Opis opracowania

Celem opracowania jest analiza danych dotyczących wszystkich dotychczas wyemitowanych produkcji z uniwersum Scooby - Doo. Projekt składa się z analizy podstawowych wskaźników statystycznych oraz badania zależności pomiędzy poszczególnymi parametrami. Szczególna uwaga jest przykładana do oceny odcinka oraz zaangażowania społeczności w zależności od innych parametrów, ma to na celu uzyskanie informacji co czyni poszczególne produkcje atrakcyjnymi dla widza. Oprócz tego badane jest także kilka innych zależności.

```
import csv, sqlite3
import pandas as pd
import seaborn as sb
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as ss
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
import datetime as dt
```

Wczytanie danych, zapisanie do bazy SQLite

Zmiana nazwy kolumn w celu uniknięcia problemów z niedozwolonymi znakami w nazwach kolumn

```
df = pd.read_csv('scooby-doo.csv')

df.columns = df.columns.str.replace('[./]', '_')

df.to_csv('scooby-doo.csv', index=False)

conn = sqlite3.connect('scooby-doo.db')
c = conn.cursor()

df.to_sql('data', conn, if_exists = 'append', index = False)
```

```
615
```

```
df
```

```
      id                  series_name    network season  \
0      1  Scooby Doo, Where Are You!         CBS      1
1      2  Scooby Doo, Where Are You!         CBS      1
2      3  Scooby Doo, Where Are You!         CBS      1
3      4  Scooby Doo, Where Are You!         CBS      1
4      5  Scooby Doo, Where Are You!         CBS      1
..   ...                         ...         ...    ...
```

```
610  611   Scooby-Doo and Guess Who?  Boomerang      2
611  612   Scooby-Doo and Guess Who?  Boomerang      2
612  613   Scooby-Doo and Guess Who?  Boomerang      2
613  614   Scooby-Doo and Guess Who?  Boomerang      2
614  615   Scooby-Doo and Guess Who?  Boomerang      2

                                  title  imdb  engagement  date_aired
\
0             What a Night for a Knight   8.1       556.0   9/13/1969

1                 A Clue for Scooby Doo   8.1       479.0   9/20/1969

2                  Hassle in the Castle   8.0       455.0   9/27/1969

3                Mine Your Own Business   7.8       426.0   10/4/1969

4                 Decoy for a Dognapper   7.5       391.0  10/11/1969

..                                  ...   ...         ...         ...

610       A Haunt of a Thousand Voices!   8.4        67.0   10/1/2021

611             The Movieland Monsters!   7.0        55.0   10/1/2021

612             Scooby-Doo, Dog Wonder!   6.4        51.0   10/1/2021

613  The Legend of the Gold Microphone!   6.7        45.0   10/1/2021

614         Lost Soles of Jungle River!   7.1        61.0   10/1/2021


     run_time     format  ... batman scooby-dum scrappy-doo hex_girls
\
0          21  TV Series  ...  False      False       False     False

1          22  TV Series  ...  False      False       False     False

2          21  TV Series  ...  False      False       False     False

3          21  TV Series  ...  False      False       False     False

4          21  TV Series  ...  False      False       False     False

..        ...        ...  ...    ...        ...         ...       ...

610        23  TV Series  ...  False      False       False     False

611        22  TV Series  ...  False      False       False     False

612        23  TV Series  ...  False      False       False     False
```

```
613        22  TV Series  ...  False      False       False     False

614        22  TV Series  ...  False      False       False     False


    blue_falcon       fred_va                    daphne_va
velma_va  \
0         False  Frank Welker  Stefanianna Christopherson  Nicole
Jaffe
1         False  Frank Welker  Stefanianna Christopherson  Nicole
Jaffe
2         False  Frank Welker  Stefanianna Christopherson  Nicole
Jaffe
3         False  Frank Welker  Stefanianna Christopherson  Nicole
Jaffe
4         False  Frank Welker  Stefanianna Christopherson  Nicole
Jaffe
..          ...           ...                         ...           ..
.
610       False  Frank Welker               Grey DeLisle  Kate
Micucci
611       False  Frank Welker               Grey DeLisle  Kate
Micucci
612        True  Frank Welker               Grey DeLisle  Kate
Micucci
613       False  Frank Welker               Grey DeLisle  Kate
Micucci
614       False  Frank Welker               Grey DeLisle  Kate
Micucci

           shaggy_va     scooby_va
0        Casey Kasem   Don Messick
1        Casey Kasem   Don Messick
2        Casey Kasem   Don Messick
3        Casey Kasem   Don Messick
4        Casey Kasem   Don Messick
..               ...           ...
610  Matthew Lillard  Frank Welker
611  Matthew Lillard  Frank Welker
612  Matthew Lillard  Frank Welker
613  Matthew Lillard  Frank Welker
614  Matthew Lillard  Frank Welker

[615 rows x 75 columns]
```

Dodawana jest także dodatkowa kolumna 'catchphrases', która zawiera informację o łącznej ilości zwrotów typowych dla bohaterów

```
alter_query = 'alter table data add column catchphrases INTEGER'
c.execute(alter_query)

update_query = 'update data set catchphrases = ifnull(if_it_wasnt_for,
0) + ifnull(and_that, 0) + ifnull(split_up, 0) +
ifnull(another_mystery, 0) + ifnull(set_a_trap, 0) + ifnull(jeepers,
0) + ifnull(jinkies, 0) + ifnull(my_glasses, 0) +
ifnull(just_about_wrapped_up, 0) + ifnull(zoinks, 0) + ifnull(groovy,
0) + ifnull(scooby_doo_where_are_you, 0)'
c.execute(update_query)

conn.commit()

data = pd.read_sql_query('select * from data', conn)
```

Czyszczenie danych

Uzupełnienie wierszy dla odcinków podzielonych na segmenty (takie rekordy mają osobną ocenę, jednak dla tych rekordów informacje o potworze są zawarte w rekordzie, który opisuje pierwszy odcinek z danej grupy)

```
cols = data.columns.difference(['if_it_wasnt_for', 'and_that'])
data[cols] = data[cols].ffill()

data["date_aired"] = pd.to_datetime(data["date_aired"])
```

Istnieje kilka produkcji, które odbiły się głośnym echem wśród odbiorców i które zgromadziły (w porównaniu do pozostałych produkcji) ogromną ilość ocen na portalu imbd.com. Ze względu na skalę różnicy w wartościach zmiennej 'engagement' dla tych produkcji, analiza oraz obliczenia byłyby obarczone sporym błędem. W celu uniknięcia takiej sytuacji, dane odstające zostaną odrzucone. Ograniczenie zmiennej 'engagement' do wartości 10000 powoduje usunięcie 4 rekordów (około 0.5% danych)

```
data['engagement'].describe()
```

```
count       615.000000
mean        557.578862
std        4702.332303
min           7.000000
25%          26.500000
50%          52.000000
75%         122.000000
max      100951.000000
Name: engagement, dtype: float64
```

```
outliers = pd.read_sql_query('select * from data where engagement >
5000', conn)
outliers
```

```
    id        series_name            network     season  \
0  337  Warner Home Video  Warner Home Video      Movie
1  339  Warner Home Video  Warner Home Video      Movie
2  340  Warner Home Video  Warner Home Video      Movie
3  422  Warner Home Video    Cartoon Network      Movie
4  553       Supernatural             The CW  Crossover

                               title  imdb  engagement date_aired
run_time  \
0   Scooby-Doo and the Witch's Ghost   7.3      6527.0  10/5/1999
66
1  Scooby-Doo and the Alien Invaders   6.9      5625.0  10/3/2000
73
2     Scooby-Doo and the Cyber Chase   7.0      6632.0  10/9/2001
73
3     Scooby-Doo! The Mystery Begins   5.3      5805.0  9/13/2009
82
4                       Scoobynatural   9.6      6929.0  3/29/2018
42

      format  ... scooby-dum scrappy-doo hex_girls blue_falcon
fred_va  \
0      Movie  ...          0           0         1           0  Frank
Welker
1      Movie  ...          0           0         0           0  Frank
Welker
2      Movie  ...          0           0         0           0  Frank
Welker
3      Movie  ...          0           0         0           0  Robbie
Amell
4  Crossover  ...          0           0         0           0  Frank
Welker

           daphne_va       velma_va        shaggy_va     scooby_va  \
0  Marry Kay Bergman      B.J. Ward      Scott Innes   Scott Innes
1  Marry Kay Bergman      B.J. Ward      Scott Innes   Scott Innes
2       Grey DeLisle      B.J. Ward      Scott Innes   Scott Innes
3        Kate Melton  Hayley Kiyoko     Nick Palatas  Frank Welker
4       Grey DeLisle   Kate Micucci  Matthew Lillard  Frank Welker

   catchphrases
0            13
1            22
2            21
3            16
4             7

[5 rows x 76 columns]
```

```
data = data[data['engagement'] < 5000]
```

Zapisanie danych do nowej tabeli w bazie SQLite

```
data.to_sql('preproccessedData', conn, if_exists='append',
index=False)
```

```
611
```

Podstawowe statystyki opisowe

Wartość średnia dla zmiennych numerycznych

```
means = data.mean(numeric_only=True)
means
```

```
id                          307.358429
imdb                          7.285434
engagement                  259.029460
run_time                     23.173486
monster_real                  0.248773
monster_amount                1.715221
caught_fred                   0.266776
caught_daphne                 0.045827
caught_velma                  0.070376
caught_shaggy                 0.135843
caught_scooby                 0.276596
captured_fred                 0.127660
captured_daphne               0.153846
captured_velma                0.132570
captured_shaggy               0.147300
captured_scooby               0.140753
unmask_fred                   0.193126
unmask_daphne                 0.062193
unmask_velma                  0.193126
unmask_shaggy                 0.027823
unmask_scooby                 0.044190
snack_fred                    0.029460
snack_daphne                  0.117840
snack_velma                   0.058920
snack_shaggy                  0.073650
snack_scooby                  0.021277
unmask_other                  0.070376
caught_other                  0.142390
caught_not                    0.050736
trap_work_first               0.455738
suspects_amount               2.837971
non-suspect                   0.101473
arrested                      0.797054
culprit_amount                1.049100
```

```
door_gag                    0.104746
split_up                    0.292962
another_mystery             0.140753
set_a_trap                  0.117840
jeepers                     0.707038
jinkies                     1.101473
my_glasses                  0.072013
just_about_wrapped_up       0.045827
zoinks                      2.261866
groovy                      0.049100
scooby_doo_where_are_you    0.139116
rooby_rooby_roo             0.711948
batman                      0.006547
scooby-dum                  0.027823
scrappy-doo                 0.268412
hex_girls                   0.009820
blue_falcon                 0.054010
catchphrases                3.929624
dtype: float64
```

Moda dla wartości zmiennych nieliczbowych

```
modes = data.select_dtypes(exclude=['number']).mode().iloc[0]
modes
```

```
series_name               Scooby-Doo and Scrappy-Doo (second series)
network                                                          ABC
season                                                             1
title                                                Wrestle Maniacs
date_aired                                       2020-07-02 00:00:00
format                                                     TV Series
monster_name                                                   Alien
monster_gender                                                  Male
monster_type                                                   Ghost
monster_subtype                                             Humanoid
monster_species                                                Human
setting_terrain                                                Urban
setting_country/state                                  United States
culprit_name                          Bank Robber,Captain Clements
culprit_gender                                                  Male
motive                                                   Competition
if_it_wasnt_for                                    you meddling kids
and_that                                                       puppy
number_of_snacks                                                   0
fred_va                                                  Frank Welker
daphne_va                                               Heather North
velma_va                                                  Marla Scott
shaggy_va                                                 Casey Kasem
```

```
scooby_va                                                    Don Messick
Name: 0, dtype: object
```

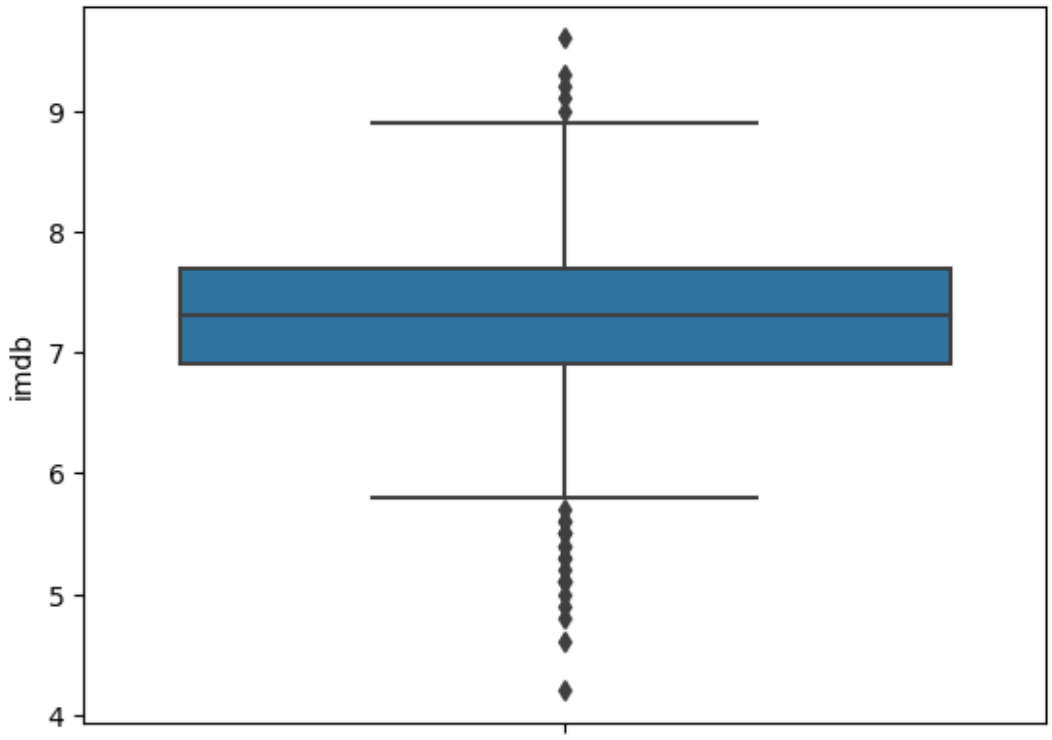
IMDB - podstawowe statystyki opisowe

```
data['imdb'].describe()
```

```
count    611.000000
mean       7.285434
std        0.717875
min        4.200000
25%        6.900000
50%        7.300000
75%        7.700000
max        9.600000
Name: imdb, dtype: float64
```

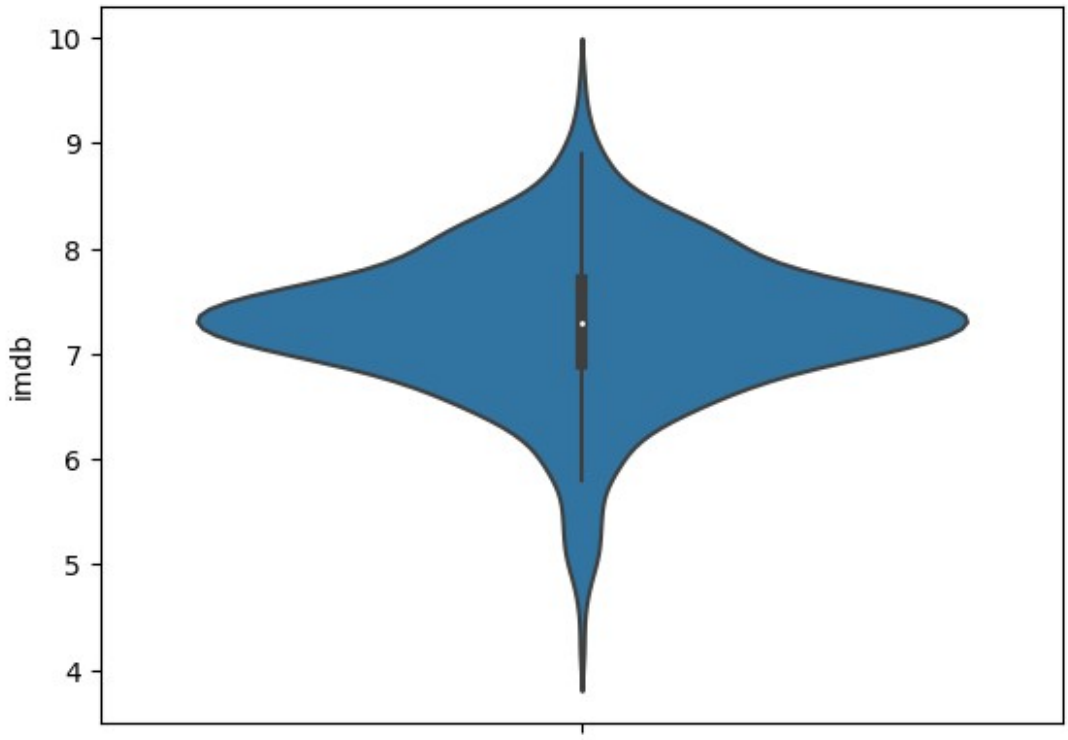
```
sb.boxplot(y = 'imdb', data = data)
```

```
<Axes: ylabel='imdb'>
```

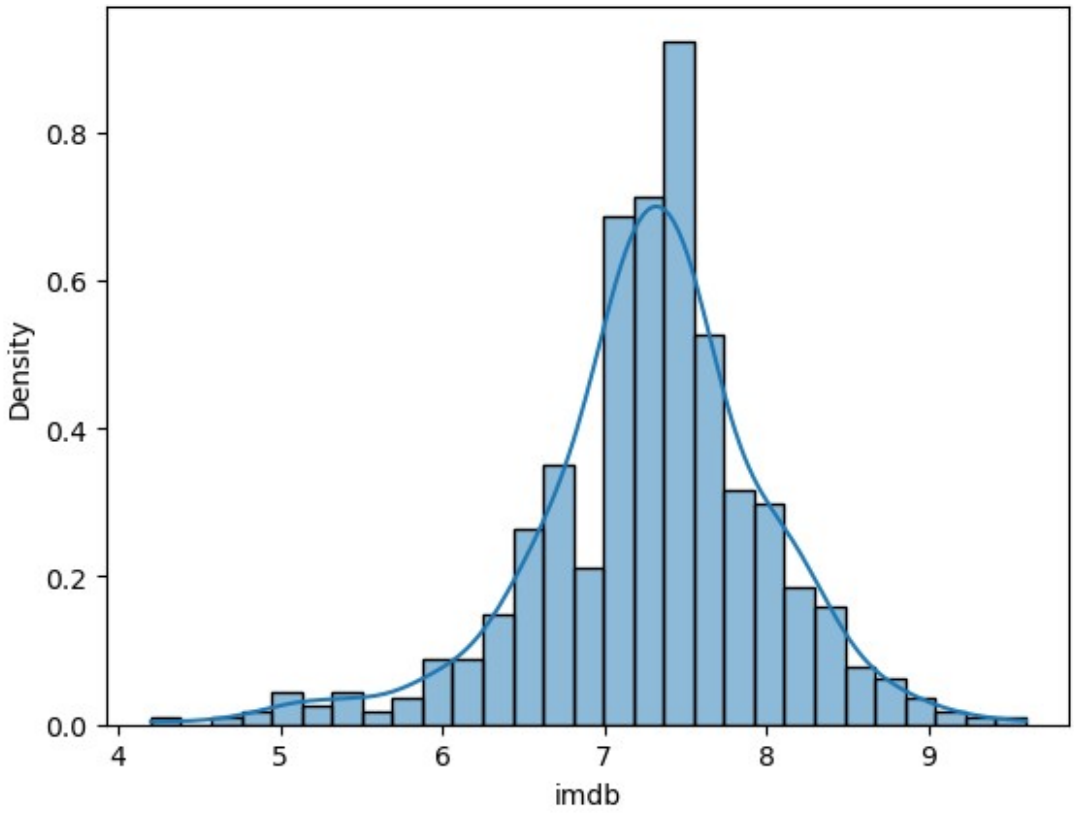



```
sb.violinplot(y = 'imdb', data = data)
```

```
<Axes: ylabel='imdb'>
```




```
ax = sb.histplot(data, x = 'imdb', kde = True, stat = 'density', label
= 'imdb')
```



Engagement - podstawowe statystyki opisowe

```
data['engagement'].describe()
```

```
count     611.000000
mean      259.029460
std       791.399762
min         7.000000
25%        26.000000
50%        52.000000
75%       117.000000
max      6929.000000
Name: engagement, dtype: float64
```

Ppnad 90% danych posiada parametr 'engagement' z przedziału [0,500]. W celu wyeliminowania wartości skrajnych oraz zwiększenia dokładności analizy rozpatrujemy wyłącznie dane z tego przedziału.

