8961, 1240]
475 [5618, 58559, 81834, 89745, 364, 1198, 589, 166528, 139385, 896
1]
569 [50, 349, 10, 329, 231]
223 [2571, 260, 7361, 4973, 2762, 608, 3793, 2396, 2683, 55854]
100 [1213, 2329, 1203, 1278, 1674, 838, 223, 5989, 17, 2005]
333 [1193, 337, 924, 3753, 3580, 3624, 3646]
153 [3949, 1208, 112552, 778, 1028, 91529, 1246, 72641, 364, 1193]
241 [50, 7361, 56367, 6016, 4973, 5878, 2858, 215, 4878, 112852]
301 [608, 56782, 7361, 5418, 2396, 6377, 17, 4973, 2858, 2716]
507 [110, 266, 316]
167 [778, 260, 1196, 3275, 1291, 33794, 110, 4016, 4011, 1200]
5 [296, 608, 589, 34, 247, 594, 232, 596, 300, 253]
112 [58559, 778, 110, 1270, 16, 1393, 17, 508, 2, 3717]
```

```
547 [1288, 1299, 1199, 3039, 1224, 1131, 2859, 2919, 2840]
471 [296, 2324, 44191]
460 [4995, 1, 4993, 44199, 5989, 8665, 4886, 40815, 1307, 2112]
452 [1198, 2858, 1221, 3175, 2028, 110, 1214, 1997, 4571, 589]
4 [296, 32, 1225, 4029, 2571, 1282, 4226, 441, 1079, 2583]
429 [349, 292, 339, 164, 329, 274, 380, 351, 270, 185]
315 [914, 1035, 1207, 2729, 2863, 2136]
463 [1221, 2019, 2167, 5378, 1552, 2006, 1320]
58 [50, 362, 345, 508, 165, 590, 588, 333, 337, 349]
479 [593, 1278, 1210, 5404, 910, 110, 3111, 1372, 3869, 1968]
33 [1213, 2324, 1059, 28, 457, 34, 307, 497, 2501, 1682]
185 [5952, 223, 1, 5418, 3897, 2078, 4995, 3481, 595, 2355]
137 [1225, 1288, 904, 913, 593, 1200, 951, 1244, 968, 778]
38 [318, 246, 364, 110, 595, 17, 151, 597, 224, 509]
158 [3363, 2948, 27674, 2872, 3301, 327]
412 [910, 903, 1197, 1304, 1252, 1641, 457, 2076, 1219, 1610]
579 [1653, 1907, 110, 2687, 2087, 3453, 500, 3176, 1608, 2599]
149 [1214, 1374, 1270, 589, 1372, 2, 1371, 1544, 405, 173]
108 [497, 4144, 1968, 1095, 800, 2947, 838, 541, 5989, 5135]
574 [380, 292, 344, 231]
562 [1704, 356, 5618, 4034, 6377, 48394, 1387, 8874, 93840, 1258]
527 [1997, 3147, 3114, 1246, 590, 2099, 595, 919, 3044, 2947]
389 [260, 648, 17, 736, 141, 112, 637, 743, 762]
305 [58559, 7143, 1089, 593, 47, 1207, 56782, 4973, 1252, 8874]
168 [296, 908, 1222, 356, 6016, 1221, 44555, 57669, 1247, 2890]
42 [356, 2268, 260, 1213, 1231, 457, 2716, 1079, 1625, 1036]
385 [858, 593, 1213, 356, 5060, 1228, 150, 1035, 588, 1527]
419 [296, 68157, 55765, 48780, 175, 3578, 58559, 337, 1240, 64969]
66 [1222, 296, 1089, 3578, 1258, 922, 1136, 4226, 1196, 1213]
344 [3147, 4973, 4022, 111362, 2628, 112852, 7099, 59315, 108190, 31
6]
132 [318, 50, 1244, 2959, 1028, 5971, 1089, 3949, 1210, 31658]
251 [2571, 293]
424 [74458, 8950, 1252, 1207, 47, 1923, 2762, 2076, 1219, 296]
179 [296, 590, 253, 648, 34, 349, 367, 671, 165, 376]
593 [356, 110, 1221, 260, 593, 1196, 3578, 2858, 1411, 3210]
326 [4973, 2959, 99114, 92259, 5952, 318, 2324, 593, 7153, 2028]
466 [356, 89745, 78499, 76093, 92259, 5989, 64716, 69844, 4993, 5428
6]
332 [750, 1249, 1617, 4226, 3578, 2762, 6711, 1213, 6787, 7153]
552 [2571, 7153, 2502, 3435, 1259, 1610, 2125, 1220, 2289, 3148]
144 [2959, 1193, 2858, 3147, 4306, 1079, 8368, 2728, 1035, 17]
468 [296, 1, 266, 165, 153]
74 [1208, 1673, 44555, 56782, 27773, 3083, 1212, 6666, 3424, 3949]
421 [1193, 1247, 1178, 81845, 2398, 4223, 1099, 3439]
489 [908, 1250, 1079, 4993, 1028, 5995, 4973, 1214, 1089, 54001]
212 [356, 608, 80463, 1136, 2959, 27773, 68157, 6377, 541, 71535]
50 [1204, 2959, 5618, 1136, 1148, 1281, 158238, 909, 111, 46578]
190 [2959, 1193, 8961, 5902, 4995, 4896, 69844, 7451, 8950, 8533]
602 [29, 16, 293, 457, 356, 47, 593, 110, 371, 527]
542 [1282, 3052, 8360, 48780, 46976, 3911, 5418, 4963, 924, 500]
584 [1, 595, 349, 252, 597, 282, 367, 60, 153, 186]
560 [741, 608, 356, 50, 778, 55118, 8874, 4226, 1080, 923]
93 [1704, 150, 380, 2193, 1721, 367, 1676, 736, 558, 1606]
524 [1193, 912, 2019, 1272, 1266, 86, 1275, 1242, 151, 1254]
392 [1198, 4085, 2707]
```

```
29 [48516, 150, 1270, 1222, 1262, 1198, 99114, 1201, 6502, 80489]
601 [2571, 2959, 1198, 79132, 2019, 953, 1207, 74458, 81845, 1584]
409 [1197, 3468, 1225, 1193, 260, 912, 608, 1230, 2716, 3363]
278 [203, 2369]
23 [1222, 1252, 904, 4226, 1221, 1193, 750, 1136, 1089, 1748]
410 [1237, 2858, 1248, 903, 6460, 1251, 290, 3224, 527, 2064]
537 [89745, 48516, 4963, 80463, 106782, 8533, 53125, 82053, 8984, 86
911]
572 [356, 1225, 260, 1079, 1235, 912, 590, 2788, 3037, 1682]
395 [165, 10, 588, 34, 466, 485, 168, 355]
303 [1247, 1196, 260, 2797, 1210, 1584, 2640, 5544, 1527, 3175]
322 [858, 1288, 356, 318, 778, 1258, 912, 1625, 4993, 4226]
357 [58559, 1089, 260, 1136, 2329, 1206, 1104, 1252, 356, 81834]
273 [589, 595, 337, 34, 171, 587, 282, 367, 592, 532]
56 [296, 69, 454, 225, 165, 288, 344, 420, 196]
330 [296, 1193, 1196, 48516, 858, 1208, 2028, 1198, 3703, 457]
265 [1193, 480, 2289, 110, 111, 3148, 1270, 1097, 3255, 1271]
117 [555, 32, 1028, 527, 610, 235, 262, 708, 480, 62]
106 [318, 4896, 106489, 5349, 86880, 59784, 2617, 76175]
381 [1197, 1246, 1035, 2804, 608, 2959, 8368, 364, 595, 7147]
107 [318, 111, 110, 161, 292, 380, 140, 317, 160, 344]
465 [2028, 1214, 356, 3671, 1200, 924, 2176, 2944, 377, 1370]
227 [115713, 1682, 2571, 106782, 99813, 122882, 7371, 85342, 70286,
8914]
369 [1208, 1682, 1196, 260, 4878, 1732, 1884, 3949, 2692, 2918]
157 [1270, 2686, 3174, 2490, 3752, 2724, 2392]
41 [7361, 4973, 4878, 778, 2028, 79132, 8874, 1206, 55820, 4720]
578 [81845, 49772, 71823, 68269, 2496]
354 [1208, 1225, 4973, 2324, 1201, 2502, 4306, 30707, 4262, 296]
139 [60684, 91529, 3578, 51255, 5903, 8961, 56757, 109374, 70286, 40
27]
15 [1196, 134130, 68237, 91529, 1265, 99114, 5952, 7438, 48774, 127
0]
125 [68157, 60069, 3949, 94959, 1215, 97913, 501, 119145, 164179, 33
794]
350 [260, 648, 1073, 112, 141, 494, 376, 802, 852, 12]
310 [750, 1204, 1233, 529, 246, 16, 1101, 2944, 1396, 1080]
219 [1250, 318, 44195, 50, 4878, 608, 1288, 3949, 913, 1198]
457 [1197, 1136, 1270, 110, 2762, 2600, 260, 1374, 1371, 2572]
373 [527, 50, 32, 475, 111, 280, 265, 300, 253, 163]
299 [1997, 3300, 188, 3555]
196 [2571, 2762, 142488, 70286, 62, 41569]
498 [161, 150, 588, 339, 454, 592, 231, 7, 344]
43 [364, 296, 110, 318, 589, 553, 610, 520, 350, 597]
292 [457, 8368, 88810, 1203, 1, 914, 2716, 79132, 1198, 1276]
484 [904, 919, 3911, 1200, 4776, 175, 60684, 87306, 1721, 733]
211 [318, 91529, 1222, 33794, 79132, 74458, 5418, 77561, 5445, 7028
6]
428 [296, 1196, 6874, 541, 3740, 1748, 5952, 2571, 1206, 590]
256 [92259, 1198, 30707, 1196, 115713, 68358, 5444, 3300, 7323, 7028
6]
377 [926, 1244, 1927, 1200, 5919, 4039, 6270, 1503, 87287, 92760]
252 [78499, 50872, 115617, 98491, 166461, 152081, 95167, 143355]
353 [32, 293, 553, 161, 319, 198, 340, 163, 236, 288]
556 [112852, 69844, 40815, 5816, 49649]
347 [595, 150, 10, 39, 165]
```

```
607 [1213, 11, 3114, 3147, 1258, 593, 2918, 3347, 780, 1270]
431 [903, 1625, 555, 1639, 172, 158]
450 [260, 111, 3949, 1387, 3535, 3915, 3969, 3404, 3955, 3354]
244 [1196, 1228, 2028, 3703, 1183, 1527, 2916, 547, 1302, 3300]
604 [364, 14, 708, 330, 235, 165, 329, 151, 636, 611]
266 [1275, 50, 1136, 1199, 1213, 1, 1215, 1220, 1210, 1222]
564 [1270, 106782, 93510, 104211, 60756, 61024, 7346, 160271, 81784,