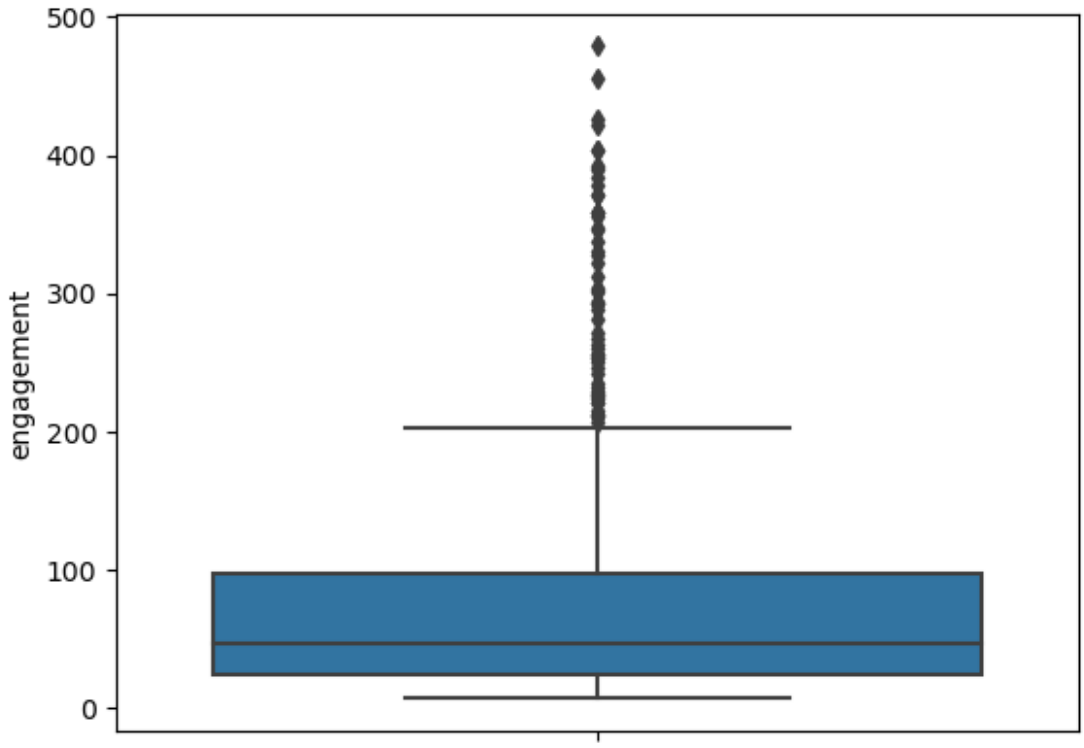
```
data[data['engagement'] < 500]['engagement'].describe()
```

```
count    568.000000
mean      83.045775
std       88.936464
min        7.000000
```

```
25%       25.000000
50%       47.000000
75%       97.250000
max      479.000000
Name: engagement, dtype: float64
```

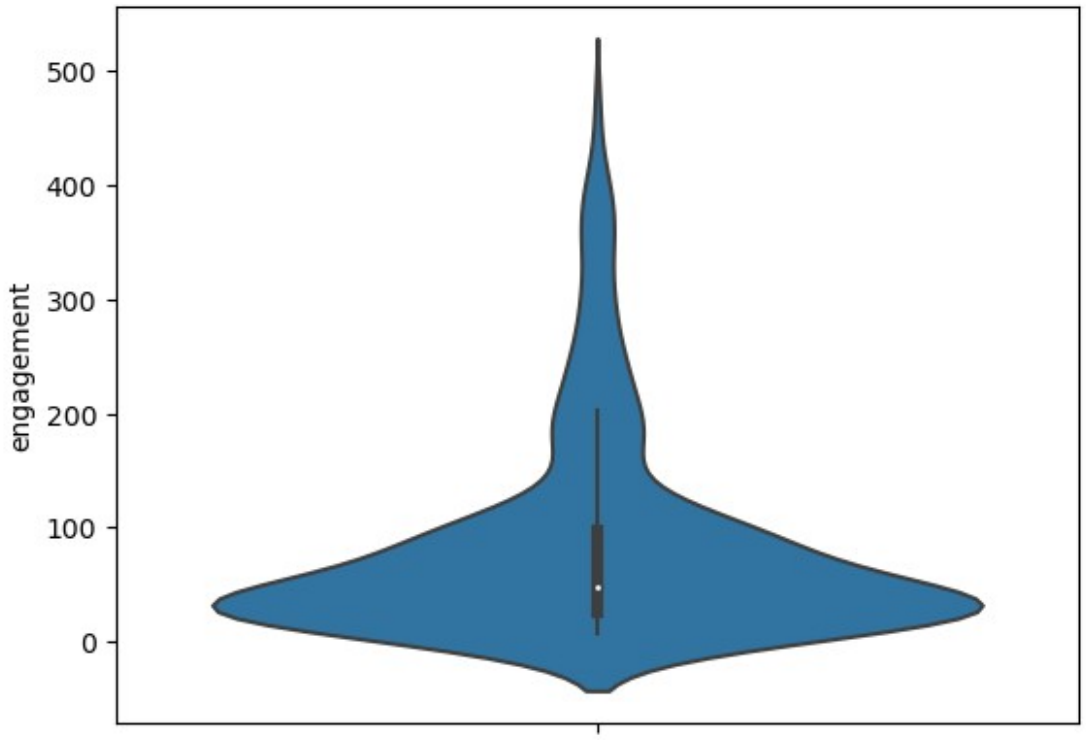
```
sb.boxplot(y = 'engagement', data = data[data['engagement'] < 500])
```

```
<Axes: ylabel='engagement'>
```



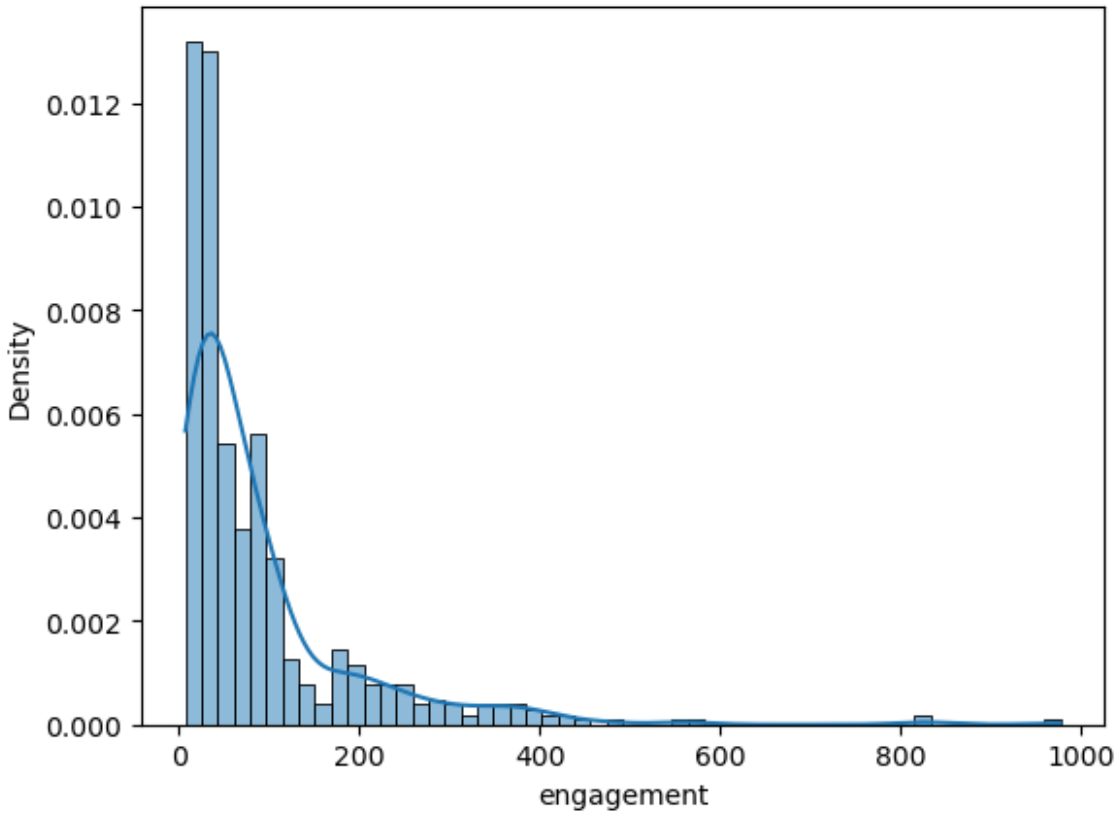

```
sb.violinplot(y = 'engagement', data = data[data['engagement'] < 500])
```

```
<Axes: ylabel='engagement'>
```




```
sb.histplot(data[data['engagement'] < 1000], x = 'engagement', kde =
True, stat = 'density', label = 'engagement')
```

```
<Axes: xlabel='engagement', ylabel='Density'>
```



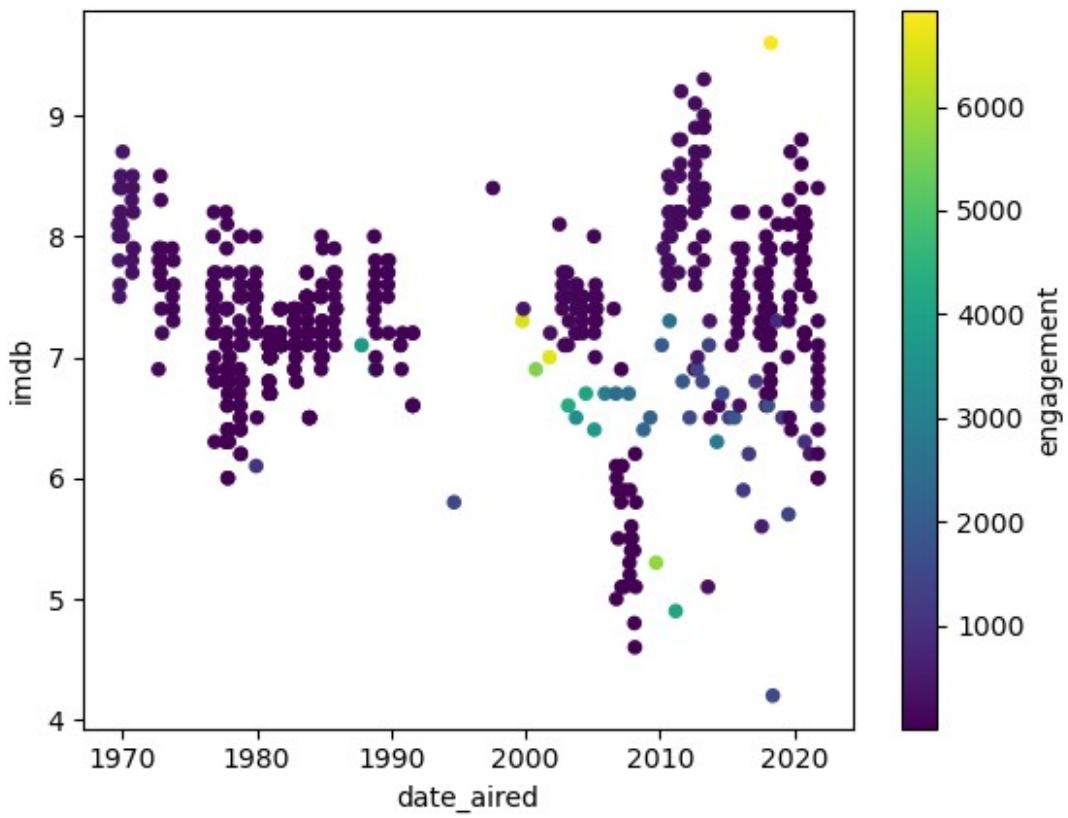
Zależność zmiennych

W celu pełnego opisu zbioru danych konieczne jest zbadanie zależności pomiędzy zmiennymi. W przypadku zmiennych ilościowych zostanie zrealizowane obliczając macierz korelacji, w przypadku zależności oceny i zaangażowania od zmiennych opisowych, konieczna jest analiza każdej zmiennej z osobna, ponieważ nie ma możliwości uporządkowania pewnych zmiennych.

Ocena oraz zaanagażowanie społeczności w czasie

```
data.plot.scatter(x = 'date_aired', y = 'imdb', c = 'engagement')
```

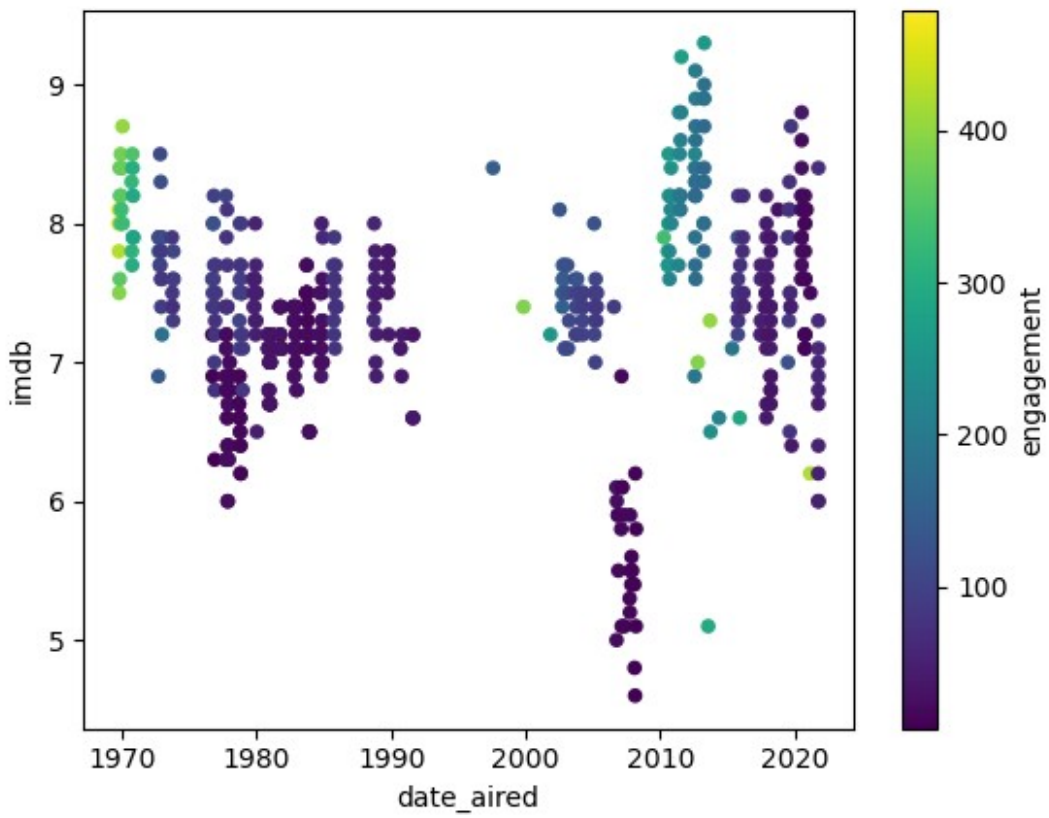
```
<Axes: xlabel='date_aired', ylabel='imdb'>
```



Ponieważ w zbiorze danych istnieje kilka wartości odstających w kolumnie engagement, powinniśmy również zbadać zaangażowanie społeczności po odrzuceniu wartości skrajnych, gdyż zaburzają one dokładność wykresu i analizy

```
no_engagement_outliers_over_500 = data[data['engagement'] < 500]
no_engagement_outliers_over_500.plot.scatter(x = 'date_aired', y =
'imdb', c = 'engagement')
```

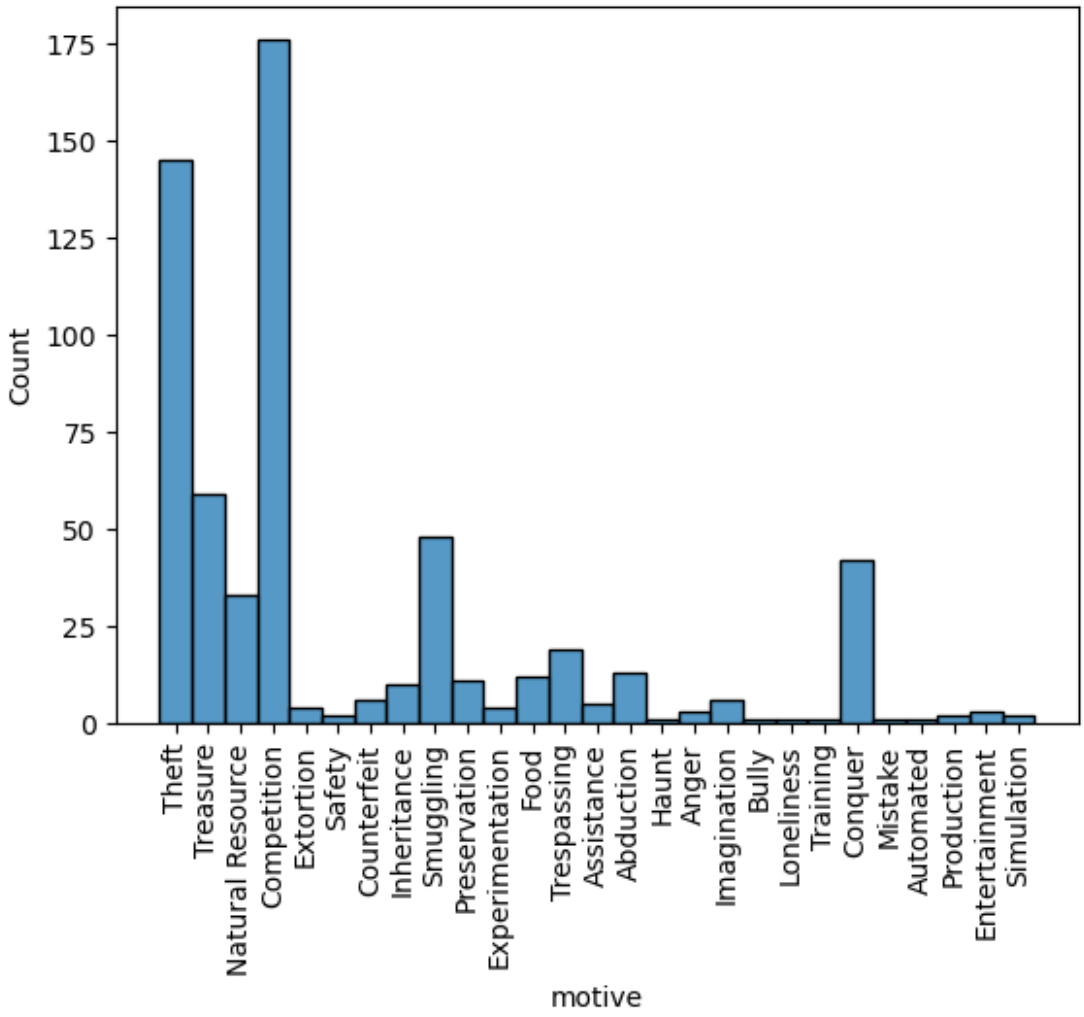
```
<Axes: xlabel='date_aired', ylabel='imdb'>
```



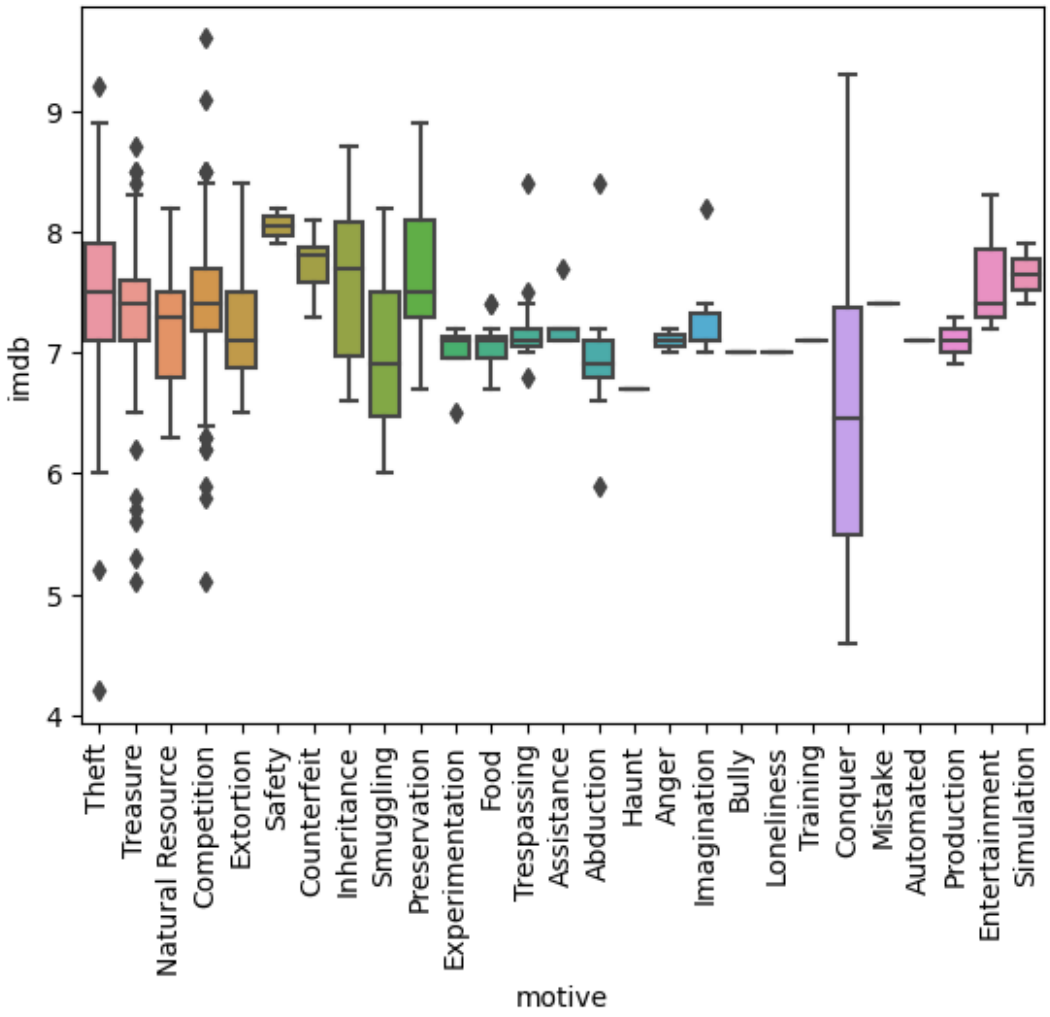
Ocena oraz zaangażowanie w zależności od motywu

```
motive_plot = sb.histplot(data, x = 'motive', label = 'motive')
motive_plot =
motive_plot.set_xticklabels(motive_plot.get_xticklabels(), rotation =
90)
```

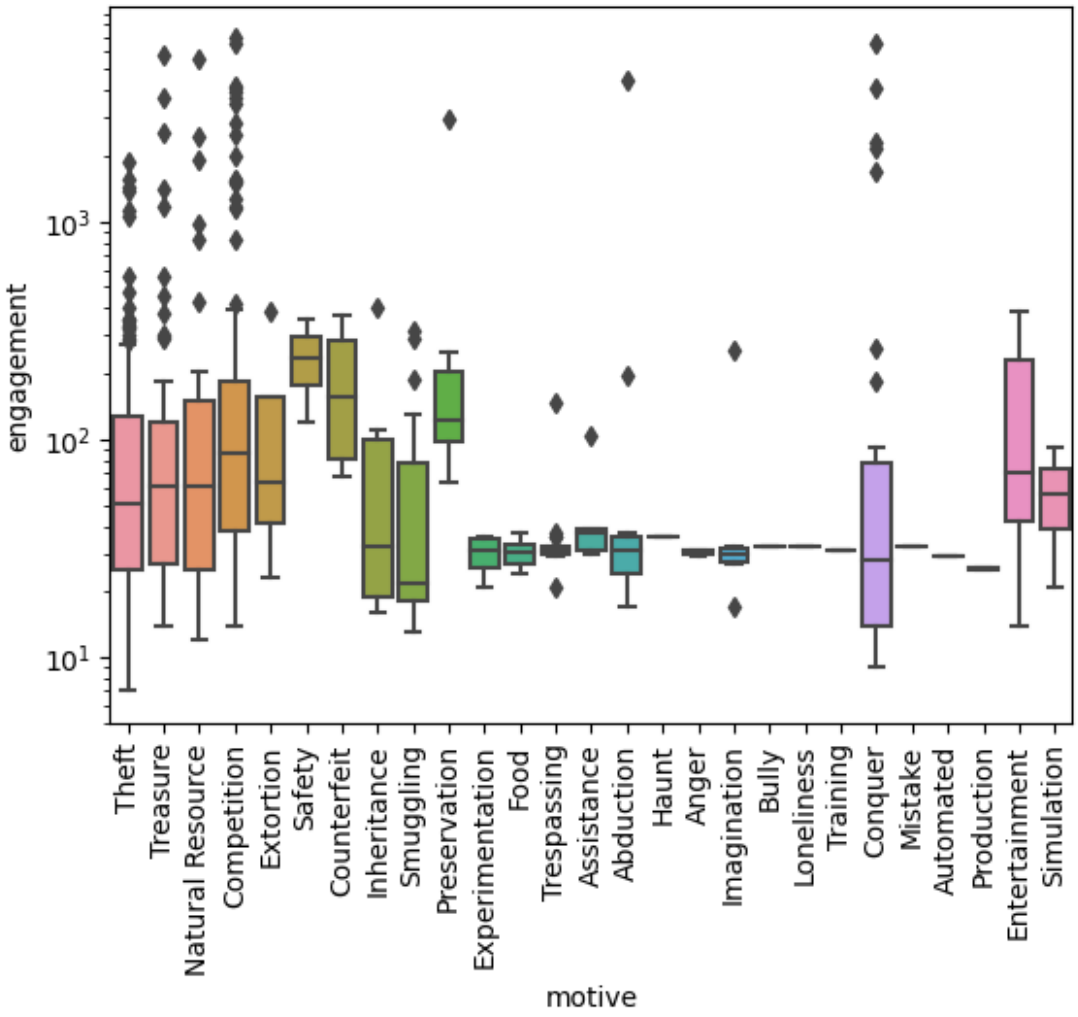
```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\3294091336.py:2:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  motive_plot =
motive_plot.set_xticklabels(motive_plot.get_xticklabels(), rotation =
90)
```




```
motive_plot = sb.boxplot(x = 'motive', y = 'imdb', data = data)
motive_plot =
motive_plot.set_xticklabels(motive_plot.get_xticklabels(), rotation =
90)
```