80693]
17 [1222, 1089, 1198, 7153, 1197, 3681, 5618, 80463, 260, 2019]
32 [608, 912, 1073, 58, 733, 337, 1183, 272, 345, 252]
279 [1213, 1198, 1214, 48780, 57669, 1200, 139385, 99114, 608, 9793
8]
456 [1, 706, 1167, 9, 640, 788, 810]
187 [2076, 5952, 44195, 593, 1148, 51540, 1222, 4027, 8950, 1617]
143 [96821, 6377, 71535, 56949, 70183, 103606, 69122, 84374, 61250,
7255]
96 [858, 1198, 3578, 2028, 50, 307, 1221, 527, 306, 272]
203 [110, 8368, 5418, 8665, 4306, 3255, 7458, 98809, 2001, 2302]
571 [1278, 1350, 1269, 1214, 3018, 2902, 2868, 1980, 426, 724]
506 [38061, 54272, 47629, 61240, 41571, 52975]
502 [318, 1674, 1625, 2081, 8529, 45, 2657, 4412]
415 [2959, 1252, 1258, 1276, 930, 44195, 111, 3507, 4499, 480]
12 [40629, 920, 2421, 168, 2485, 63992]
9 [1198, 923, 187, 5952, 2300, 5965, 41, 4131, 3735, 5451]
309 [904, 1283, 1387, 930, 1207, 2571, 30707, 933, 3996, 1265]
559 [296, 150, 32, 1073, 161, 474, 1527, 508, 36, 587]
191 [261, 6, 47, 58, 25, 150, 72, 496, 178, 194]
390 [6016, 2288, 3910, 7371, 2872, 4239, 6350, 34437, 4014, 64957]
238 [1953, 1287, 1961, 4954, 4329, 4917, 4947, 4901, 4238, 1092]
181 [356, 593, 47, 230, 86, 380, 236, 281, 316, 61]
404 [47, 296, 165, 356, 349, 590, 553, 357, 440, 515]
195 [608, 4022, 1210, 1208, 2871, 1249, 1270, 3424, 3347, 2005]
174 [296, 350, 454, 589, 648, 440, 520, 158, 432, 196]
183 [1201, 589, 1222, 1210, 474, 1610, 2534, 3107, 2662, 1527]
575 [2623, 2504, 2617, 2574]
348 [260, 527, 92259, 2065, 6953, 1801, 2325, 4958, 1772, 2616]
213 [4306, 5952, 1954, 1036, 6539, 7147, 48304, 2409, 1370, 3005]
248 [166528, 59315, 176371, 53125, 2115, 189333, 159093, 151763, 122
896]
567 [122916, 76093, 50, 101, 58559, 166461, 163645, 48780, 2138, 718
99]
88 [56782, 54286, 1193, 68157, 4262, 44191, 6874, 16, 1625, 3255]
605 [296, 2791, 7153, 8368, 356, 1028, 590, 589, 1269, 45722]
237 [50, 46976, 106782, 94959, 54286, 36529, 5679, 1892]
197 [858, 589, 2025, 1755, 235, 153, 1882]
287 [1136, 50, 223, 527, 1921, 4878, 8368, 2193, 1732, 6944]
422 [750, 923, 3037, 4855, 968, 3175, 4022, 3095, 2670, 3798]
496 [904, 912, 106920, 109374, 104841, 7151, 40826, 8865]
408 [741, 1210, 3578, 48394, 91529, 1198, 1036, 1196, 8641, 166528]
240 [110, 595, 296, 590, 362, 150, 736, 516, 596, 1359]
129 [2959, 1222, 260, 6874, 5618, 4896, 7143, 6502, 480, 1625]
308 [5618, 54001, 50, 6016, 2248, 45517, 161, 5444, 2959, 6377]
323 [318, 2571, 527, 474, 1, 36, 161, 551, 377, 300]
11 [457, 1704, 589, 1408, 1518, 493, 1101, 1721, 1687, 2002]
16 [4973, 608, 912, 2571, 68954, 7361, 1252, 2329, 27773, 1213]
254 [2959, 527, 1, 6377, 293, 260, 356, 44191, 48780, 1200]
```

```
36 [904, 4995, 1234, 1222, 1617, 1262, 6662, 1372, 1358, 4085]
147 [1234, 552, 151, 2302]
87 [4023]
65 [58559, 7153, 3147, 48780, 64614, 2028, 79132, 8798, 91542, 1036]
110 [608, 2858, 1089, 541, 1198, 2571, 2028, 46578, 47, 4881]
445 [2959, 7361, 293, 5995, 608, 27773, 34405, 4993, 8014, 76093]
375 [3578, 2762, 1923, 597, 2628]
75 [912, 1036, 2571, 1196, 1291, 7123, 903, 1234, 928, 4993]
80 [79132, 32, 8950, 84152, 76251, 48394, 65216, 53322, 2763, 5630]
413 [858, 1213, 54286, 2858, 99114, 2706, 6378, 64957, 736, 2424]
141 [4011, 260, 1704, 364, 2804, 56367, 7143, 55820, 44191, 51255]
169 [4896, 4993, 364, 1682, 1307, 7147, 5816, 4022, 6339, 5989]
505 [8874, 2083, 4105, 1033, 1029, 327]
148 [60069, 76093, 68954, 6377, 116797, 44191, 31658, 110102, 160718
, 4308]
124 [1223, 593, 3328, 1358, 6188, 1517, 1092, 520, 344]
113 [858, 933, 2132, 1357, 2076, 1230, 838, 1073, 4007, 3551]