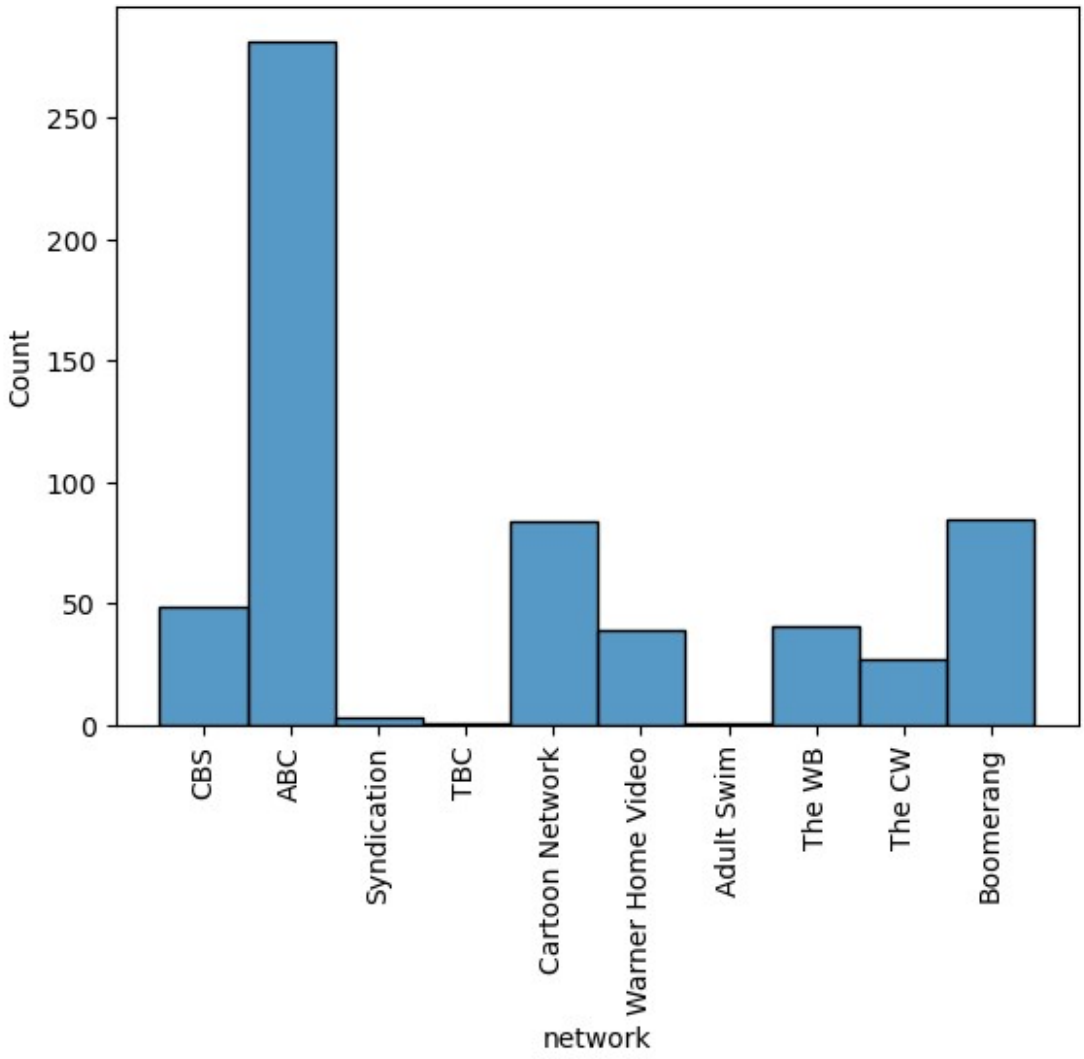



```
motive_plot = sb.boxplot(x = 'motive', y = 'engagement', data = data)
motive_plot.set_xticklabels(motive_plot.get_xticklabels(), rotation =
90)
motive_plot.set_yscale('log')
```

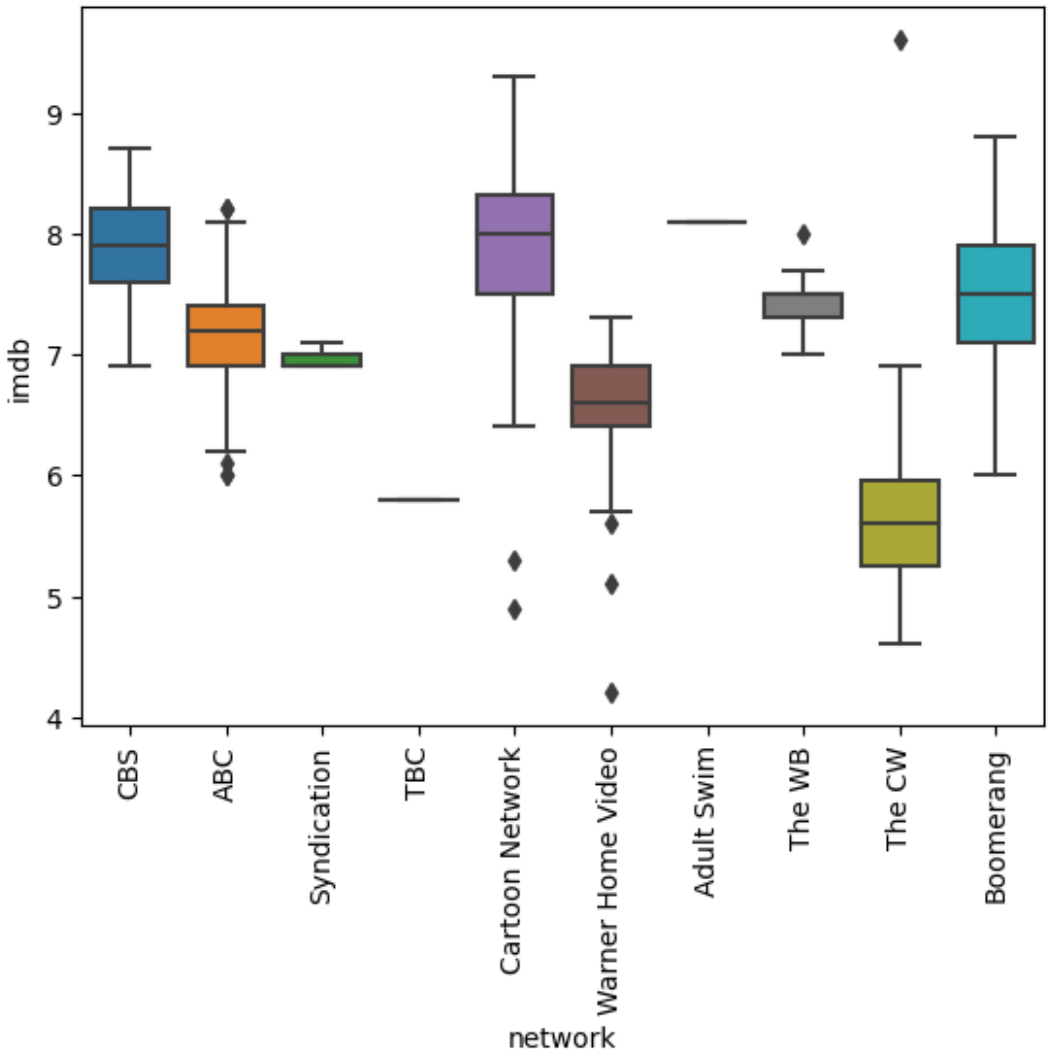


Ocena oraz zaangażowanie społeczności w zależności od nadawcy

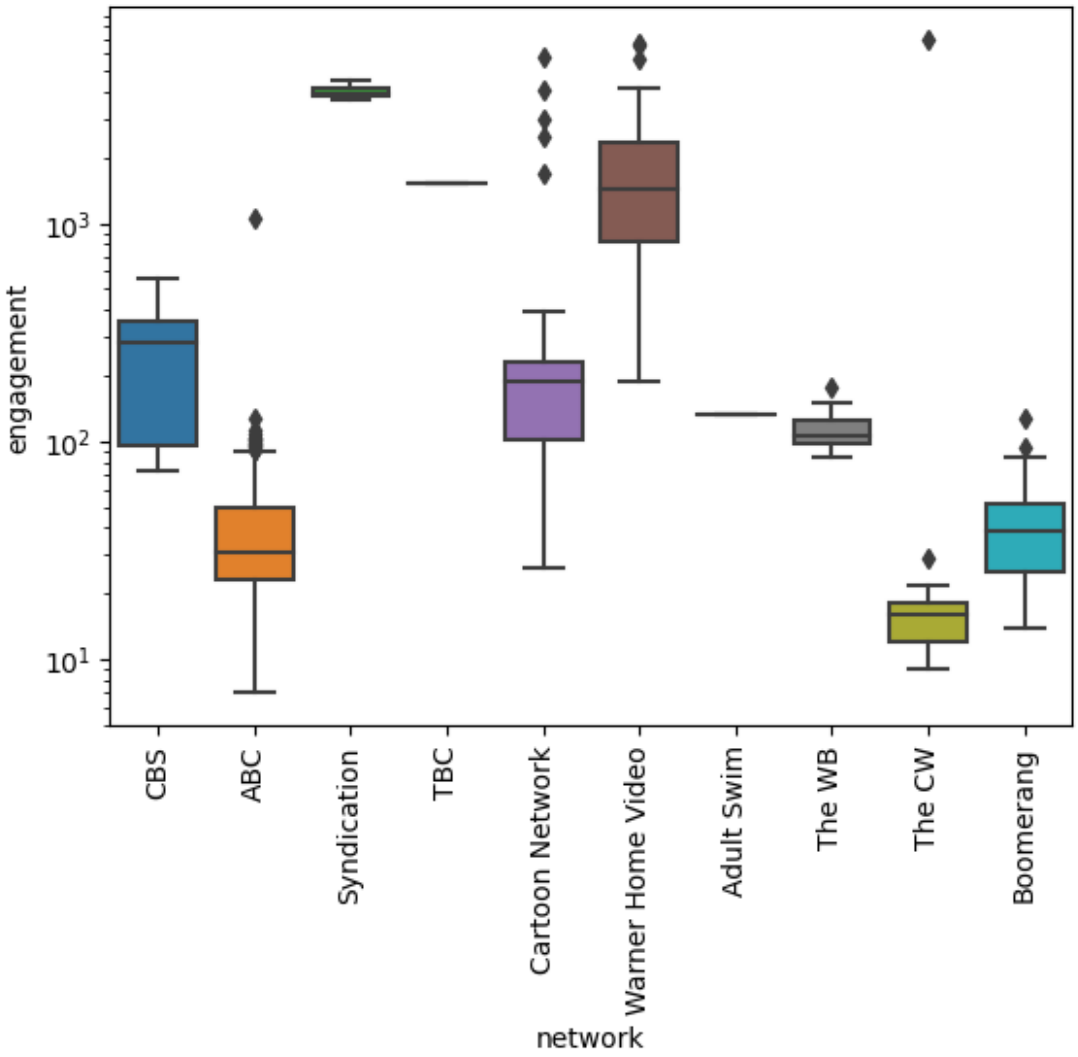
```
network_plot = sb.histplot(data, x = 'network', label = 'network')
network_plot =
network_plot.set_xticklabels(network_plot.get_xticklabels(), rotation
= 90)

C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\113526584.py:2:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  network_plot =
network_plot.set_xticklabels(network_plot.get_xticklabels(), rotation
= 90)
```




```
network_plot = sb.boxplot(x = 'network', y = 'imdb', data = data)
network_plot =
network_plot.set_xticklabels(network_plot.get_xticklabels(), rotation
= 90)
```



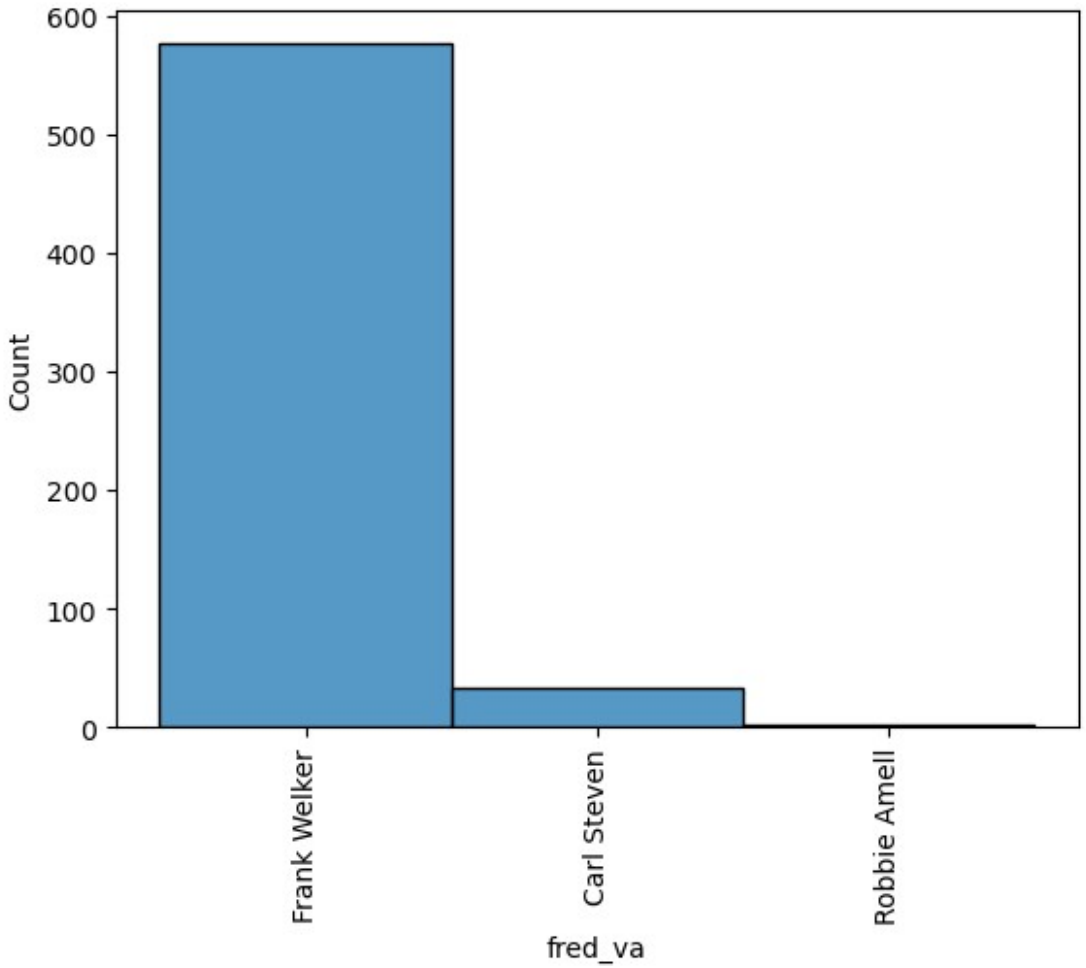

```
network_plot = sb.boxplot(x = 'network', y = 'engagement', data =
data)
network_plot.set_xticklabels(network_plot.get_xticklabels(), rotation
= 90)
network_plot.set_yscale('log')
```



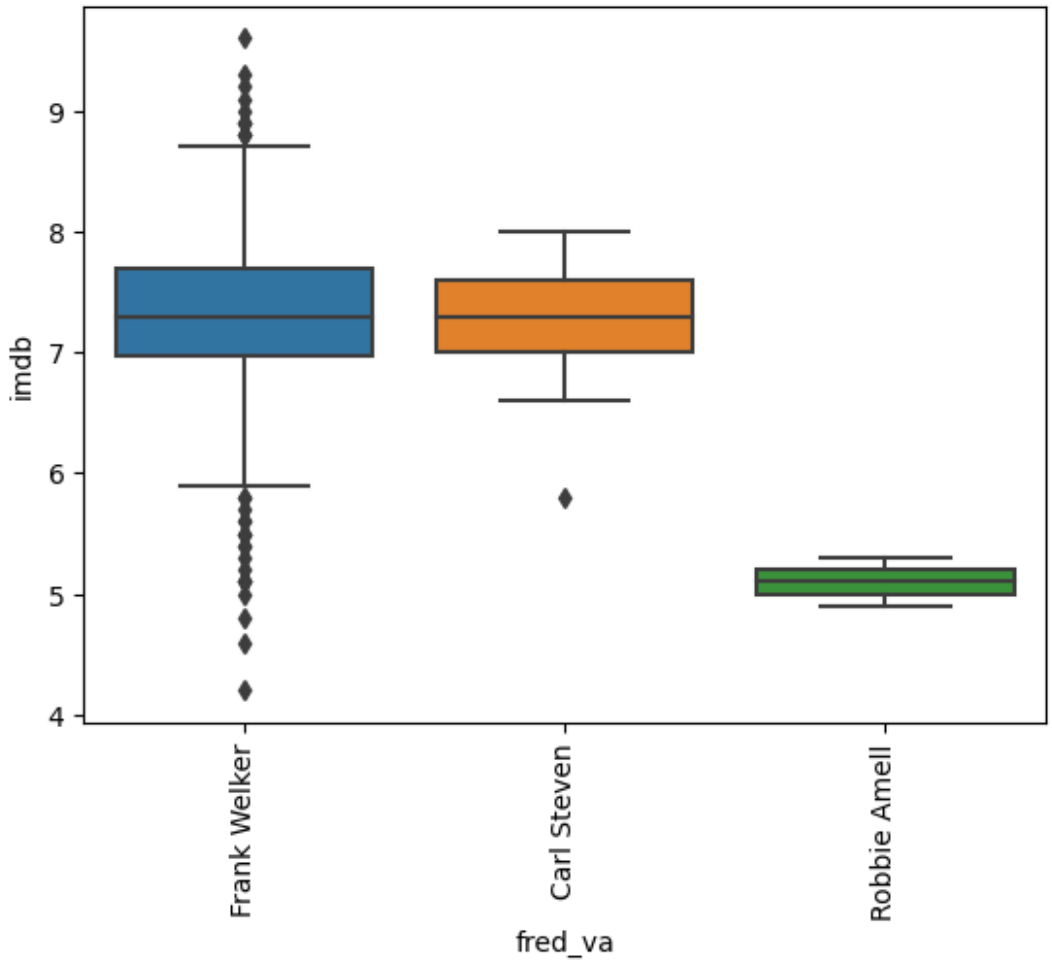
Ocena oraz zaangażowanie społeczności w zależności od doboru aktorów

```
fred_plot = sb.histplot(data, x = 'fred_va', label = 'fred')
fred_plot = fred_plot.set_xticklabels(fred_plot.get_xticklabels(),
rotation = 90)
```

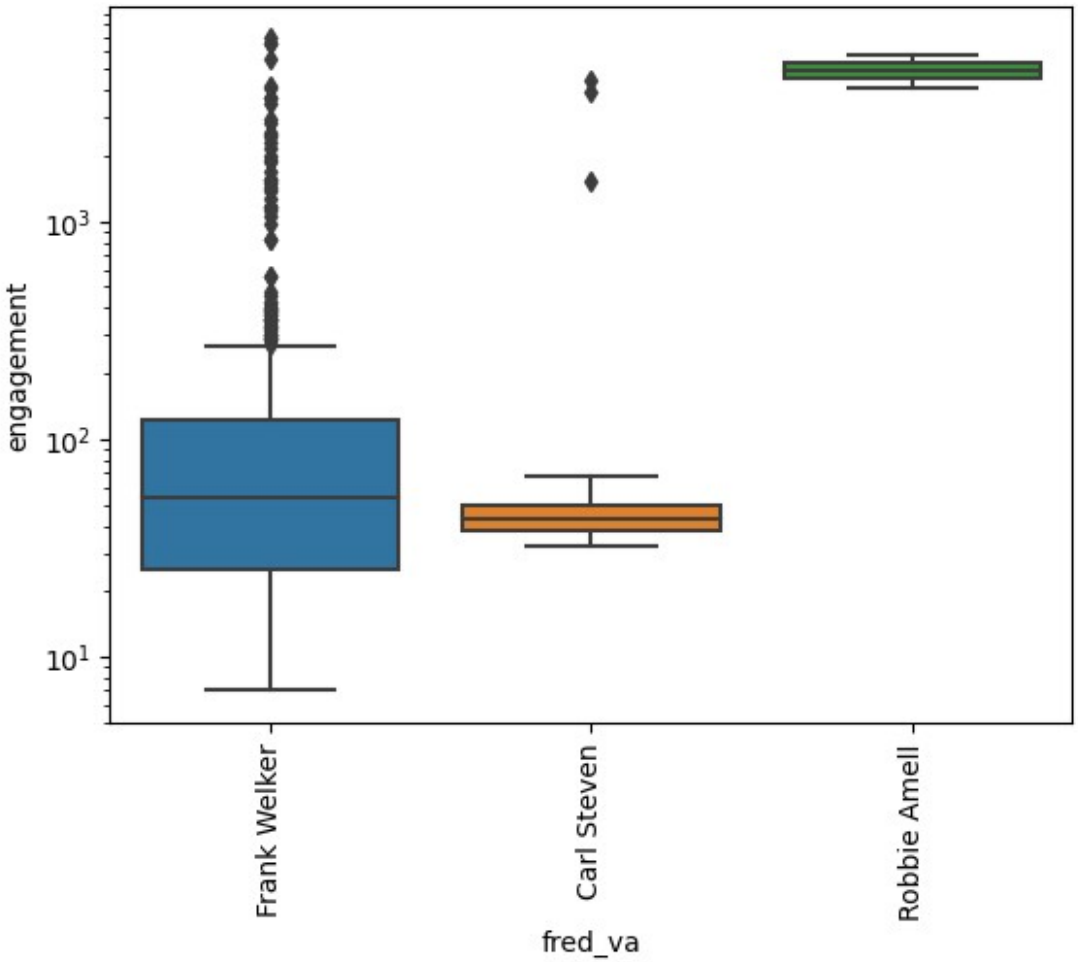
```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\2407014804.py:2:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  fred_plot = fred_plot.set_xticklabels(fred_plot.get_xticklabels(),
rotation = 90)
```




```
fred_plot = sb.boxplot(x = 'fred_va', y = 'imdb', data = data)
fred_plot = fred_plot.set_xticklabels(fred_plot.get_xticklabels(),
rotation = 90)
```

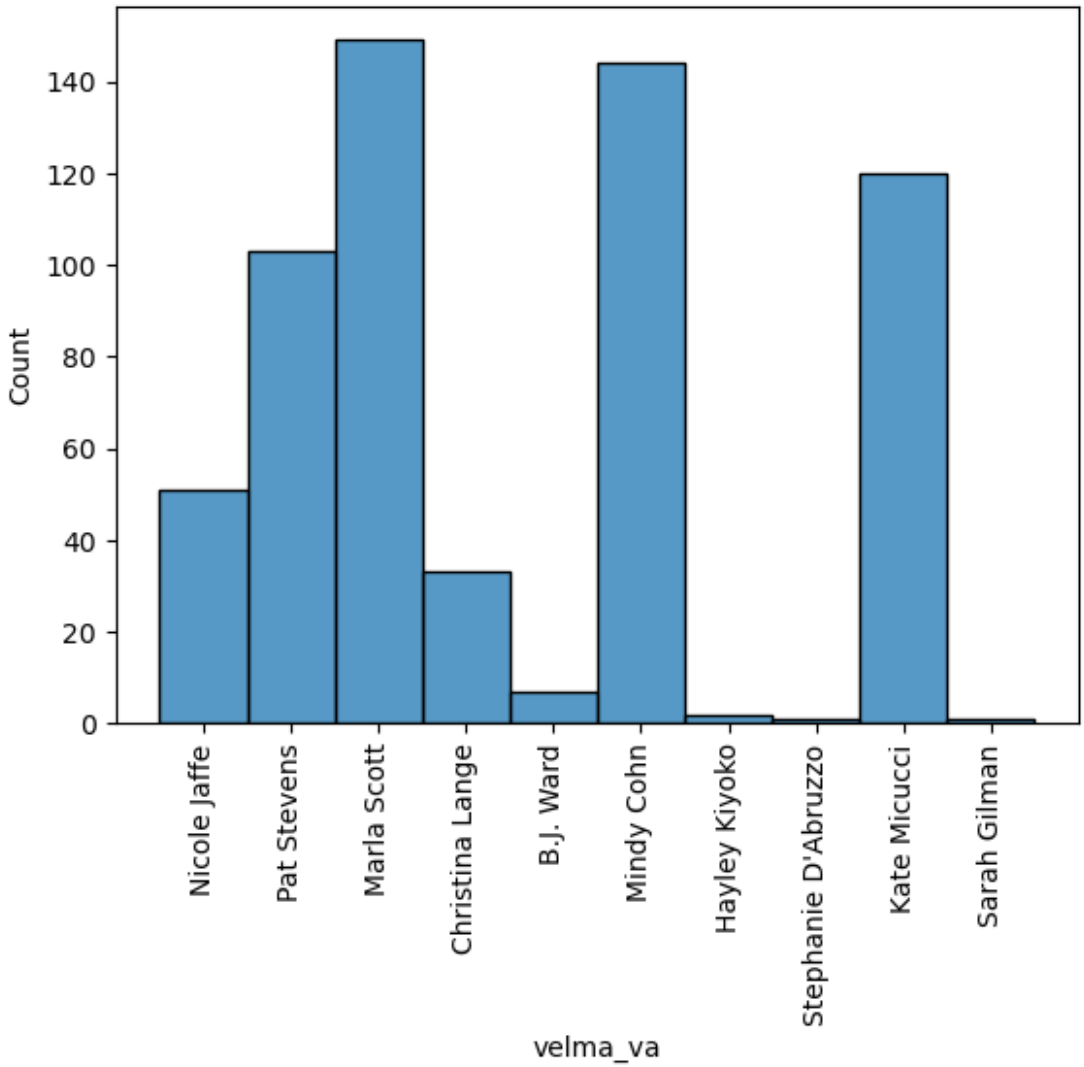



```
fred_plot = sb.boxplot(x = 'fred_va', y = 'engagement', data = data)
fred_plot.set_xticklabels(fred_plot.get_xticklabels(), rotation = 90)
fred_plot.set_yscale('log')
```

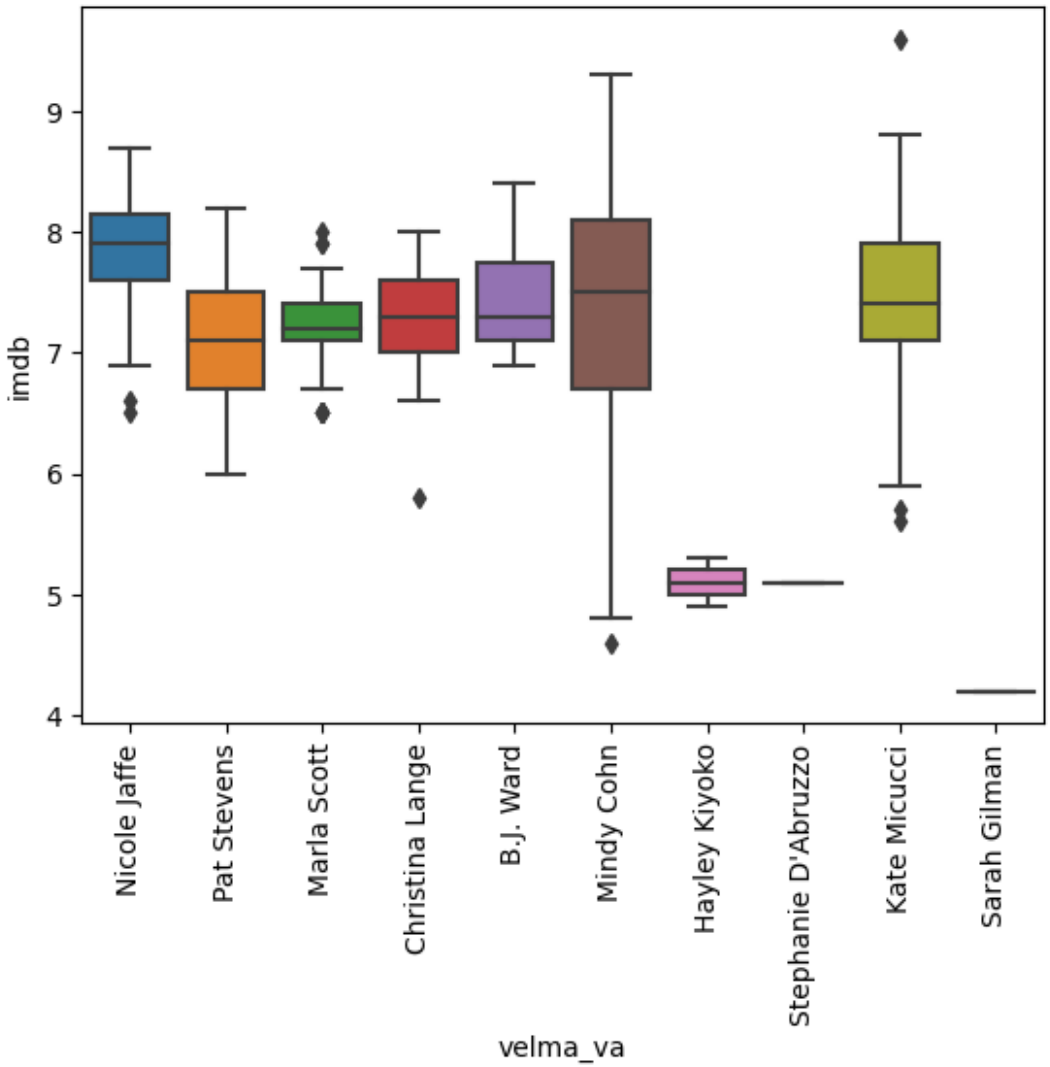



```
velma_plot = sb.histplot(data, x = 'velma_va', label = 'velma')
velma_plot = velma_plot.set_xticklabels(velma_plot.get_xticklabels(),
rotation = 90)
```

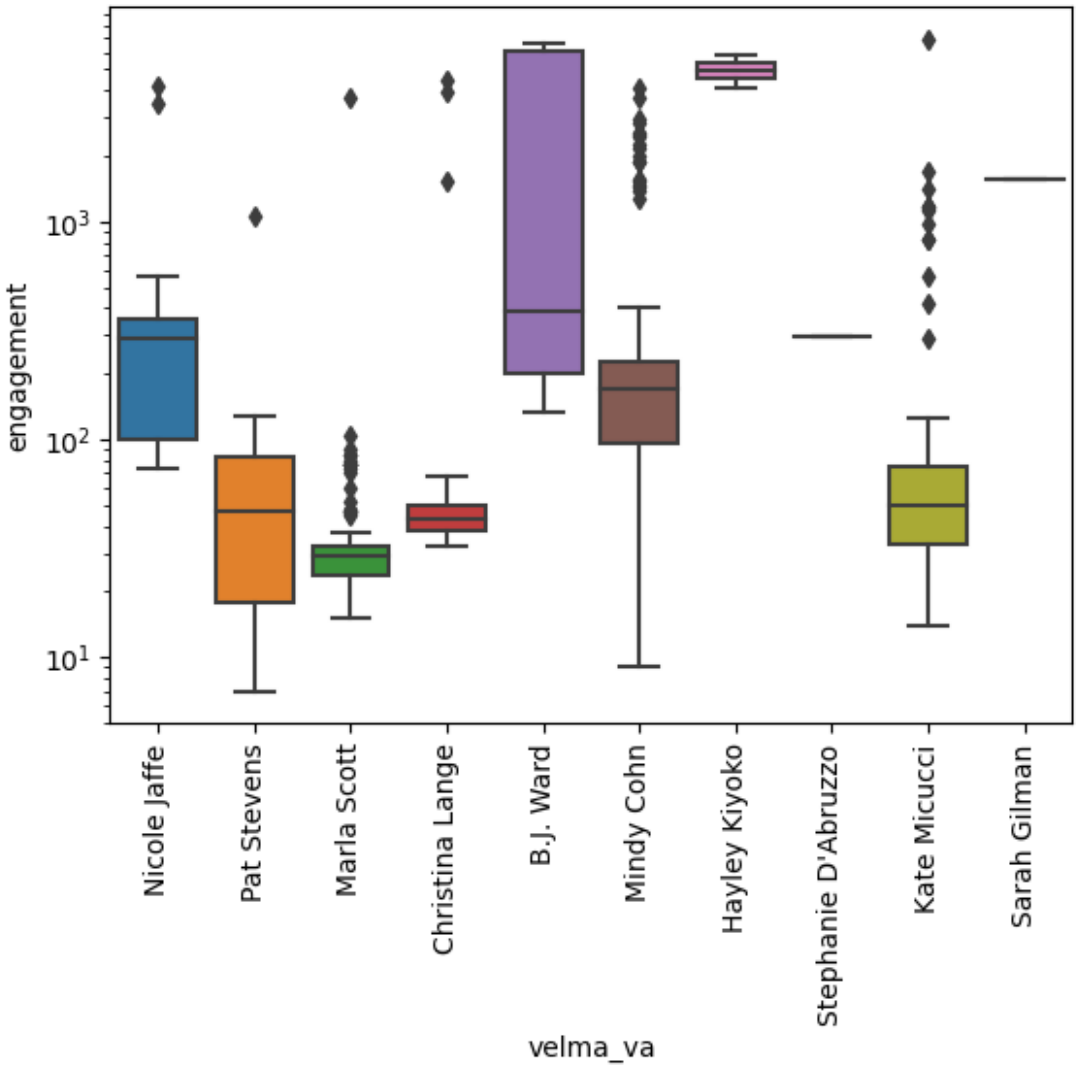
```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\1757776920.py:2:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  velma_plot =
velma_plot.set_xticklabels(velma_plot.get_xticklabels(), rotation =
90)
```




```
velma_plot = sb.boxplot(x = 'velma_va', y = 'imdb', data = data)
velma_plot = velma_plot.set_xticklabels(velma_plot.get_xticklabels(),
rotation = 90)
```

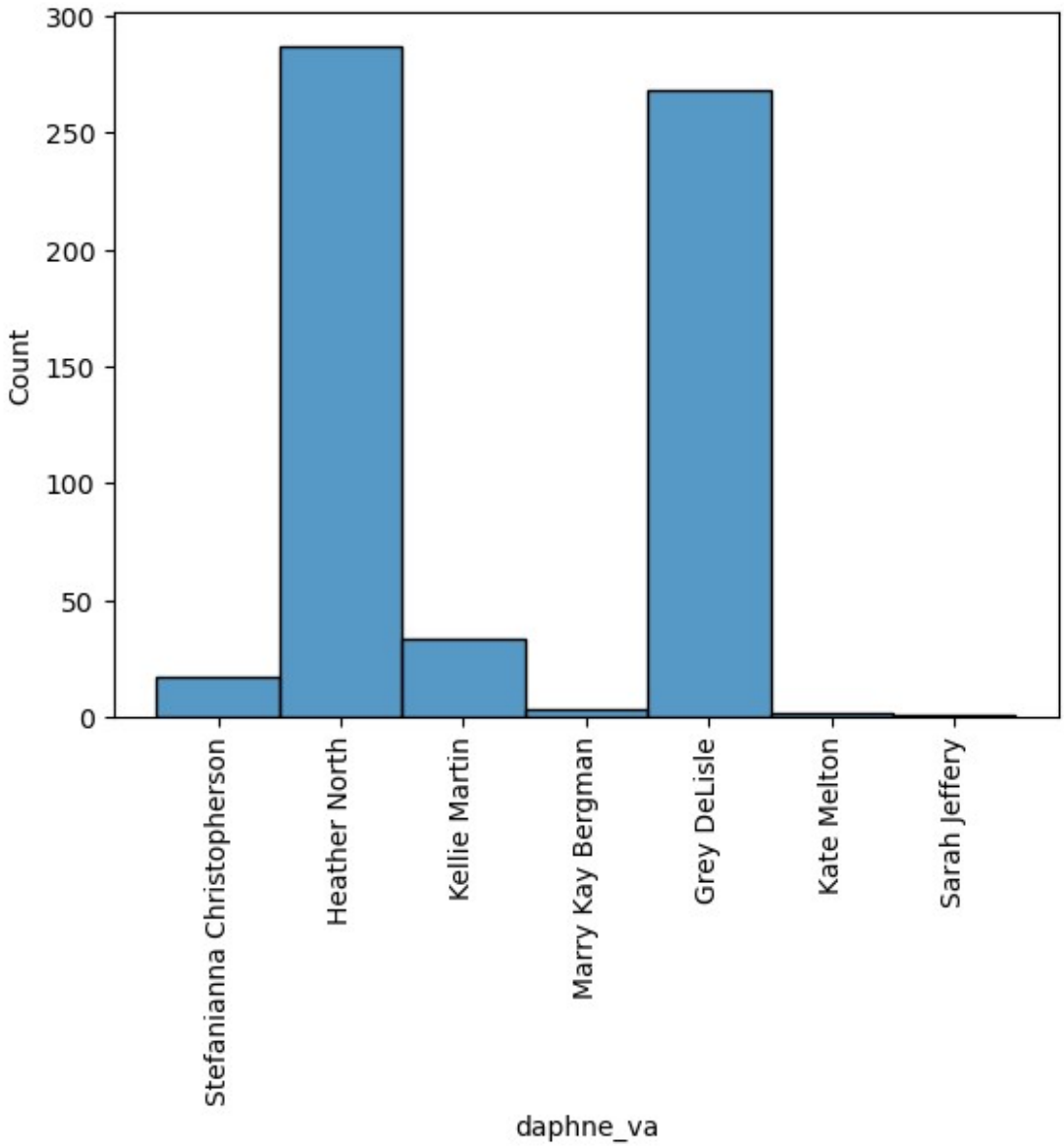



```
velma_plot = sb.boxplot(x = 'velma_va', y = 'engagement', data = data)
velma_plot.set_xticklabels(velma_plot.get_xticklabels(), rotation =
90)
velma_plot.set_yscale('log')
```

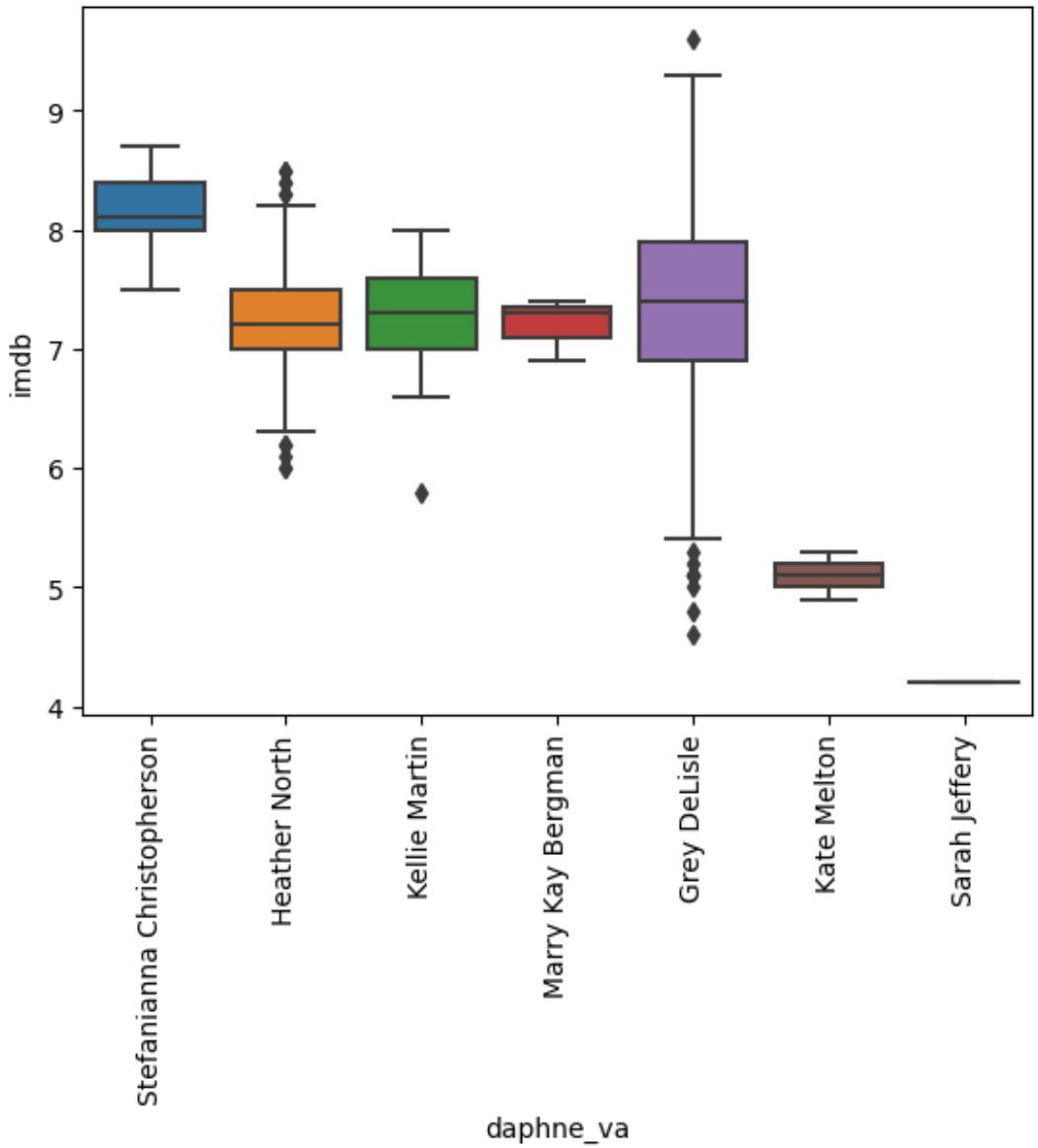



```
daphne_plot = sb.histplot(data, x = 'daphne_va', label = 'daphne')
daphne_plot =
daphne_plot.set_xticklabels(daphne_plot.get_xticklabels(), rotation =
90)
```

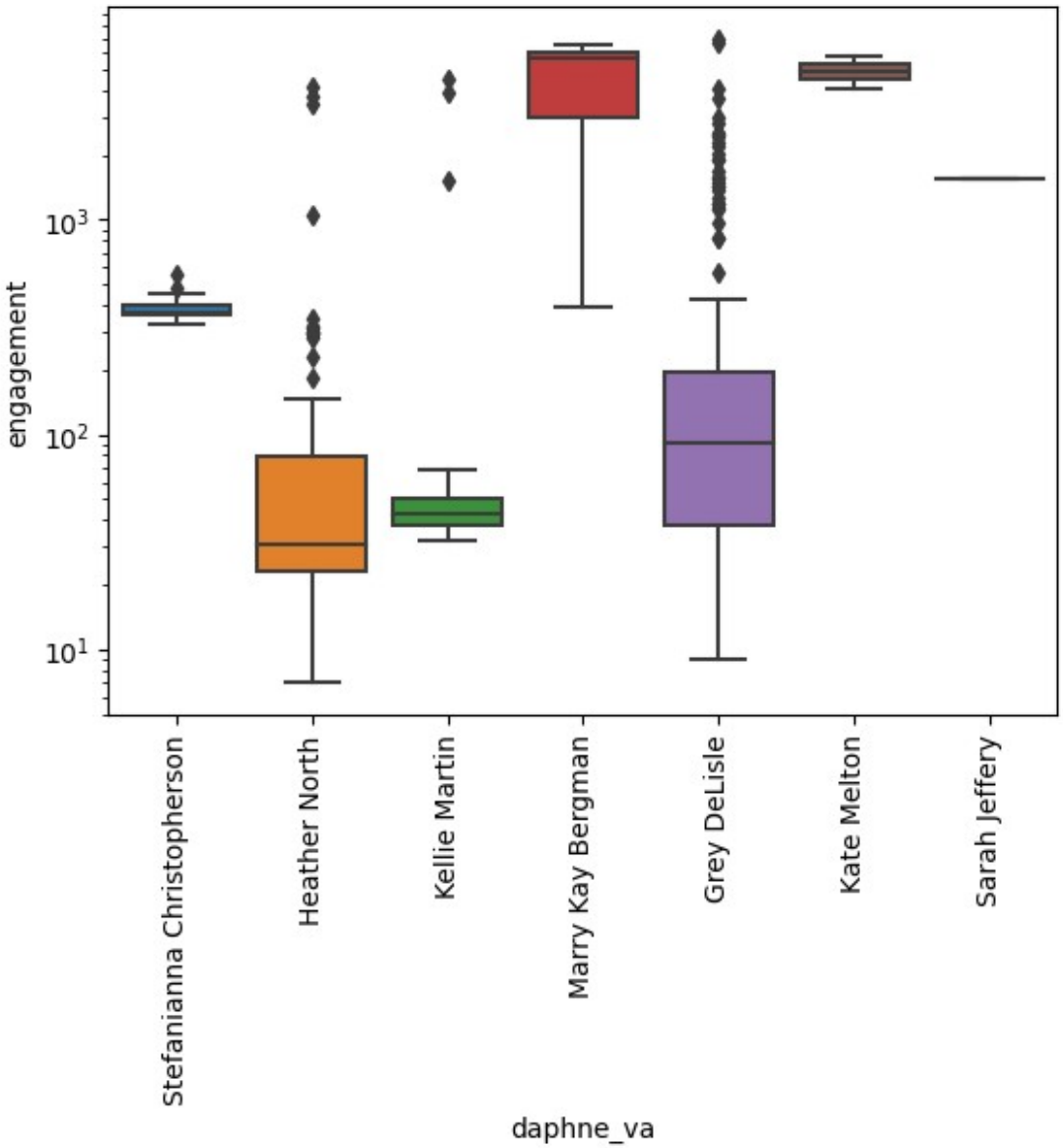
```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\1248762252.py:2:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  daphne_plot =
daphne_plot.set_xticklabels(daphne_plot.get_xticklabels(), rotation =
90)
```