214 [1356, 780, 141, 104, 628, 743]
548 [2716, 122904, 5955, 1477, 74532, 3263]
476 [34, 588, 1, 594, 597, 10, 592, 329, 587, 289]
402 [589, 593, 780, 36, 380, 9, 786, 5, 434]
359 [2571, 2324, 1270, 296, 6377, 4034, 55247, 1721, 3471, 1092]
277 [1356, 1073, 112, 780, 836]
397 [8645, 4017, 3388]
162 [50, 16, 150, 17, 508, 300, 534, 515, 31, 224]
101 [2997, 2318, 3174, 2841, 2908, 233, 3160, 2712, 2560, 3717]
138 [3996, 1959, 2924, 1019, 2582, 1805, 2116]
535 [2599, 2583, 2573, 2860, 2724, 2829, 2805]
239 [296, 58559, 1193, 3147, 1136, 356, 1266, 1704, 2571, 8961]
492 [858, 778, 32, 36, 1073, 1061, 647, 628, 112, 830]
145 [50, 296, 150, 590, 588, 165, 380, 592]
513 [750, 1210, 593, 7153, 6787, 32, 1304, 1374, 7]
34 [318, 4973, 4874, 145, 6378, 70, 2617, 45499, 44974, 31184]
311 [1247, 2291, 589, 2797, 2174, 715, 6561]
594 [1393, 5299, 6577, 4902, 457, 920, 1220, 588, 1370, 799]
531 [1261, 1215, 1198, 733, 4284, 4447, 3699]
47 [508, 139385, 2908, 104841, 80463, 539, 5014, 3408, 1393, 8464]
49 [2571, 168252, 5218]
336 [858, 2959, 4973, 2329, 2762, 150, 7153, 4886, 3174, 5445]
488 [904, 910, 1207, 8368, 3897, 1394, 1704, 1293, 678, 7153]
180 [1213, 1196, 1242, 2761, 2140, 3755, 2717]
27 [1247, 3836, 3114, 1222, 648, 1947, 1127, 1287, 1566, 920]
135 [2959, 1921, 1610, 457, 2804, 1387, 3360, 1198, 2692, 3020]
490 [1246, 48774, 46578, 56367, 3481, 48780, 4246, 1721, 87232, 6483
9]
13 [1198, 2571, 3793, 1173, 3893, 3952, 3863]
500 [2542, 2700, 101, 4306, 1282, 3083, 1476, 1914, 449, 595]
37 [318, 1061, 316, 380, 329]
30 [5952, 33794, 96821, 1210, 58559]
441 [58559, 106782, 68358, 1241, 104, 4402, 2683, 8917, 87529]
384 [1036, 457, 968, 3996, 3471, 2987, 2683, 3098, 4323, 4276]
436 [527, 356, 293, 161, 333, 2, 568, 536, 339, 471]
127 [4226, 910, 1276, 3671, 3039, 272, 3101]
545 [1267, 33794, 1876, 1438, 1911, 3273]
250 [1035, 4886, 1408, 2739, 2407, 3418, 45]
497 [2959, 7361, 58559, 79132, 68954, 2329, 1198, 608, 112290, 2]
```

```
401 [356, 112552, 68954, 60069, 7153, 59315, 162578, 4016, 134853, 5
6757]
376 [750, 1197, 5952, 1258, 4995, 260, 1214, 1196, 1240, 4896]
511 [4993, 79132, 91529, 115713, 97752, 59387, 105844, 112623, 9520
7]
503 [527, 1221, 2028, 1206, 1259, 1997, 147, 3421, 8042, 2114]
86 [7153, 5952, 356, 1265, 1210, 6539, 79132, 5418, 480, 4226]
444 [110, 58, 21, 509, 235, 185]
539 [4886, 2502, 8360, 89864, 6753, 1801, 3087, 3264]
321 [364, 356, 380, 21, 539, 592, 440, 165, 24, 208]
494 [1204, 110, 480, 344]
133 [150, 110, 555, 367]
582 [260, 593, 4993, 69844, 92259, 68954, 122886, 116797, 68358, 916
30]
554 [1213, 1225, 2396, 1035, 2804, 919, 1210, 899, 364, 594]
123 [1196, 79132, 109487, 96829, 87232, 64969, 84152, 116797, 48780,
112556]
398 [7361, 914, 60069, 31658, 1207, 1197, 7022, 261, 2144, 26631]
262 [318, 457, 593, 265, 428, 538, 36, 594, 164, 26]
329 [1201, 1292, 1080, 1258, 4789, 130482]
379 [364, 110, 480, 161, 185, 434]
327 [318, 1221, 1304, 2329, 1213, 3897, 58559, 150, 3515, 765]
78 [260, 2000, 318, 3869, 2792, 27904, 2795, 2605, 1359, 344]
540 [2571, 48516, 1210, 1073, 648, 589, 1971, 736, 2772]
189 [76093, 1265, 4993]
487 [2959, 79132, 109374, 81845, 50872, 4886, 56367, 49272, 8961, 86
377]
73 [1682, 4226, 89745, 3793, 93510, 91542, 96588, 91529, 53125, 1111
13]
83 [1199, 68954, 5418, 1204, 6016, 1280, 4022, 1348, 40278, 1282]
530 [356, 349, 539, 592, 300, 339, 586]
446 [296, 21, 377, 17, 539, 780, 329, 231, 168, 186]
481 [1183, 1721, 1687, 1662, 1465, 1407, 1665, 1438, 1690, 1614]