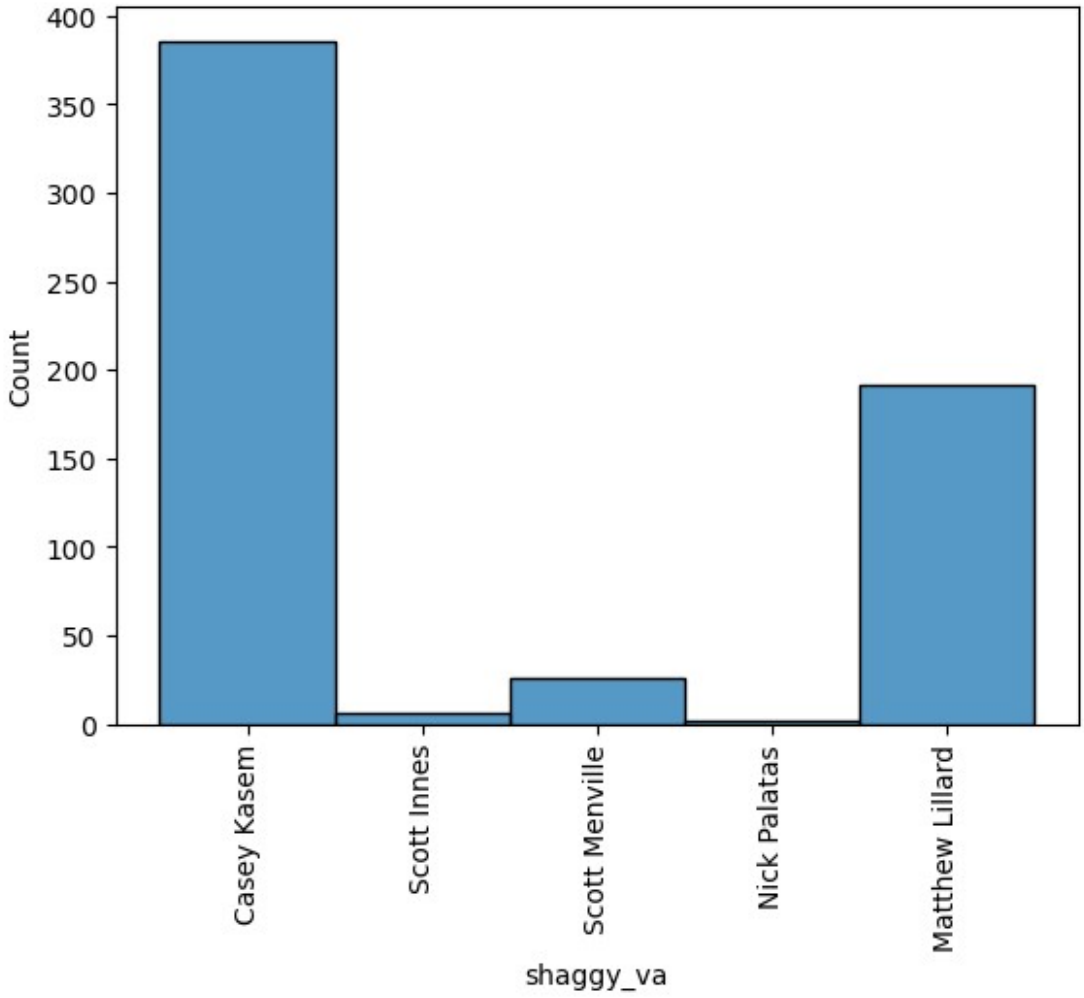

```
daphne_plot = sb.boxplot(x = 'daphne_va', y = 'imdb', data = data)
daphne_plot =
daphne_plot.set_xticklabels(daphne_plot.get_xticklabels(), rotation =
90)
```



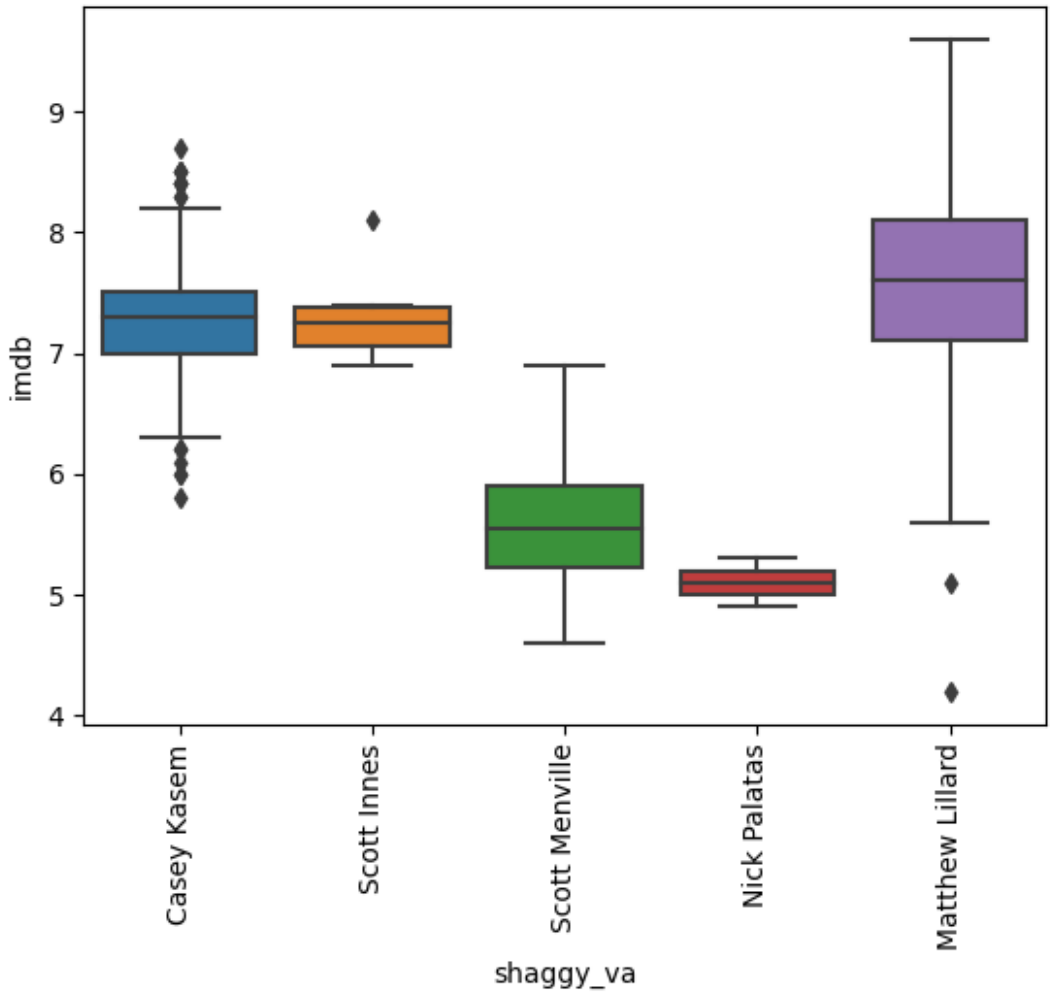

```
daphne_plot = sb.boxplot(x = 'daphne_va', y = 'engagement', data =
data)
daphne_plot.set_xticklabels(daphne_plot.get_xticklabels(), rotation =
90)
daphne_plot.set_yscale('log')
```



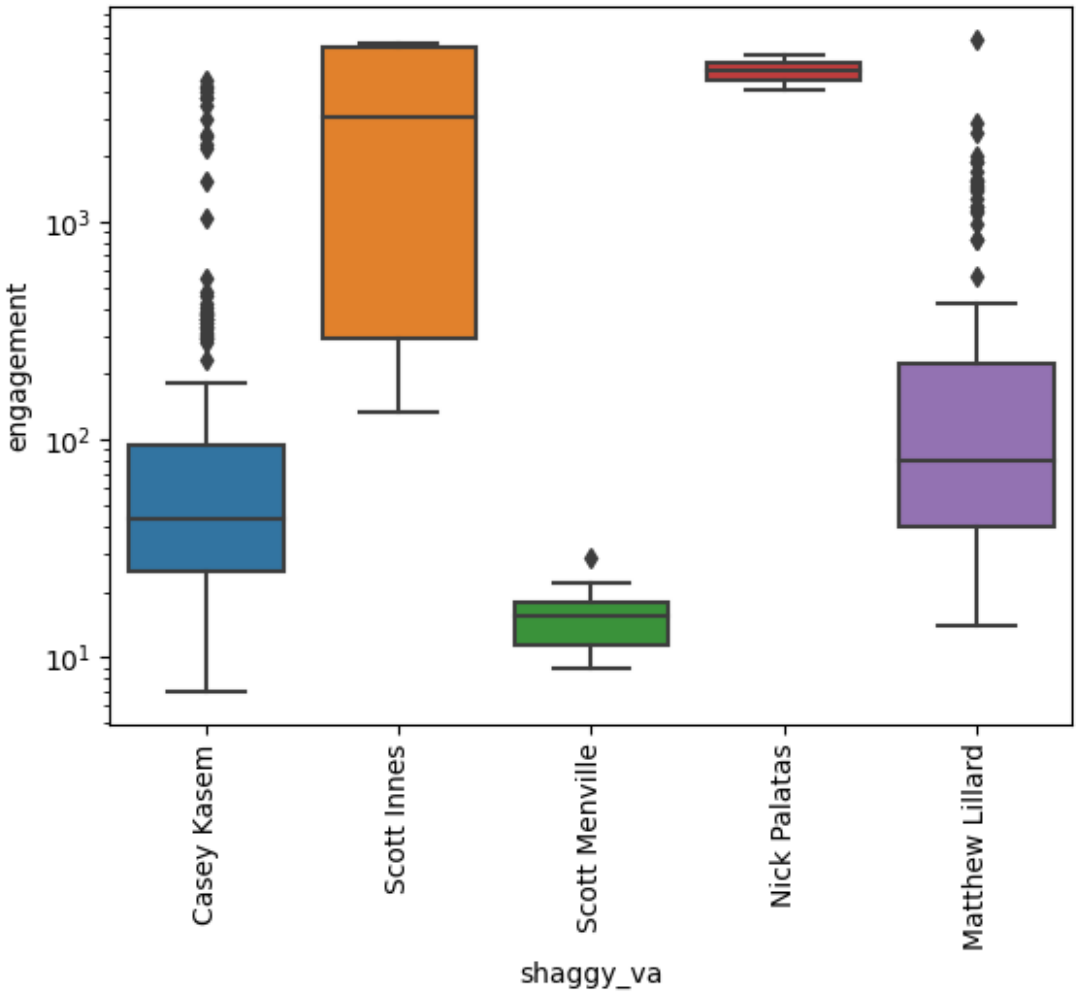

```
shaggy_plot = sb.histplot(data, x = 'shaggy_va', label = 'shaggy')
shaggy_plot =
shaggy_plot.set_xticklabels(shaggy_plot.get_xticklabels(), rotation =
90)

C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\3752273912.py:2:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  shaggy_plot =
shaggy_plot.set_xticklabels(shaggy_plot.get_xticklabels(), rotation =
90)
```



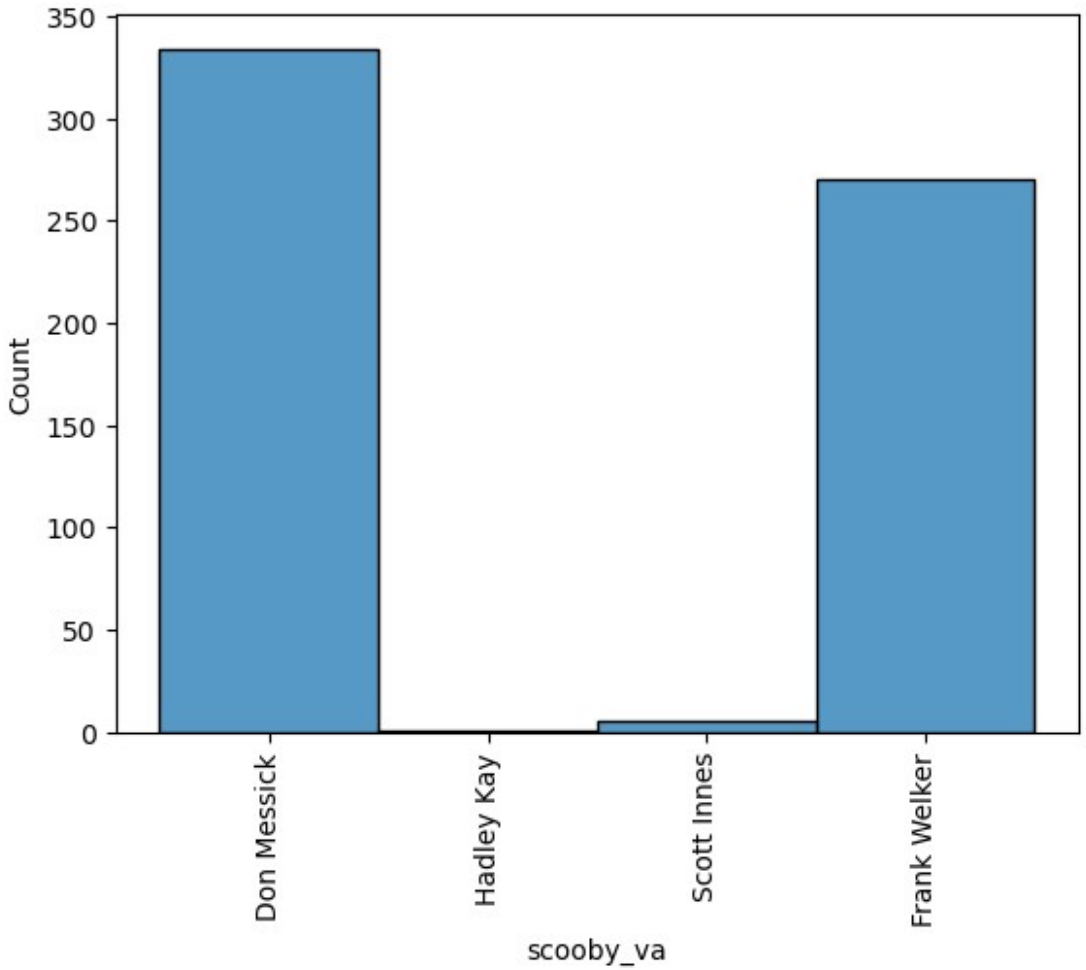

```
shaggy_plot = sb.boxplot(x = 'shaggy_va', y = 'imdb', data = data)
shaggy_plot =
shaggy_plot.set_xticklabels(shaggy_plot.get_xticklabels(), rotation =
90)
```



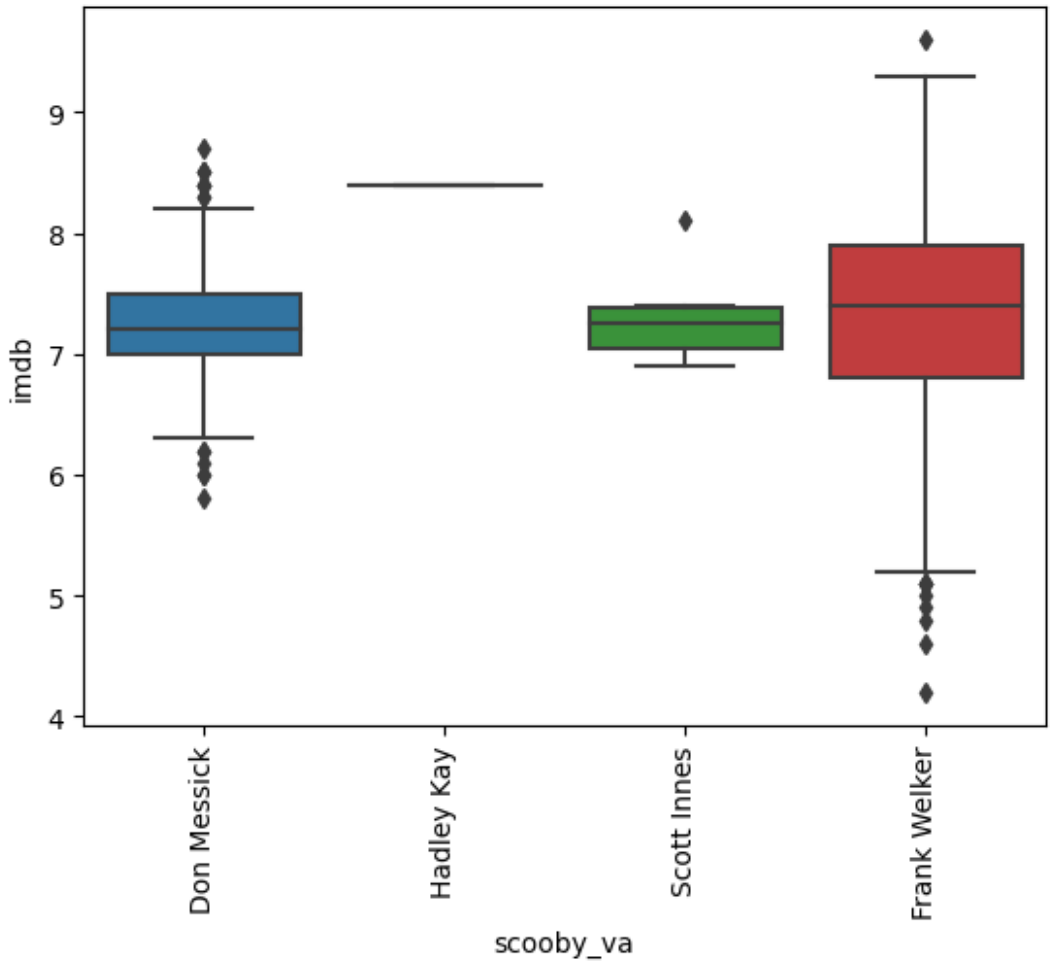

```
shaggy_plot = sb.boxplot(x = 'shaggy_va', y = 'engagement', data =
data)
shaggy_plot.set_xticklabels(shaggy_plot.get_xticklabels(), rotation =
90)
shaggy_plot.set_yscale('log')
```



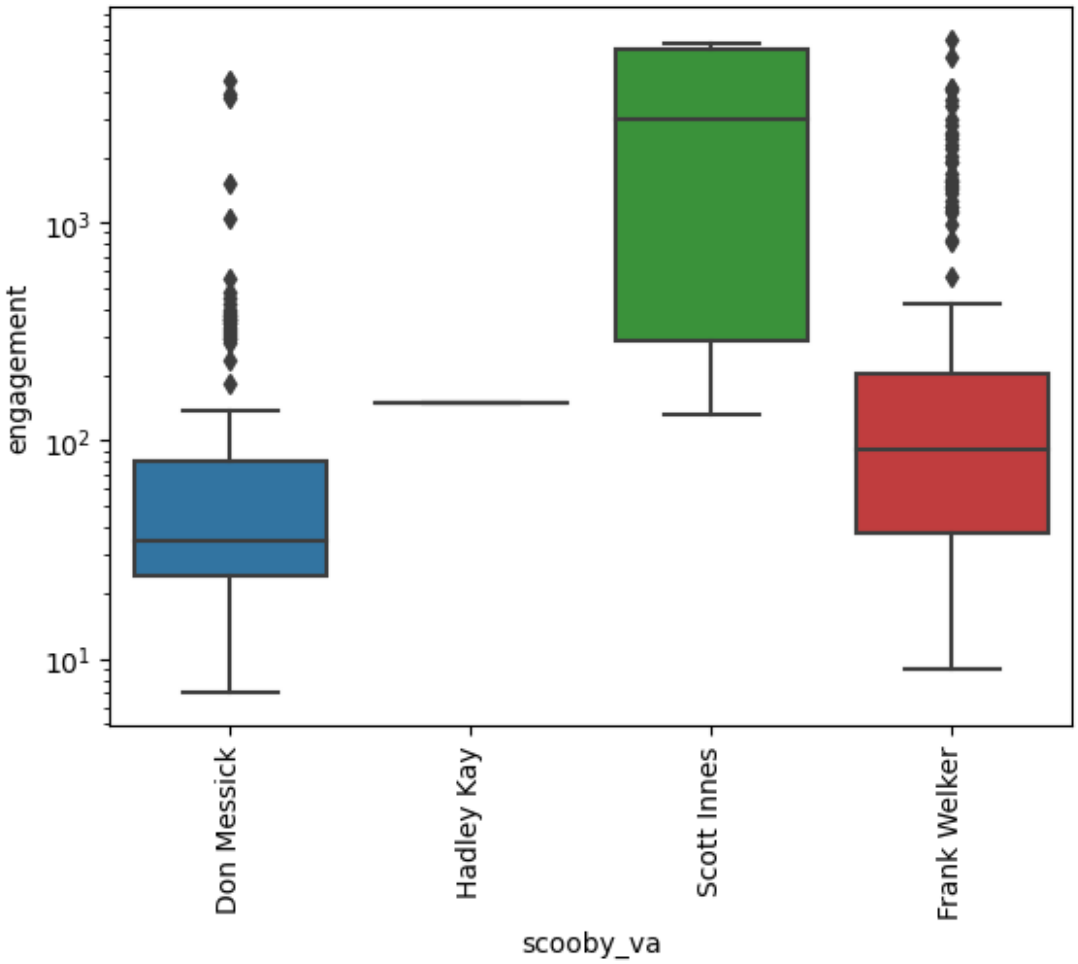

```
scooby_plot = sb.histplot(data, x = 'scooby_va', label = 'scooby')
scooby_plot =
scooby_plot.set_xticklabels(scooby_plot.get_xticklabels(), rotation =
90)

C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\3047341647.py:2:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  scooby_plot =
scooby_plot.set_xticklabels(scooby_plot.get_xticklabels(), rotation =
90)
```




```
scooby_plot = sb.boxplot(x = 'scooby_va', y = 'imdb', data = data)
scooby_plot =
scooby_plot.set_xticklabels(scooby_plot.get_xticklabels(), rotation =
90)
```