184 [4226, 122926, 112852, 164179, 3793, 80463, 39292, 122920, 17940
1, 122912]
253 [4973, 6016, 3083, 55620, 26524, 55069]
166 [1197, 3147, 6711, 1221, 7090, 1225, 5618, 608, 33166, 49272]
442 [1231, 616, 524, 4361, 1186]
40 [296, 111, 610, 199, 353, 431, 300, 371, 616, 281]
341 [356, 115713, 1036, 6863, 79134, 94677, 5459, 788]
493 [1172, 3578, 1291, 6, 1198, 1214, 3996, 2916, 493, 4085]
306 [5618, 56152, 80463, 2762, 96588, 81847, 72998, 4963, 106696, 16
8418]
289 [750, 2797, 6327, 3]
426 [2797, 81834, 4226, 356, 88810, 364, 296, 588, 5444, 8368]
142 [356, 34, 508, 380, 231, 434]
557 [2028, 5618, 1291, 2617]
172 [1215, 1953, 527, 2762, 661, 277]
283 [1500, 1393, 805, 500, 1517, 2082]
583 [1732, 4306, 7293, 2572, 106696, 594, 216, 1088, 1721, 99117]
163 [1721, 1805, 1821, 1485, 1837]
499 [2083, 6184, 53519]
69 [50, 4027, 3994, 589, 4880, 1580, 1721, 4876]
216 [2248, 2716, 1179, 1304, 3671, 1267, 34, 1292, 2239, 1238]
178 [2959, 1213, 356, 457, 44191, 1270, 4226, 589, 480, 8961]
417 [318, 296, 58559, 1732, 68157, 2858, 1213, 47099, 109487, 10678
```

```
2]
393 [589, 3578, 1198, 4995, 48394, 293, 112852, 86190, 91658, 37857]
447 [110, 589, 595, 592, 236, 2, 257, 368, 10, 316]
2 [109487, 3578, 79132, 74458, 46970, 131724]
67 [318, 2028, 6377, 99114, 60069, 6539, 5952, 5418, 5574, 2012]
257 [1288, 5445, 494, 7, 420]
170 [318, 457, 150, 368, 95, 208, 181]
427 [5014, 5015, 5349, 5254, 5312, 5325, 5816, 5127, 6238, 5108]
24 [2028, 139385, 1704, 96079, 134130, 68358, 6, 122886, 5064, 5297
3]
270 [6, 1356, 494, 141, 52, 376, 5, 784, 788]
346 [1617, 2160, 1704, 1219, 111, 48394, 492, 3535, 3186, 1968]
243 [527, 36, 648, 736, 592, 145, 434, 442, 172, 466]
151 [107, 747, 609, 828, 786, 802, 634, 1359, 765, 784]
388 [6377, 745, 8360, 104245, 95105, 78637, 631, 33615, 44022, 575]
609 [318, 110, 161, 480, 1161, 613, 339, 253, 137, 116]
399 [1291, 1270, 3114, 1]
521 [6, 32, 25, 837, 802, 81, 637, 736, 880, 640]
411 [50, 356, 457, 261, 590, 150, 11, 296, 529, 534]
516 [1197, 1958, 4337, 4333]
443 [296, 5952, 608, 48394, 56174, 106487, 78574, 97921, 103249, 108
190]
92 [2398, 69757, 51662, 35836, 2114, 55282]
128 [1198, 923, 1284, 945]
61 [1208, 7361, 1258, 16, 293, 27831]
215 [1247, 4878, 1259, 2692, 2542, 296, 4993, 2918, 593, 1968]
526 [2692, 2028, 308, 1246, 4027, 3168, 109374, 60950, 151455, 4014]
295 [318, 1221, 1089, 50, 1201, 44555, 79132, 1466, 1375, 3755]
378 [2959, 71535, 74458, 4226, 593, 858, 44191, 68358, 8798, 36]
60 [318, 2067, 3424, 362, 805]
39 [1276, 1215, 745, 1197, 1198, 1954, 1248, 2912, 589, 2078]
458 [364, 27, 165, 292, 367, 222, 361, 48]
459 [68157, 74458, 97913, 72998]
90 [1411, 25, 116, 85]
588 [527, 457, 296, 589, 364, 147, 161, 733, 198, 300]
267 [1653, 1200, 3471, 3527, 2890, 3994, 2861, 3113, 1909, 3101]
585 [296, 4011, 3362, 2502, 2858, 2076, 4239, 3424, 7445, 4571]
229 [527, 474, 348, 480, 345, 508, 539, 380, 165, 10]
407 [293, 1291, 76093, 4993]
396 [318, 6377, 1148, 1674, 5445, 783]
533 [1200, 79132, 6502, 356, 119145, 3727, 57274, 54995, 80831]
528 [48516, 593, 778, 2571, 6377, 80906, 4886, 59315, 49530, 2858]
35 [595, 261, 235, 237, 185]
225 [1079, 5060, 1270, 3039, 3052, 1753, 588, 2003, 1333, 785]
349 [593, 527, 337, 590, 595, 353, 587, 367, 105, 434]
598 [5418, 103543]
235 [296, 318, 589, 364, 592, 349, 539, 500]
176 [553, 377]
165 [858, 1291, 5993, 5218, 5299, 1573, 5450, 2422]
188 [1244, 913, 1950, 906, 2795, 2243, 3097]
207 [2858, 2021, 2949, 3263, 100, 743]
491 [4896, 2716, 64969, 104, 95441, 45720, 97921, 104211, 4306, 367]
400 [50, 296, 48516, 1258, 1036, 134130, 293, 59615]
102 [296, 593, 454, 21, 350, 539, 173, 172, 186]
152 [541, 27773, 6016, 1207, 57669, 80489, 89864, 91630, 61323, 8577
4]
```

```
44 [1393, 1639, 1476, 1517, 889, 805, 1429, 661, 667, 1552]
236 [1278, 1982, 2580, 2596, 2459, 2718, 3024, 2657, 1345, 2447]
97 [593, 4993, 4963, 1210, 4161, 5420]
22 [2959, 68157, 7153, 44191, 3578, 5902, 49272, 50872, 7438, 56782]
261 [296, 58559, 1222, 1210, 96079, 4878, 56367, 80463, 1625, 8914]
206 [58, 1356, 780, 95, 786]
550 [58559, 60069, 79132, 134130, 93510, 80549, 88140]
371 [6711, 81834, 6283, 924, 1274, 2810, 6365, 592, 48, 1377]
164 [589, 1196, 1210, 1240, 3793, 3639, 1580, 4440, 1047]
31 [1393, 1356, 1361, 1302, 733, 141, 780, 1363, 852, 344]
218 [2502, 471, 1372, 1517, 1380, 172]
300 [527, 4973, 593, 79132, 4848, 112183, 112556]
512 [150, 590, 151, 377, 165, 539, 597, 288, 160, 442]
568 [296, 2863, 147]
71 [608, 1356, 260, 1036, 780, 707]
449 [4226, 2395, 2012, 2011, 2770]
154 [79132, 109487, 76093, 110102, 72998, 1580, 96610, 122892, 13063
4]
53 [922, 2686, 1441, 2616]
394 [457, 150, 225, 339, 553, 208]
285 [1193, 1104, 1219, 1103, 5995, 3481, 113207, 3262, 4091]
467 [2324, 527, 1408]
472 [858, 4993, 1892, 765]
435 [48780, 33794, 524, 3510, 48]
224 [858, 969, 953, 1307, 1265, 1393, 3552, 3210, 2915, 2918]
296 [50, 356, 318, 180031, 160848, 169034]
173 [296, 587, 410]
536 [593, 110, 592, 454, 500, 370, 419]
231 [5952, 92259, 1240, 54286, 63082]
532 [2571, 2329, 1304, 527, 290, 1610, 4086, 4654]
259 [1584, 509, 2683]
501 [2, 709, 73, 95, 135]
281 [46578, 35836, 2023]
360 [1639, 1721, 1754, 1646, 1616, 1438]
338 [1221, 296, 170705, 122916, 30749, 187541, 175569, 190207]
291 [89745, 60069, 68954, 4896, 5218, 54001, 102125]
258 [527, 2762, 47099, 122886]
538 [81845, 3671, 68954, 902, 595, 44889]
355 [1721, 915, 367, 673]
155 [3052, 1923, 2948, 2355, 3555, 2724, 1377, 1911]
363 [318, 1198, 1784, 176389]
194 [1035, 2406, 48, 168, 2699]
374 [1258, 457, 1333, 780, 1345]
55 [48516, 2005, 1293, 2393, 2100]
302 [32, 47, 25, 707, 786]
14 [296, 593, 282, 39, 351, 344]
175 [2300, 3969, 45880, 3791, 33880, 32289, 2526, 43904]
72 [1962, 150, 27193, 780, 1093, 2634, 344]
126 [356, 364, 589, 377, 590, 165, 292, 34]
324 [1197, 3915, 3864]
46 [318, 253, 380, 282, 432, 435]
208 [2028, 1610, 589, 380, 2023, 2881]
25 [2571, 527, 116797, 74458, 3578, 68157, 4993, 59315]
99 [349, 22, 225, 145, 122, 315, 160, 434]
121 [337, 16, 272, 780, 141, 377, 282, 237, 39, 432]
70 [318, 1682, 953, 1954, 1304, 508]
```

```
276 [609, 653, 344, 88, 673, 810, 19]
79 [2571, 1242, 2944, 1374, 1345]
515 [3147, 58559, 318, 91529, 175569]
293 [1682, 1, 1917, 344, 3646, 2411]
518 [3418, 1747, 31, 1371, 1597, 24, 1876]
595 [1278, 1711, 2770]
192 [150, 349, 225, 316, 288]
485 [253, 165, 231]
455 [356, 150, 11, 252, 539, 454, 292, 367, 282]
406 [531, 44840]
3 [4518, 2018, 6835, 2851, 7991, 2424]
320 [6283]
364 [141, 733]
120 [260, 852, 12]
118 [1104, 1343]
272 [1682, 122922, 50872]
245 [2289, 581, 1092]
549 [318, 5952, 1196, 293]
77 [109487, 5418, 4878, 7438, 4226]
519 [56367, 1968, 72998, 69122]
581 [112552, 134130, 4886, 4896, 92259, 44191]
383 [1246, 150, 1234, 1953, 1994, 2447]
26 [454, 153]
576 [2289, 1345, 1321, 372, 2641]
81 [356, 316]
134 [364, 161, 367, 273, 227, 173]
423 [296, 318, 150, 653]
205 [96079]
508 [1222, 1175, 799, 1378]
114 [54001, 40815, 95510]
340 [318, 288, 344]
130 [225]
```

Our RMSE being 1.8 means that we are off, on average, by 1.8 points.