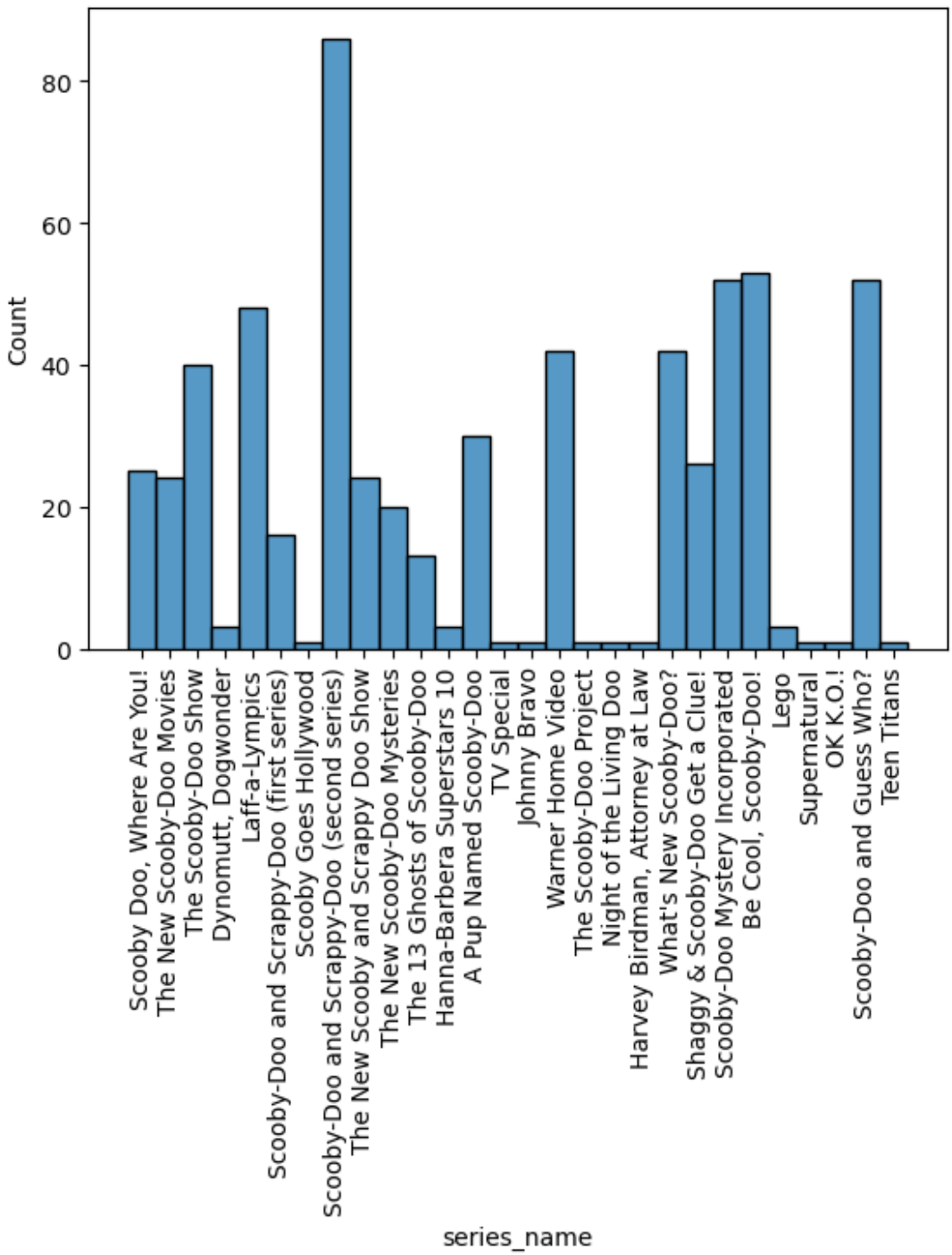



```
scooby_plot = sb.boxplot(x = 'scooby_va', y = 'engagement', data =
data)
scooby_plot.set_xticklabels(scooby_plot.get_xticklabels(), rotation =
90)
scooby_plot.set_yscale('log')
```

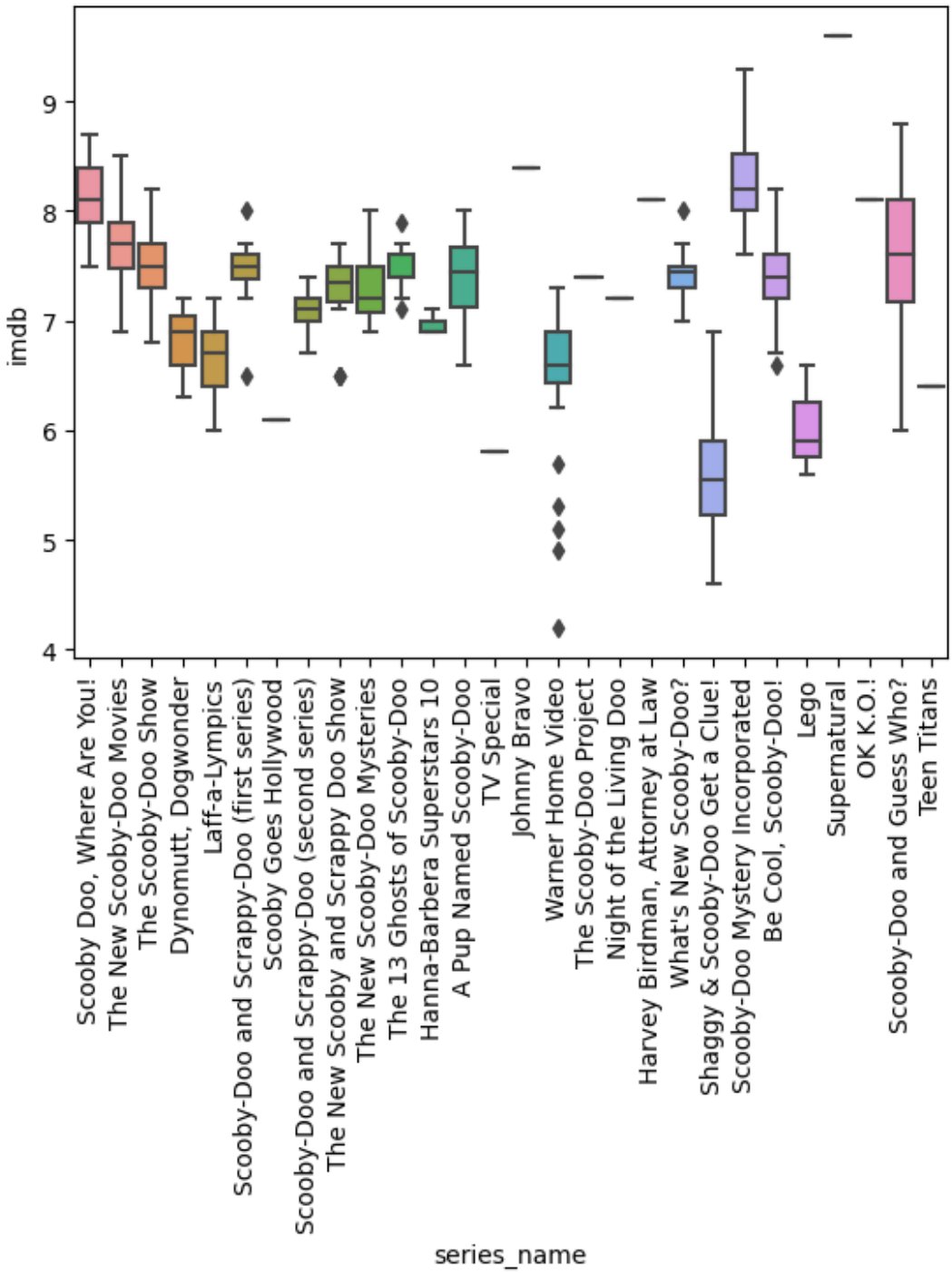


Ocena oraz zaangażowanie społeczności w zależności od serii

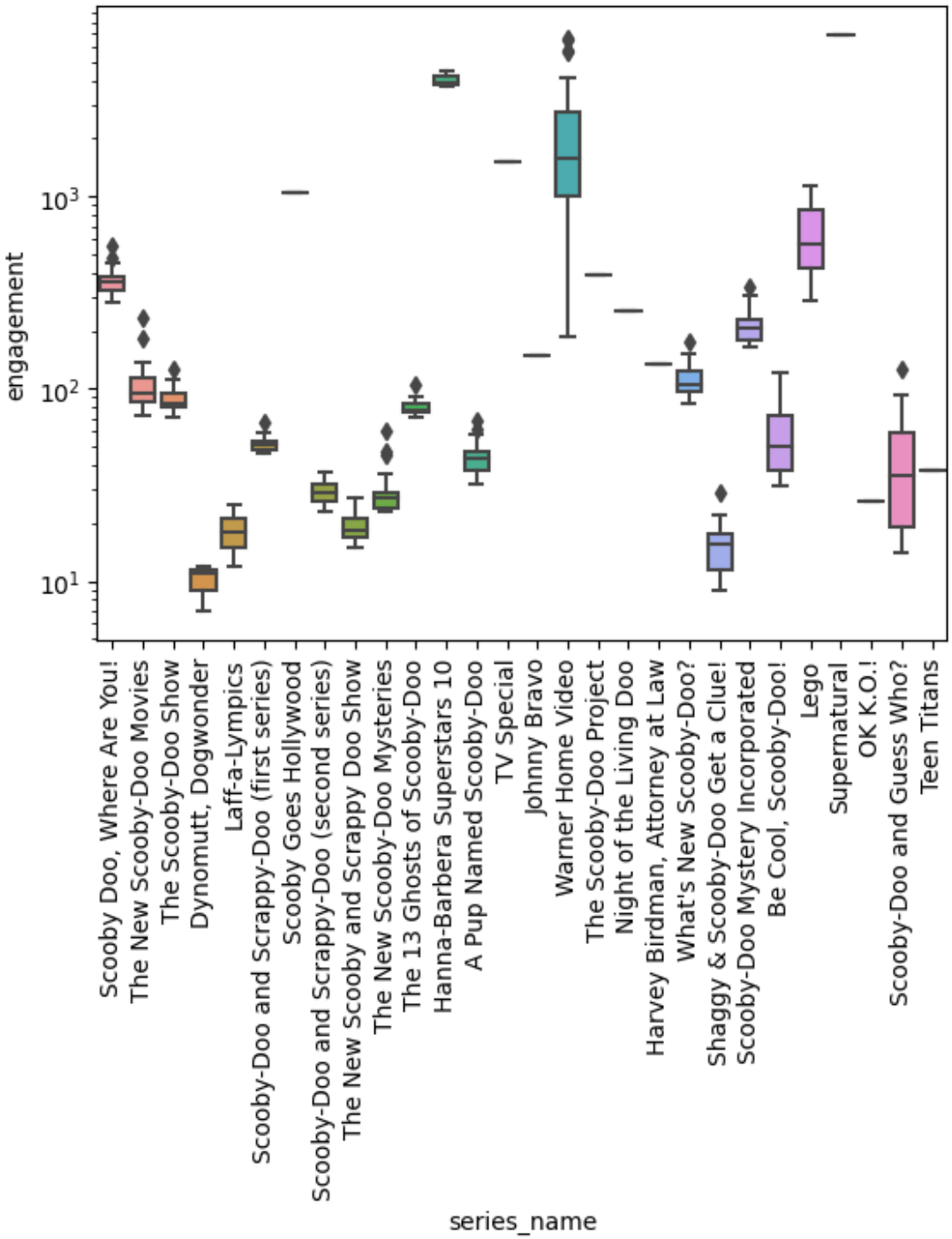
```
series_plot = sb.histplot(data, x = 'series_name', label = 'series')
series_plot =
series_plot.set_xticklabels(series_plot.get_xticklabels(), rotation =
90)

C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\3601418326.py:2:
UserWarning: FixedFormatter should only be used together with
FixedLocator
  series_plot =
series_plot.set_xticklabels(series_plot.get_xticklabels(), rotation =
90)
```




```
series_plot = sb.boxplot(x = 'series_name', y = 'imdb', data = data)
series_plot =
series_plot.set_xticklabels(series_plot.get_xticklabels(), rotation =
90)
```

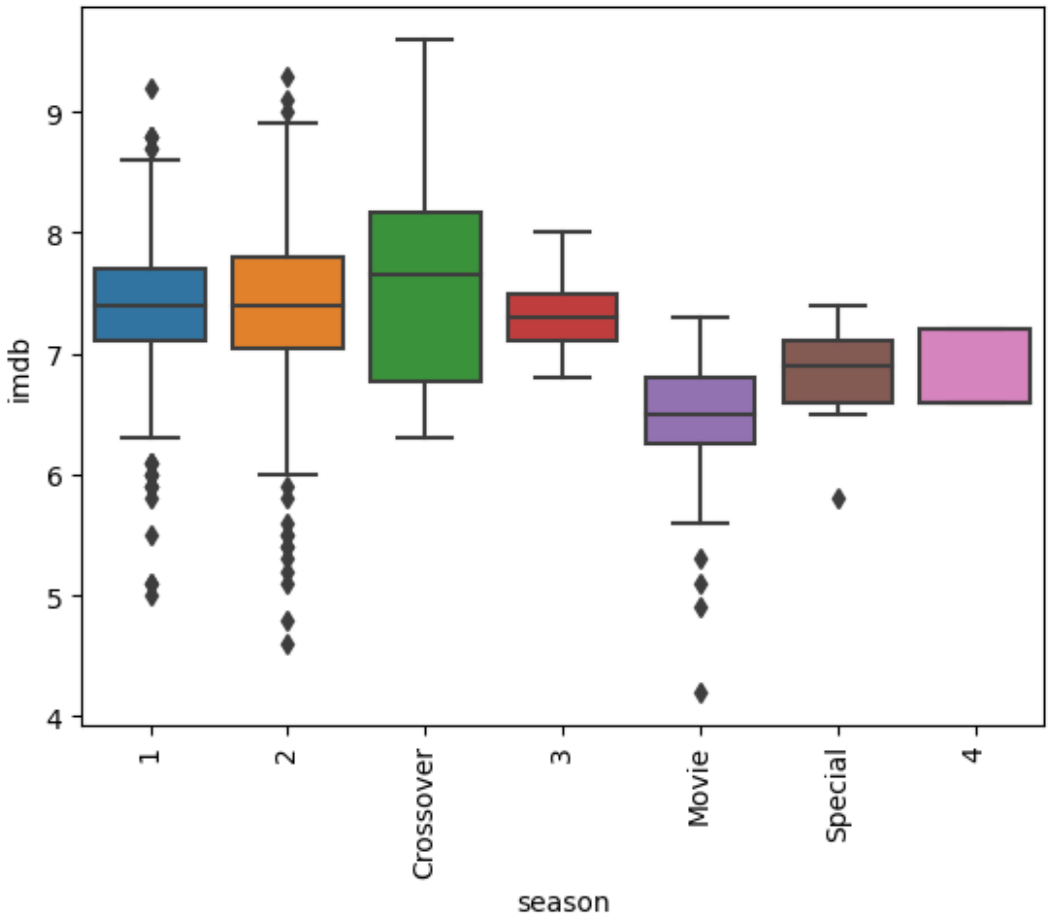



```
series_plot = sb.boxplot(x = 'series_name', y = 'engagement', data =
data)
series_plot.set_xticklabels(series_plot.get_xticklabels(), rotation =
90)
series_plot.set_yscale('log')
```



Ocena oraz zaangażowanie społeczności w zależności od sezonu/typu

```
season_plot = sb.boxplot(x = 'season', y = 'imdb', data = data)
season_plot =
season_plot.set_xticklabels(season_plot.get_xticklabels(), rotation =
90)
```



Zależność poszczególnych zmiennych numerycznych

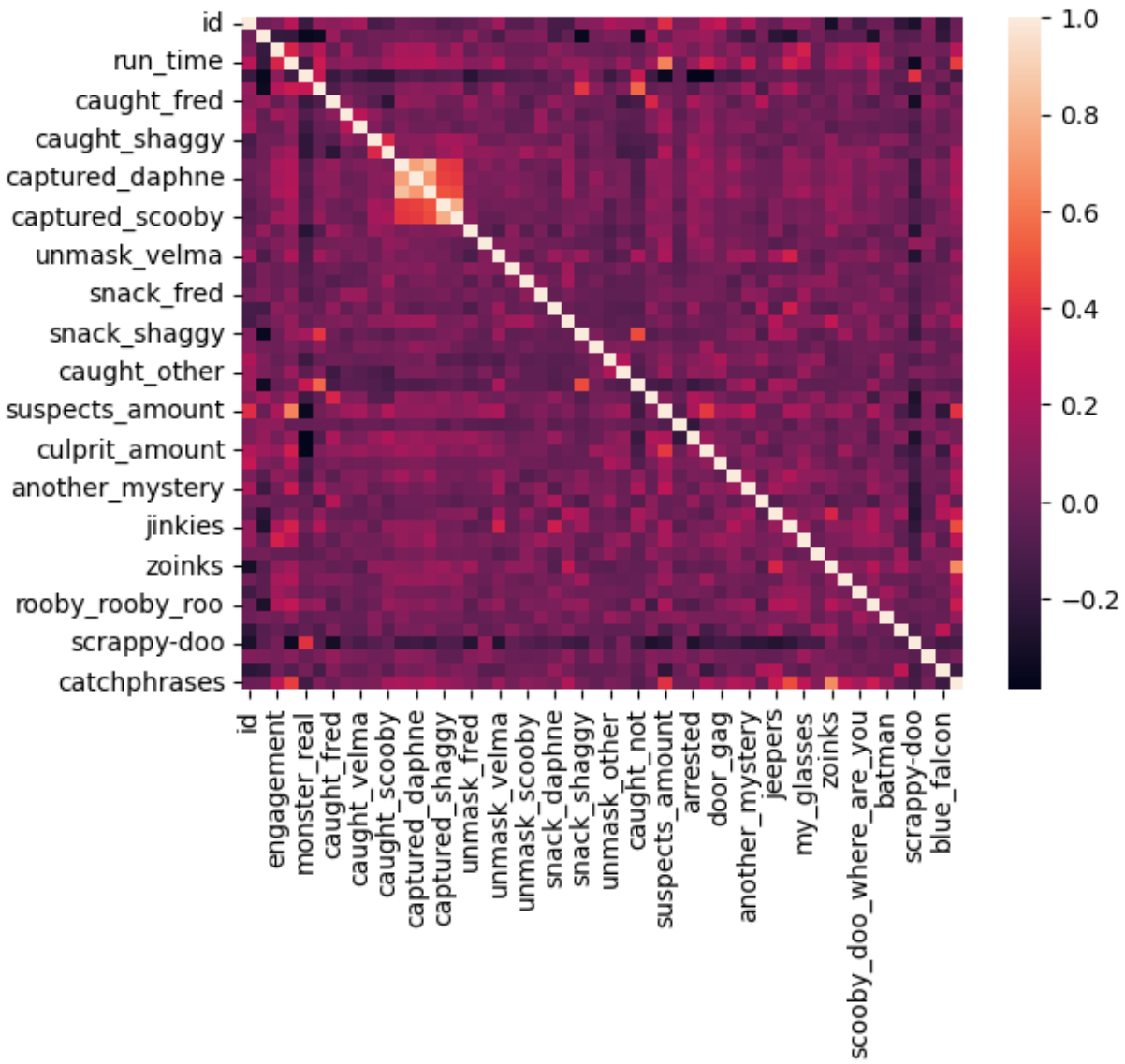
Macierz kowariancji zmiennych numerycznych

```
data.cov(numeric_only = True)
```

Macierz korelacji zmiennych numerycznych

```
correlation_matrix = data.corr(numeric_only = True)
correlation_matrix

numeric_data = data.select_dtypes(include = np.number)

sb.heatmap(correlation_matrix)
```

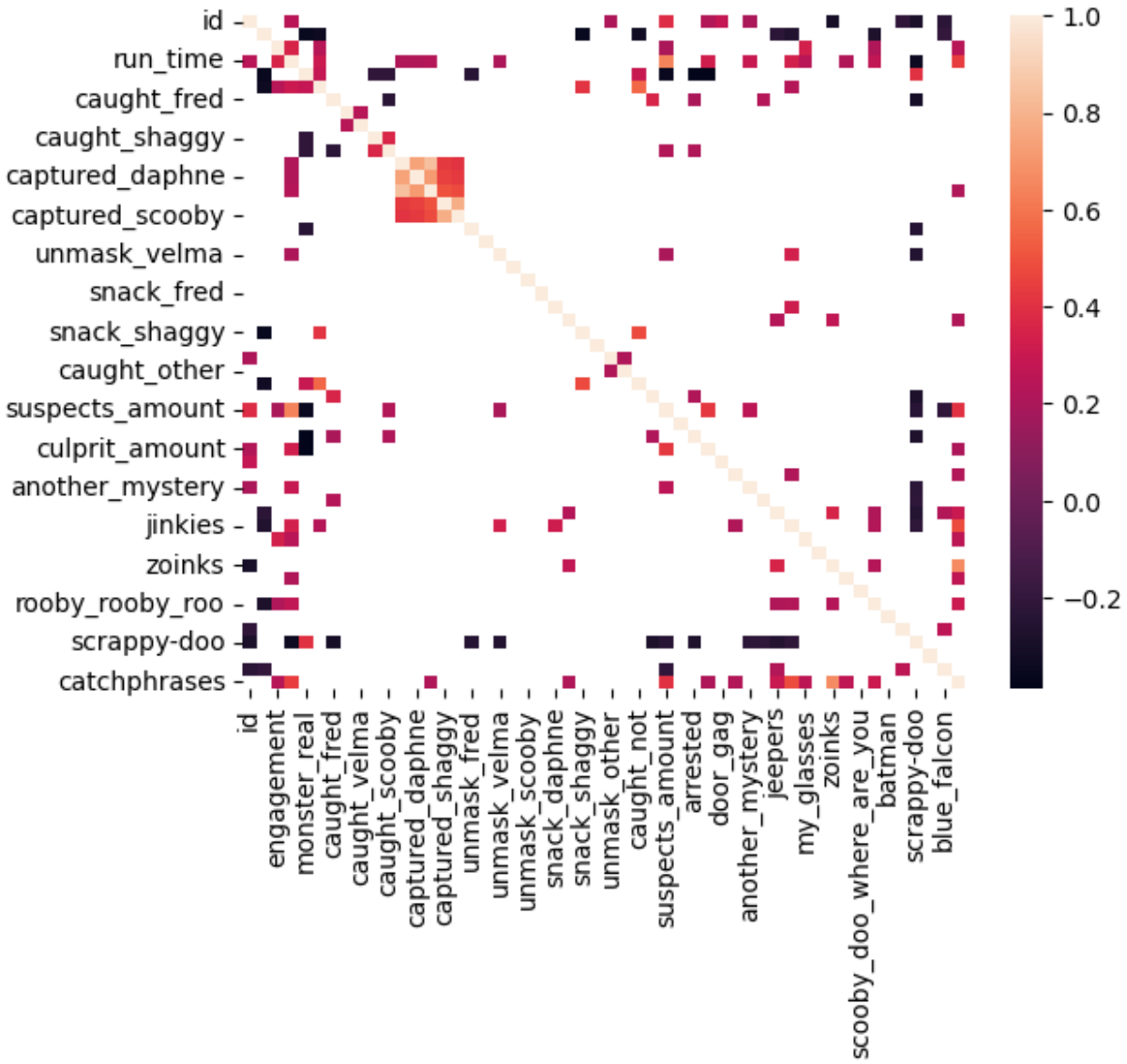
```
<Axes: >
```



Jak można zaobserwować w macierzy korelacji, bardzo wiele cech nie jest skorelowanych. Poniżej możemy zobaczyć cechy, ktore łączy co najmniej słaba korelacja

```
sb.heatmap(correlation_matrix[abs(correlation_matrix) >= 0.2])
```

```
<Axes: >
```



Ponieważ nasza analiza skupia się głównie na ocenie oraz popularności poszczególnych produkcji, skupimy się na wyłącznie na cechach, które są co najmniej słabo skorelowane z 'imdb' lub 'engagement'

Choć żadna zmienna nie wykazuje silnej korelacji z ostateczną oceną użytkowników, to istnieją pewne cechy, które w przeciętnym (lub małym - zależnie od przyjętej klasyfikacji, przyjęta została klasyfikacja Stanisza) stopniu korelują z ostateczną oceną produkcji przez widzów.