# Getting recomendations for a normal user

As we already have user recomendations, we will be using suprise for recomendations, and checking our work based on how similar the moves are with our distance metrics.

In [52]:
```
target_user = ratings.sample(random_state=42)['userId'].iloc[0]
target_user
```

Out[52]: 432

In [47]:
```
#take a look at the user's preferences
target_user_df = ratings.loc[ratings['userId']==target_user]
target_user_df = target_user_df.sort_values(by=['rating'],ascending
=False)
target_user_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 260 entries, 66909 to 66950
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   userId     260 non-null    int64
 1   movieId    260 non-null    int64
 2   rating     260 non-null    float64
 3   timestamp  260 non-null    int64
dtypes: float64(1), int64(3)
memory usage: 10.2 KB
```

In [48]:
```
#only use movies we can name
target_user_df['title'] = target_user_df['movieId'].map(get_movie_n
ame)
target_user_df['genres'] = target_user_df['movieId'].map(get_movie_
genres)

target_user_df = target_user_df[target_user_df.title != 'Movie not
found']
```

In [49]:
```
#from the larger dataset I know their least favorite is The Ape 0.5
target_user_df
```

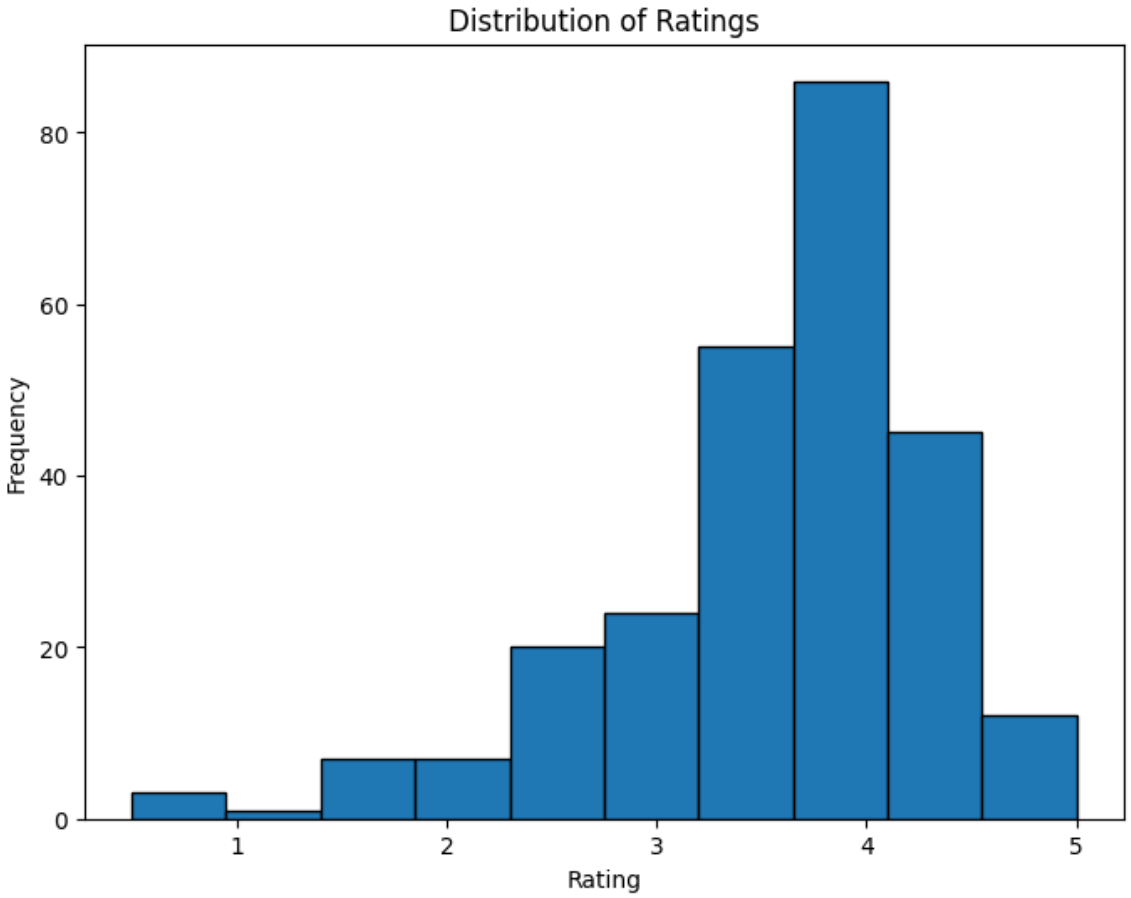
Out[49]:

| | userId | movieId | rating | timestamp | title | genres |
|---|---|---|---|---|---|---|
| **66909** | 432 | 3949 | 5.0 | 1315242940 | Requiem for a Dream | [Drama] |
| **67045** | 432 | 81591 | 5.0 | 1315244863 | Black Swan | [Drama, Thriller] |
| **67029** | 432 | 71379 | 5.0 | 1316391330 | Paranormal Activity | [Horror, Thriller] |
| **66971** | 432 | 8533 | 5.0 | 1315244178 | Notebook, The | [Drama, Romance] |
| **66864** | 432 | 1835 | 5.0 | 1315244490 | City of Angels | [Drama, Fantasy, Romance] |
| **...** | ... | ... | ... | ... | ... | ... |
| **66957** | 432 | 6947 | 1.5 | 1335139655 | Master and Commander: The Far Side of the World | [Adventure, Drama, War] |
| **66876** | 432 | 2278 | 1.0 | 1315242196 | Ronin | [Action, Crime, Thriller] |
| **66886** | 432 | 2683 | 0.5 | 1316391662 | Austin Powers: The Spy Who Shagged Me | [Action, Adventure, Comedy] |
| **66849** | 432 | 1293 | 0.5 | 1315242191 | Gandhi | [Drama] |
| **66950** | 432 | 6539 | 0.5 | 1316391748 | Pirates of the Caribbean: The Curse of the Bla... | [Action, Adventure, Comedy, Fantasy] |

260 rows × 6 columns

In [54]:
```
# @title Distribution of Ratings

plt.figure(figsize=(8, 6))
plt.hist(target_user_df['rating'], bins=10, edgecolor='black')
plt.xlabel('Rating')
plt.ylabel('Frequency')
_ = plt.title('Distribution of Ratings')
```



In [58]:
```
top_10_user = target_user_df[:10]
top_10_user
```

Out[58]:

| | userId | movieId | rating | timestamp | title | genres |