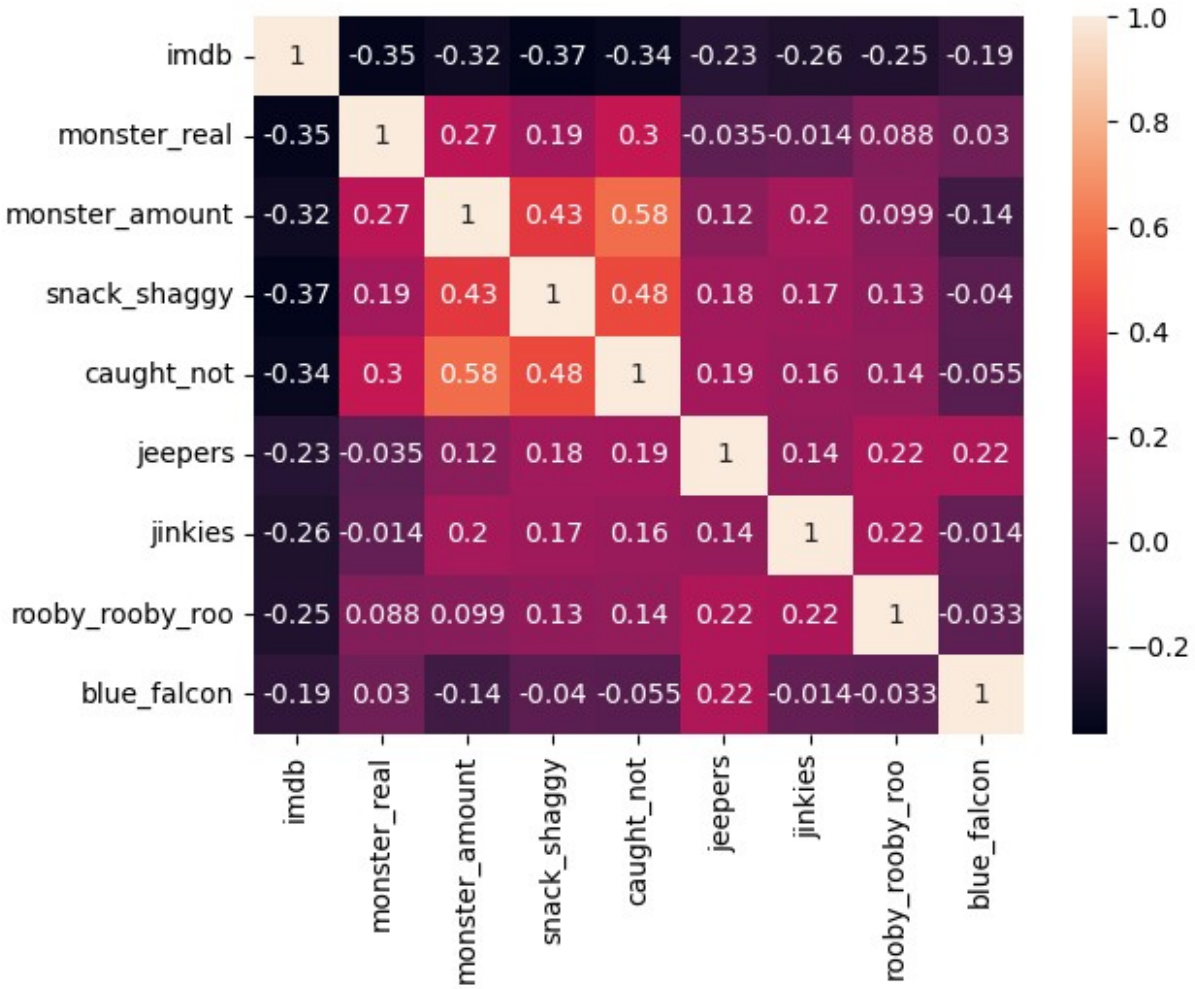
W przeciętnym stopniu na ocenę wpływają:

1. realność potwora (sytuacja, w której potwór okazuje się nie być człowiekiem, robotem lub czymś wytrenowanym przez człowieka) negatywnie wpływa na ocenę
2. ilość potworów - im więcej potworów występuje w produkcji, tym gorzej jest ona oceniana
3. niepowodzenie w schwytaniu potwora - niepowodzenie negatywnie wpływa na ocenę produkcji

4. zaoferowanie scooby-chrupki przez Kudłatego negatywnie wpływa na ocenę :)

```
columns_to_drop = list(filter(lambda label:
abs(correlation_matrix['imdb'][label]) < 0.2,
correlation_matrix.columns))

correlated_data = data.drop(columns = columns_to_drop)

sb.heatmap(correlated_data.corr(numeric_only = True), annot = True)
```

```
<Axes: >
```



Co ciekawe, ocena widzów wykazuje jedynie bardzo słabe powiązanie z zaangażowaniem społeczności (ilością osób oceniających daną produkcję). Współczynnik korelacji pomiędzy tymi cechami wynosi jedynie około -0.18, co wskazuje na bardzo słabą zależność ujemną. Oznacza to, że wraz ze wzrostem zaangażowania spada ocena

```
correlation_matrix['imdb']['engagement']
```

```
-0.1881782708856488
```

Zgodnie z macierzą korelacji, istnieją pewne cechy, które w silnym i przeciętnym (przyjęta została klasyfikacja Stanisza) stopniu korelują z zaangażowaniem widzów.

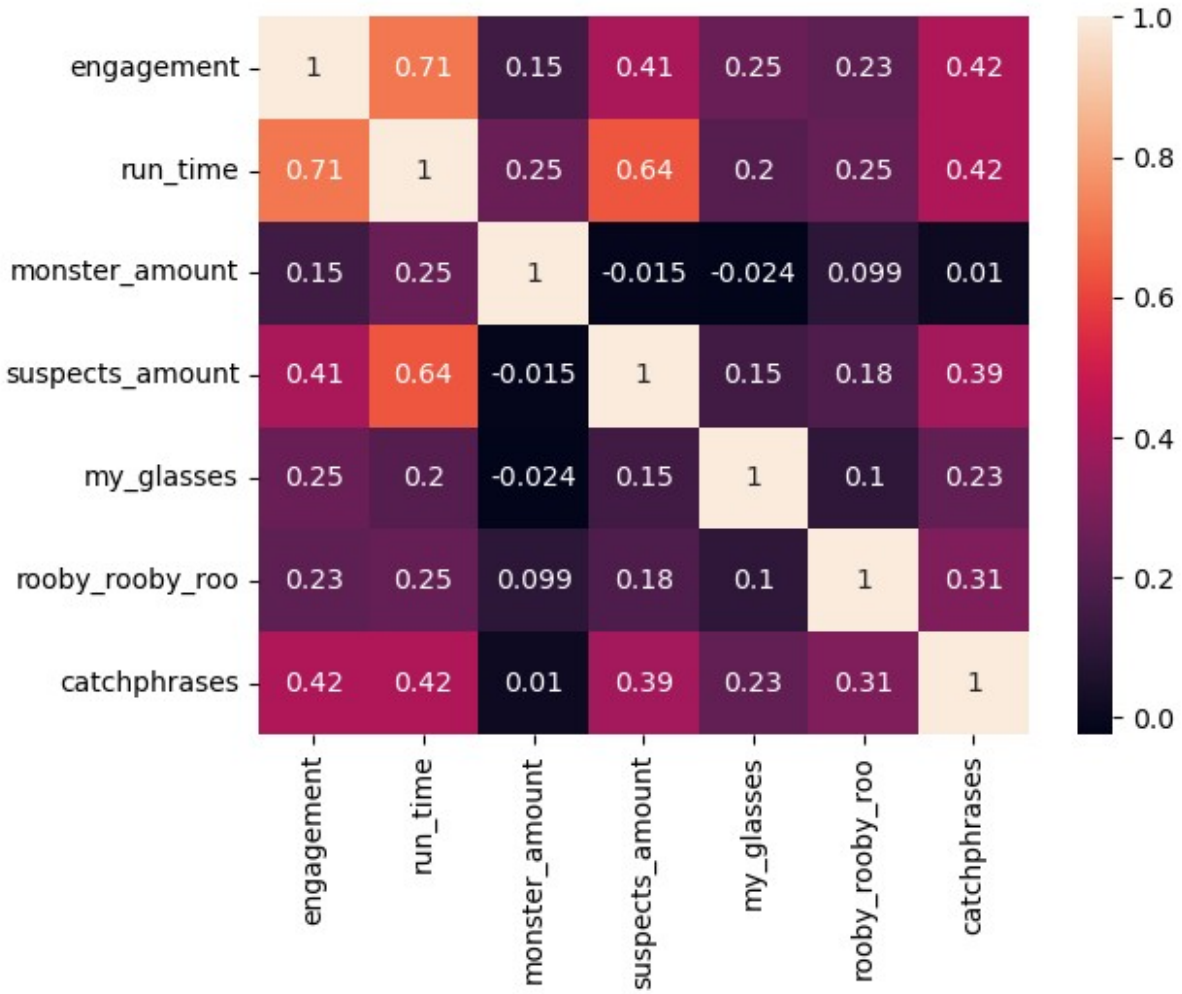
W bardzo wysokim stopniu na zaangażowanie widzów wpływa czas trwania produkcji, wartość wskaźnika wskazuje na bardzo wysoką korelację. Czym dłuższa jest produkcja, tym bardziej społeczność jest zaangażowana w jej ocenianie.

Na zaangażowanie w przeciętnym stopniu wpływają także:

1. ilość podejrzanych - im więcej podejrzanych, tym większe zaangażowanie
2. ilość zwrotów typowych dla poszczególnych bohaterów - im więcek powiedzonek, tym większe zaangażowanie

```
columns_to_drop = list(filter(lambda label:
abs(correlation_matrix['engagement'][label]) < 0.2,
correlation_matrix.columns))

correlated_data = data.drop(columns = columns_to_drop)

sb.heatmap(correlated_data.corr(numeric_only = True), annot = True)
```

```
<Axes: >
```

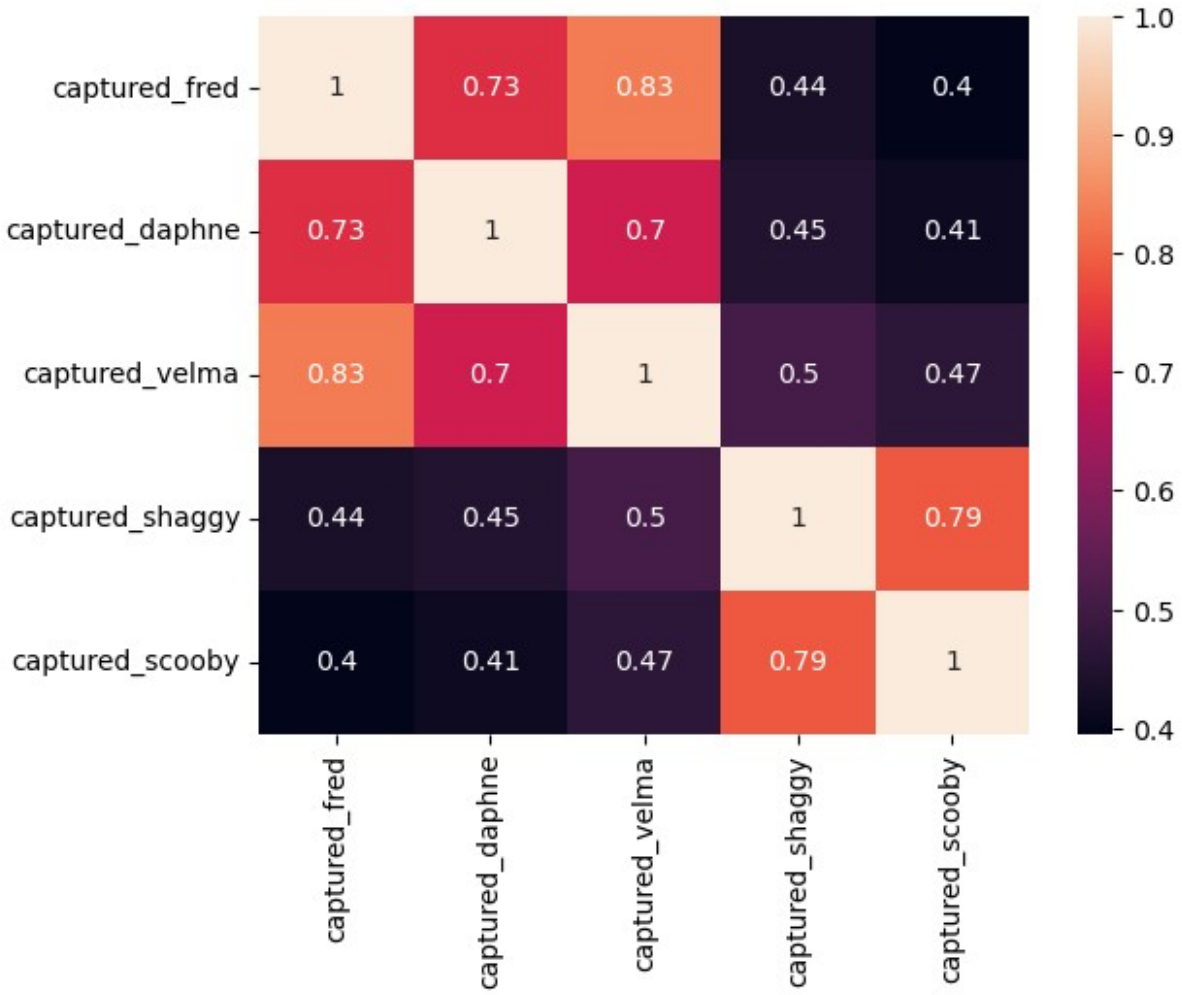


W trakcie analizowania korelacji z oceną i zaangażowaniem widzów, zauważalny jest wyróżniający się fragment heatmapy związany z porywaniem poszczególnych bohaterów przez potwory. Choć nie dotyczy to bezpośrednio określonego przez nas celu opracowania, to jest to pewien zestaw cech, który wykazuje bardzo silną korelację (zgodnie z klasyfikacją Stanisza)

Na poniższym fragmencie heatmapy można dostrzec silne zależności pomiędzy faktem złapania poszczególnych bohaterów. Przykładowo, schwytanie Freda, znacznie mocniej koreluje ze schwytaniem Daphne lub Velmy niż ze schwytaniem Kudłatego lub Scoobiego, schwytanie Kudłatego natomiast bardzo mocno koreluje ze schwytaniem Scoobiego. Dane znacznie pokrywają się z grupami na jakie zazwyczaj dzieliła się grupa młodych detektywów :)

```
columns_to_drop = list(filter(lambda label:
abs(correlation_matrix['captured_daphne'][label]) < 0.3,
correlation_matrix.columns))

correlated_data = data.drop(columns = columns_to_drop)

sb.heatmap(correlated_data.corr(numeric_only = True), annot = True)
```

```
<Axes: >
```



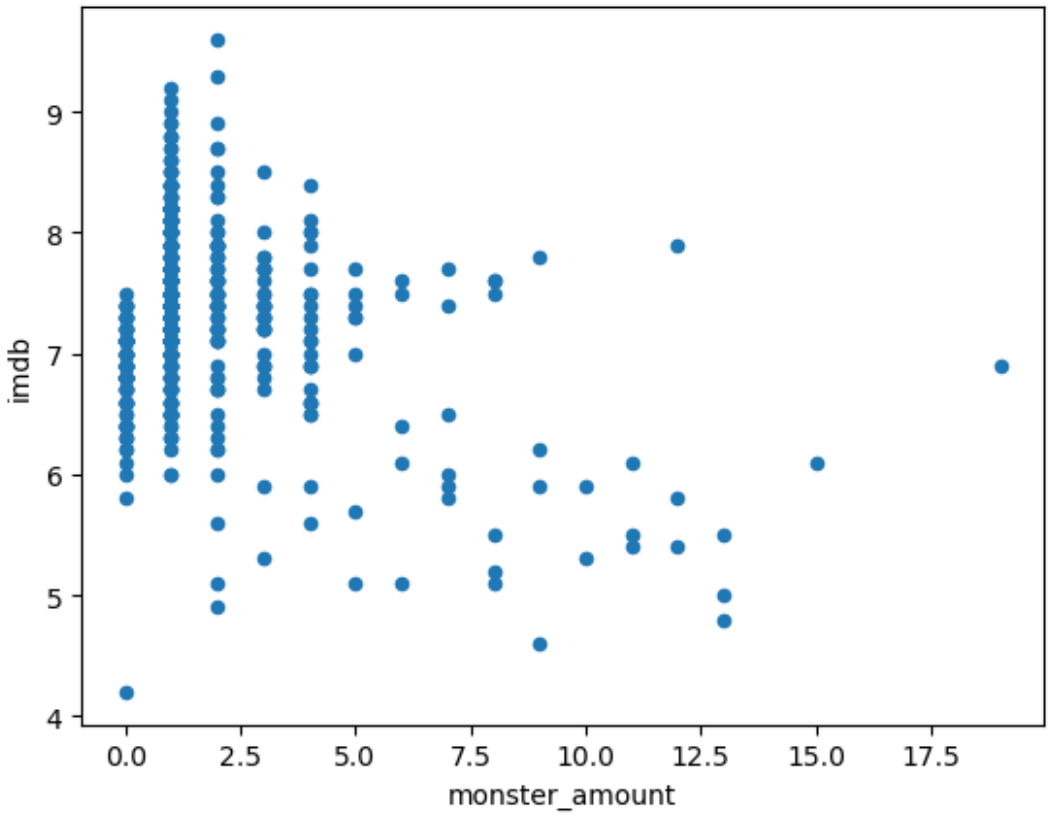
Regresja liniowa

Kiedy wiemy już które zmienne ze sobą korelują, możemy podjąć próbę wyznaczenia zależności pomiędzy poszczególnymi zmiennymi. Do tego celu wykorzystana zostanie regresja liniowa. Ponieważ zmienne nie są mocno skorelowane, dopasowanie prostej może być niedokładne.

Przykładowa regresja liniowa dla zależności zmiennych 'imdb' od 'monster_amount' oraz 'engagement' od 'run_time'

```
data.plot.scatter(x = 'monster_amount', y = 'imdb')
```

```
<Axes: xlabel='monster_amount', ylabel='imdb'>
```



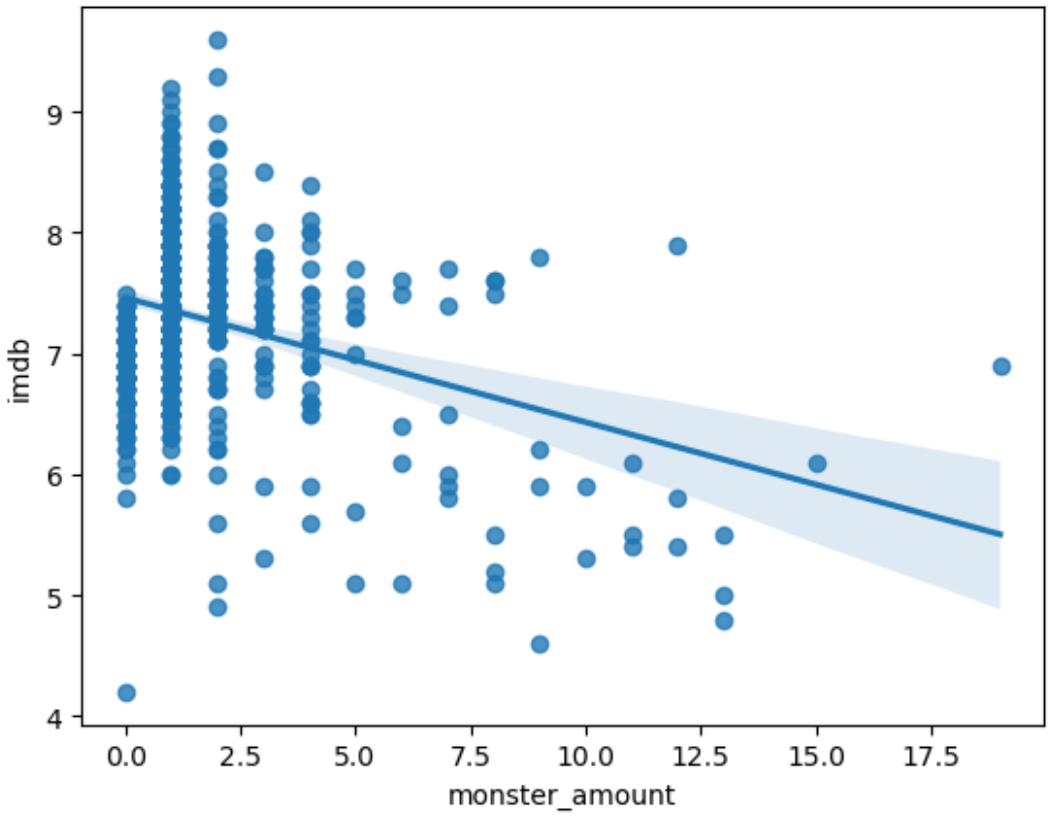

```
xs = data['monster_amount'].values.reshape((-1, 1))
ys = data['imdb'].values

model = LinearRegression(fit_intercept = True).fit(xs, ys)

d, m, b = model.score(xs, ys), model.coef_, model.intercept_
d, m, b
```

```
(0.10112802841315338, array([-0.10307096]), 7.462223184819835)
```

Zgodnie z informacją zwracają przez funkcję score(), jedynie około 10% wyników pokrywa się z przewidywaniem (mimo że jest to cecha najmnocniej skorelowana z 'imdb'). Zgodnie z wyznaczoną funkcją, w przybliżeniu równą $I = -0.1 * MA + 7.47$ wartość oceny stopniowo maleje wraz ze wzrostem ilości antagonistów.

```
sb.regplot(data = data, x = 'monster_amount', y = 'imdb')
```

```
<Axes: xlabel='monster_amount', ylabel='imdb'>
```