|---|---|---|---|---|---|---|
| **66909** | 432 | 3949 | 5.0 | 1315242940 | Requiem for a Dream | [Drama] |
| **67045** | 432 | 81591 | 5.0 | 1315244863 | Black Swan | [Drama, Thriller] |
| **67029** | 432 | 71379 | 5.0 | 1316391330 | Paranormal Activity | [Horror, Thriller] |
| **66971** | 432 | 8533 | 5.0 | 1315244178 | Notebook, The | [Drama, Romance] |
| **66864** | 432 | 1835 | 5.0 | 1315244490 | City of Angels | [Drama, Fantasy, Romance] |
| **66865** | 432 | 1873 | 5.0 | 1316391161 | Misérables, Les | [Crime, Drama, Romance, War] |
| **66871** | 432 | 1997 | 5.0 | 1316391091 | Exorcist, The | [Horror, Mystery] |
| **66818** | 432 | 364 | 5.0 | 1316391519 | Lion King, The | [Adventure, Animation, Children, Drama, Musica... |
| **66992** | 432 | 45503 | 5.0 | 1316391388 | Peaceful Warrior | [Drama] |
| **67007** | 432 | 57274 | 5.0 | 1316391360 | [REC] | [Drama, Horror, Thriller] |

In [61]:
```
bottom_10_user = target_user_df[-10:]
bottom_10_user
```

Out[61]:

| | userId | movieId | rating | timestamp | title | genres |
|---|---|---|---|---|---|---|
| **66834** | 432 | 780 | 1.5 | 1316391462 | Independence Day (a.k.a. ID4) | [Action, Adventure, Sci-Fi, Thriller] |
| **66901** | 432 | 3273 | 1.5 | 1315242328 | Scream 3 | [Comedy, Horror, Mystery, Thriller] |
| **66816** | 432 | 344 | 1.5 | 1316391506 | Ace Ventura: Pet Detective | [Comedy] |
| **66982** | 432 | 33158 | 1.5 | 1315242713 | xXx: State of the Union | [Action, Crime, Thriller] |
| **66934** | 432 | 5507 | 1.5 | 1315242710 | xXx | [Action, Crime, Thriller] |
| **66957** | 432 | 6947 | 1.5 | 1335139655 | Master and Commander: The Far Side of the World | [Adventure, Drama, War] |
| **66876** | 432 | 2278 | 1.0 | 1315242196 | Ronin | [Action, Crime, Thriller] |
| **66886** | 432 | 2683 | 0.5 | 1316391662 | Austin Powers: The Spy Who Shagged Me | [Action, Adventure, Comedy] |
| **66849** | 432 | 1293 | 0.5 | 1315242191 | Gandhi | [Drama] |
| **66950** | 432 | 6539 | 0.5 | 1316391748 | Pirates of the Caribbean: The Curse of the Bla... | [Action, Adventure, Comedy, Fantasy] |

In [50]:
```
top_n[target_user]
```

Out[50]:
```
[(912, 4.226976513678512),
 (5952, 4.2228780816318086),
 (4886, 4.178277744210748),
 (4973, 4.177529351747113),
 (4306, 4.152281623746379),
 (7153, 4.068506270452766),
 (3949, 4.044446968440032),
 (3147, 4.019193875263596),
 (6947, 4.010385103326618),
 (69481, 3.9677129843717935)]
```

In [51]:
```
for n in top_n[target_user]:
    print(get_movie_name(n[0]))
```

```
Casablanca
Lord of the Rings: The Two Towers, The
Monsters, Inc.
Amelie (Fabuleux destin d'Amélie Poulain, Le)
Shrek
Lord of the Rings: The Return of the King, The
Requiem for a Dream
Green Mile, The
Master and Commander: The Far Side of the World
Hurt Locker, The
```