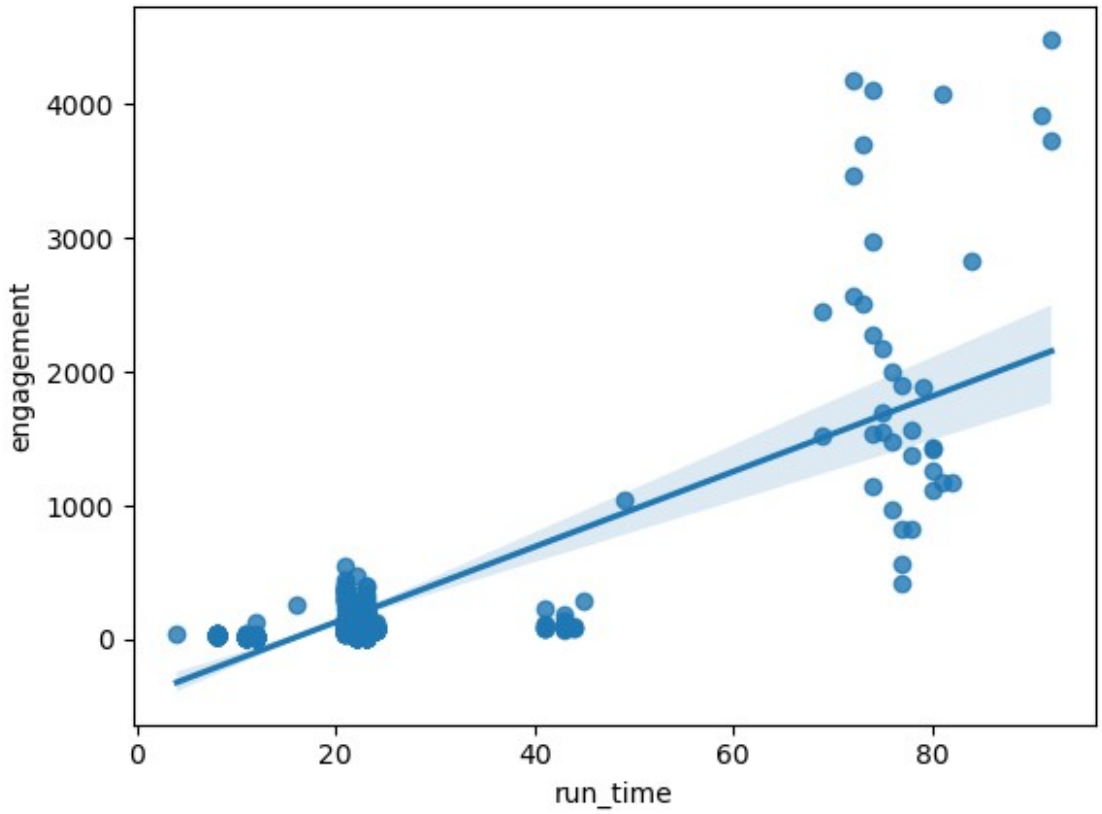
W trakcie badania zależności cechy 'engagement' od 'runtime' zostały odrzucone wartości skrajne, w tym celu usunięto około 1% wartości (rekordy z wartością 'engagement' większą niż 5000). Dzięki temu wskaźnik dopasowania wzrasta o około 50% (z poziomu 0.4 do 0.6), ponieważ wartości te znacząco zaburzały pomiary

```
xs = data[data['engagement'] < 5000]['run_time'].values.reshape((-1, 1))
ys = data[data['engagement'] < 5000]['engagement'].values

model = LinearRegression(fit_intercept = True).fit(xs, ys)

d, m, b = model.score(xs, ys), model.coef_, model.intercept_
d, m, b
```

```
(0.6192639464779743, array([28.12745756]), -432.4370393770009)
```

```
sb.regplot(data = data[data['engagement'] < 5000], x = 'run_time', y = 'engagement')
```

```
<Axes: xlabel='run_time', ylabel='engagement'>
```



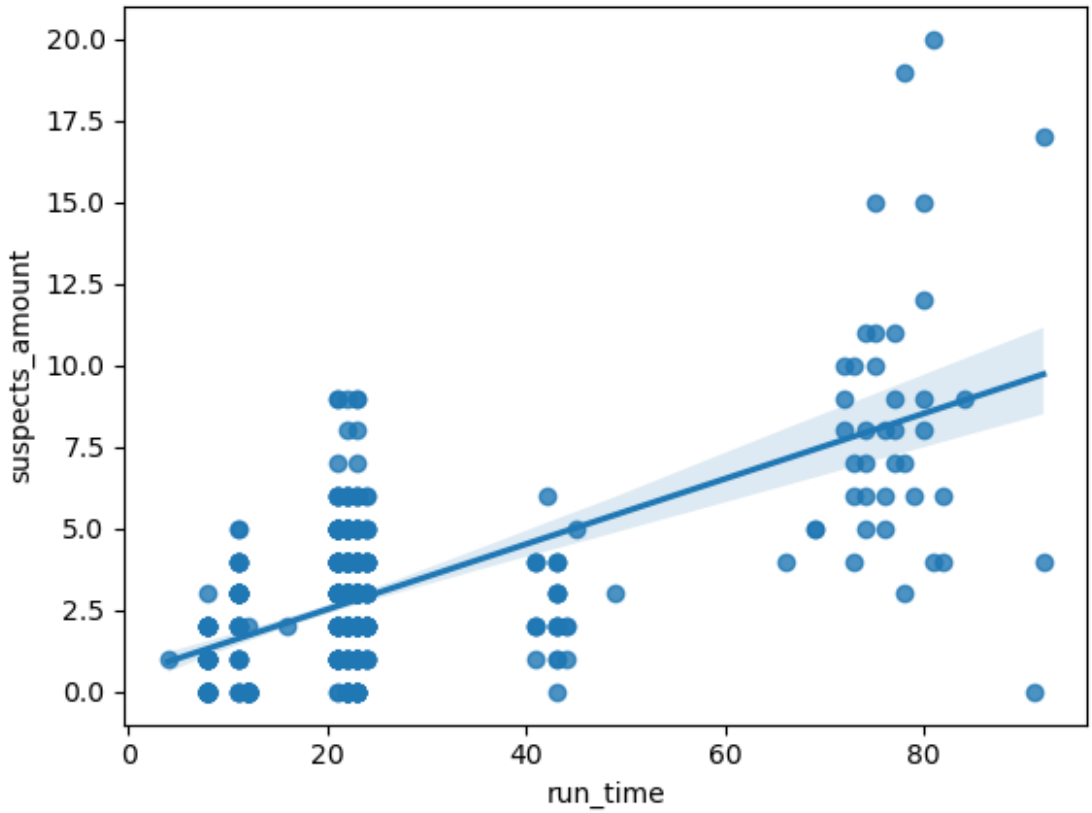
Zbadane zostały także dwie inne przykładowe zależności, które wykazywały się wysokim współczynnikiem korelacji:

1. ilość podejrzanych w zależności od czasu trwania odcinka (dopasowanie prostej na poziomie ~40%)
2. ilość powiedzonek typowych dla bohaterów w zależności od ilości wypowiedzianych 'zoinks' przez Kudłatego (dopasowanie prostej na poziomie ~43%)

```
xs = data['run_time'].values.reshape((-1, 1))
ys = data['suspects_amount'].values

model = LinearRegression(fit_intercept = True).fit(xs, ys)

d, m, b = model.score(xs, ys), model.coef_, model.intercept_
d, m, b
```

```
(0.4071713443162731, array([0.10008515]), 0.5186487157973674)
```

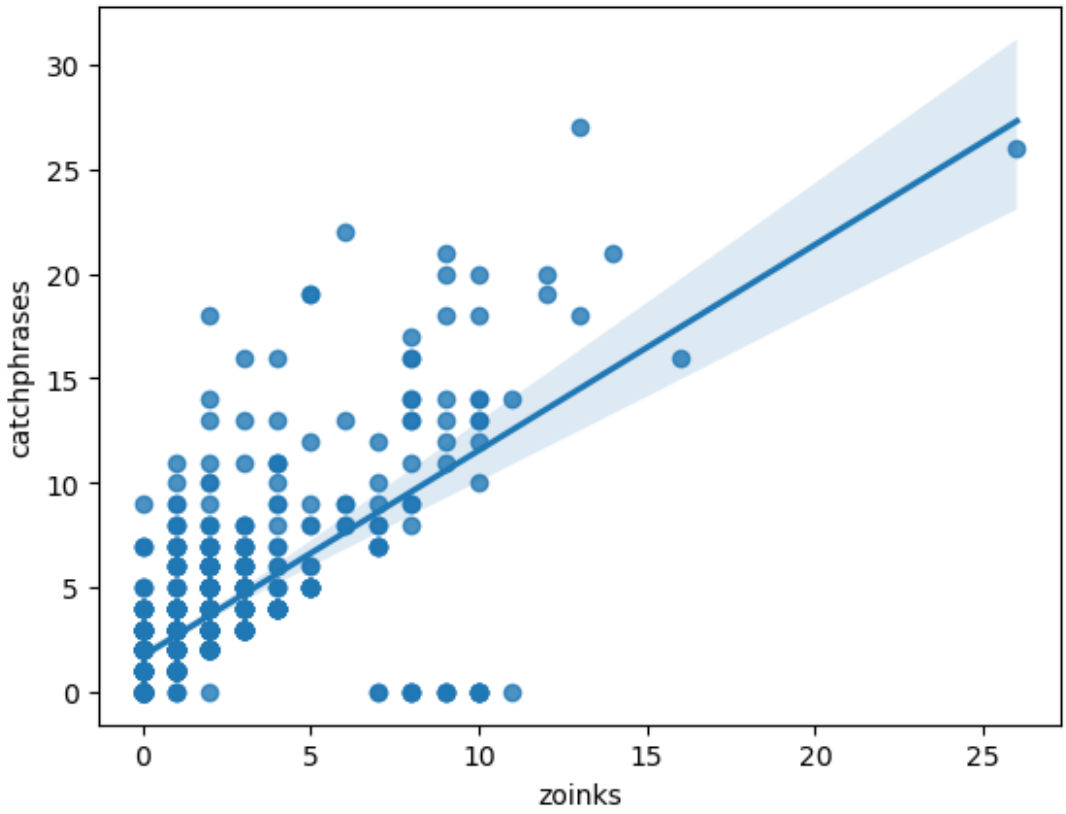
```
ax = sb.regplot(data = data, x = 'run_time', y = 'suspects_amount')
```




```
xs = data['zoinks'].values.reshape((-1, 1))
ys = data['catchphrases'].values

model = LinearRegression(fit_intercept = True).fit(xs, ys)

d, m, b = model.score(xs, ys), model.coef_, model.intercept_
d, m, b
```

```
(0.4344950427045631, array([0.98477726]), 1.7021895728867724)
```

```
ax = sb.regplot(data = data, x = 'zoinks', y = 'catchphrases')
```



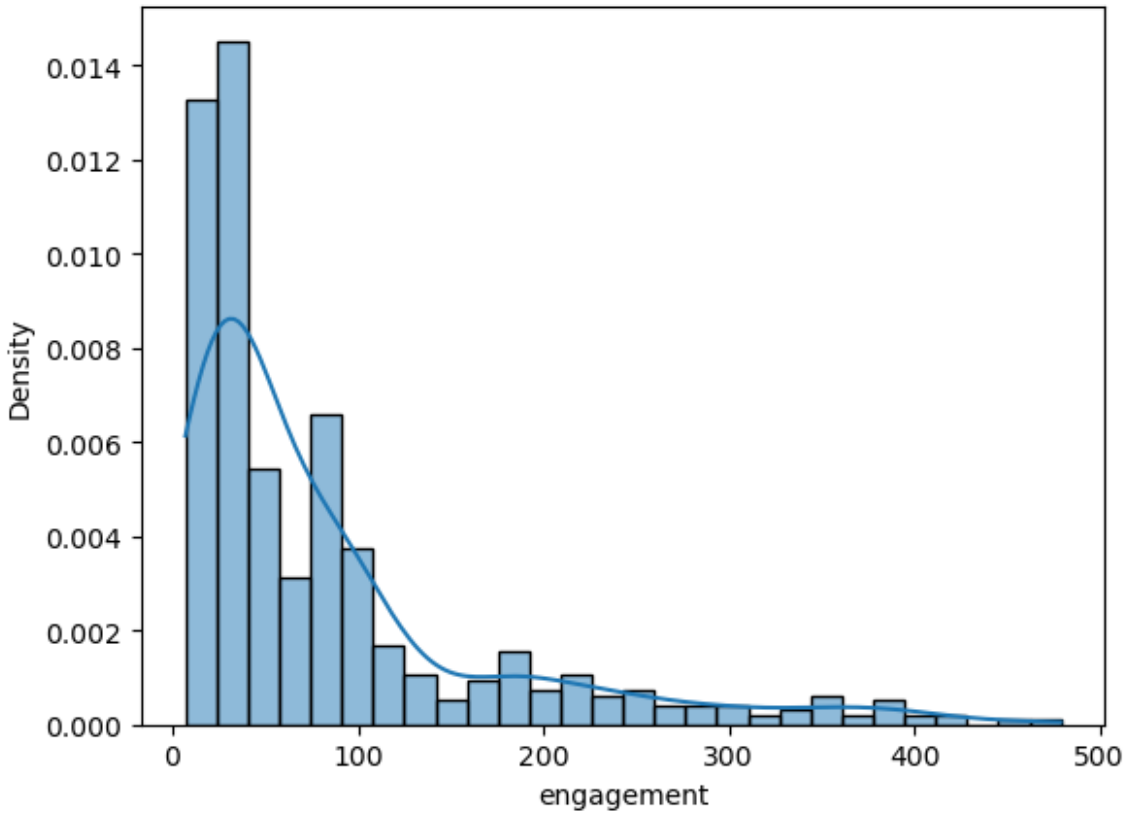
Testy statystyczne i badanie hipotez

W celu lepszej analizy zbioru danych wykonujemy poszczególne testy statystyczne dla cech 'imdb' oraz 'engagement'

Test normalności - badamy czy rozkłady cech 'engagement' oraz 'imdb' są normalne

```
sb.histplot(data[data['engagement'] < 500], x = 'engagement', kde =
True, stat = 'density', label = 'engagement')
```

```
<Axes: xlabel='engagement', ylabel='Density'>
```

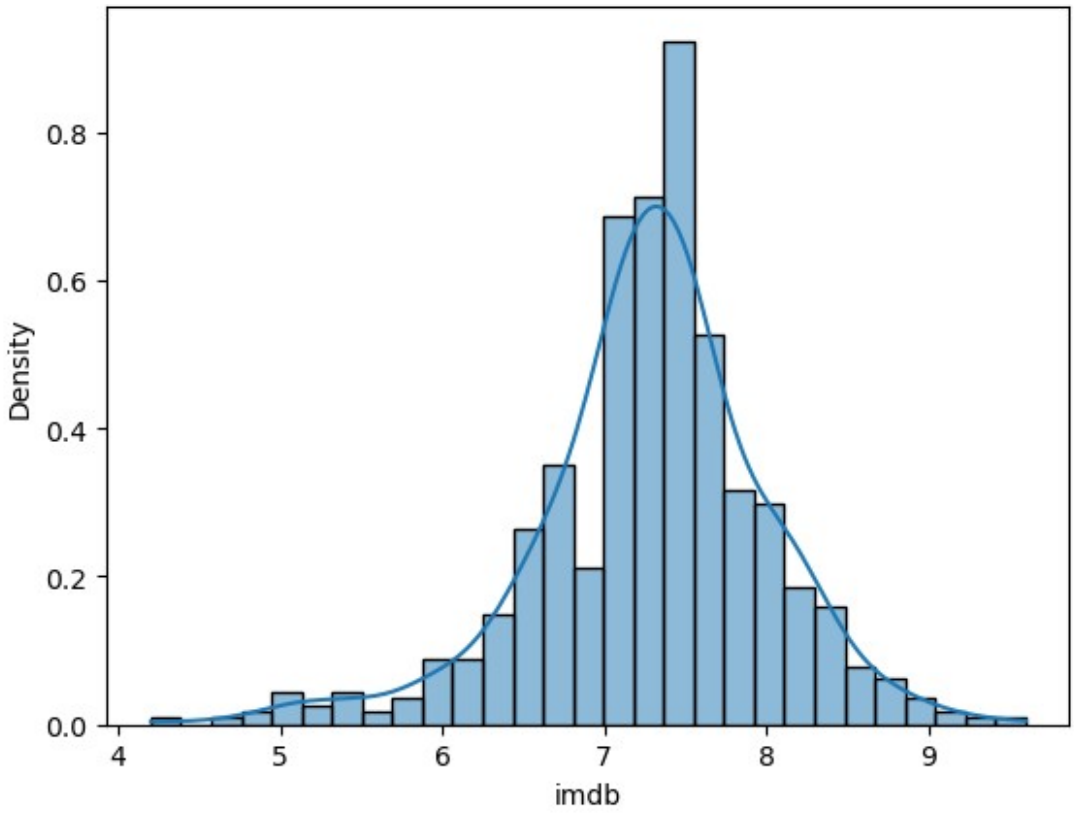



```
ss.normaltest(data['engagement'], nan_policy = 'omit', axis = None)
```

```
NormaltestResult(statistic=684.1424156507336,
pvalue=2.756525557555196e-149)
```

```
sb.histplot(data = data, x = 'imdb', kde = True, stat = 'density',
label = 'engagement')
```

```
<Axes: xlabel='imdb', ylabel='Density'>
```




```
ss.normaltest(data['imdb'], nan_policy = 'omit', axis = None)
```

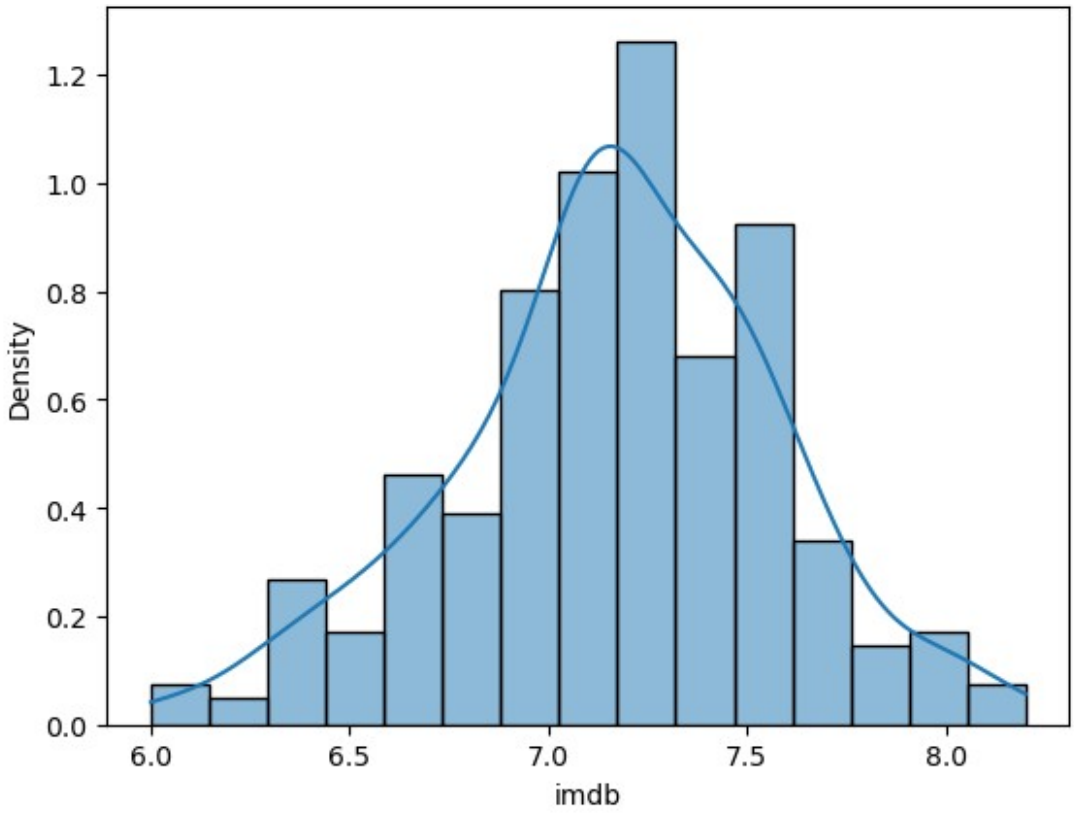
```
NormaltestResult(statistic=55.27041675340213,
pvalue=9.958214935806896e-13)
```

Dla obu zmiennych, obliczona p-wartość jest bardzo bliska zeru, co pozwala nam stwierdzić (odrzucić hipotezę zerową H_0 - rozkład normalny), że rozkłady tych zmiennych nie są normalne

Zostały też przeprowadzone przykładowe testy normalności dla cechy 'imdb' w pewnych podkategoriach 'network'. Wartości zwracane przez testy przeprowadzone dla danych zgromadzonych osobno dla stacji Boomerang oraz ABC nie pozwalają na odrzucenie hipotezy zerowej H_0 - rozkład normalny w danej podkategorii.

```
sb.histplot(data = data[data['network'] == 'ABC'], x = 'imdb', kde =
True, stat = 'density', label = 'engagement')
```

```
<Axes: xlabel='imdb', ylabel='Density'>
```

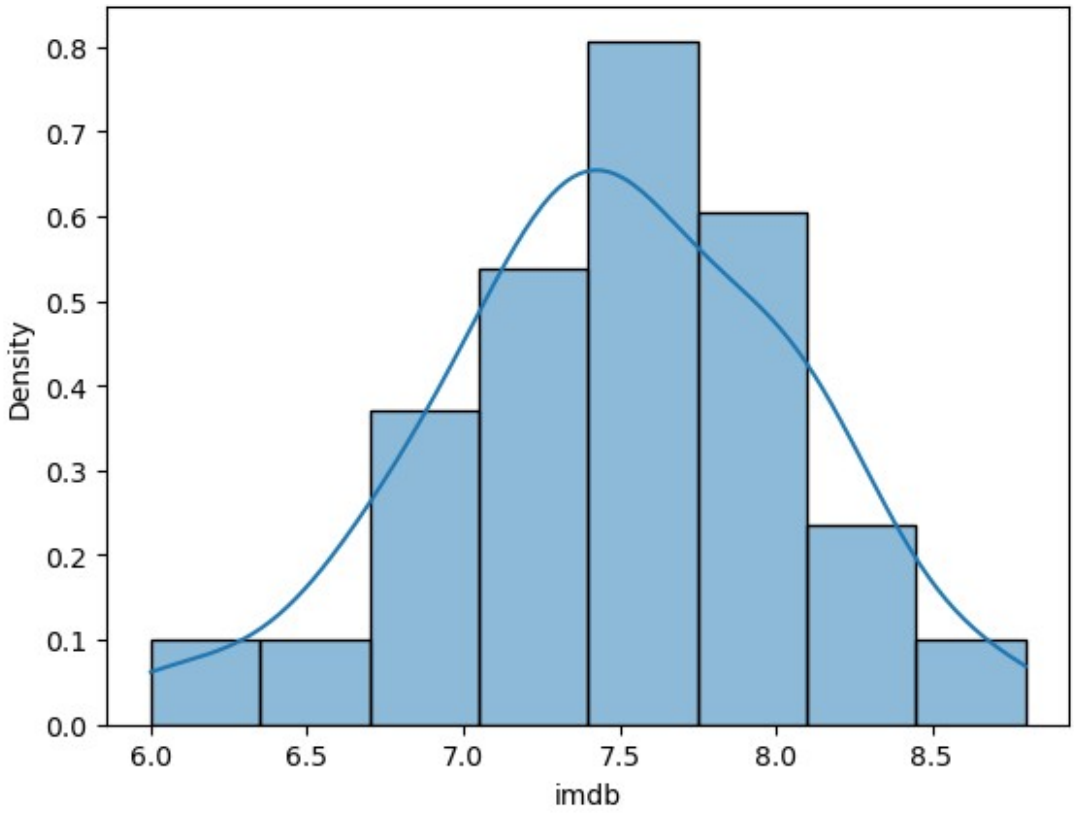



```
ss.normaltest(data[data['network'] == 'ABC']['imdb'], nan_policy =
'omit', axis = None)
```

```
NormaltestResult(statistic=3.207508456797428,
pvalue=0.201139973362446)
```

```
sb.histplot(data = data[data['network'] == 'Boomerang'], x = 'imdb',
kde = True, stat = 'density', label = 'engagement')
```

```
<Axes: xlabel='imdb', ylabel='Density'>
```

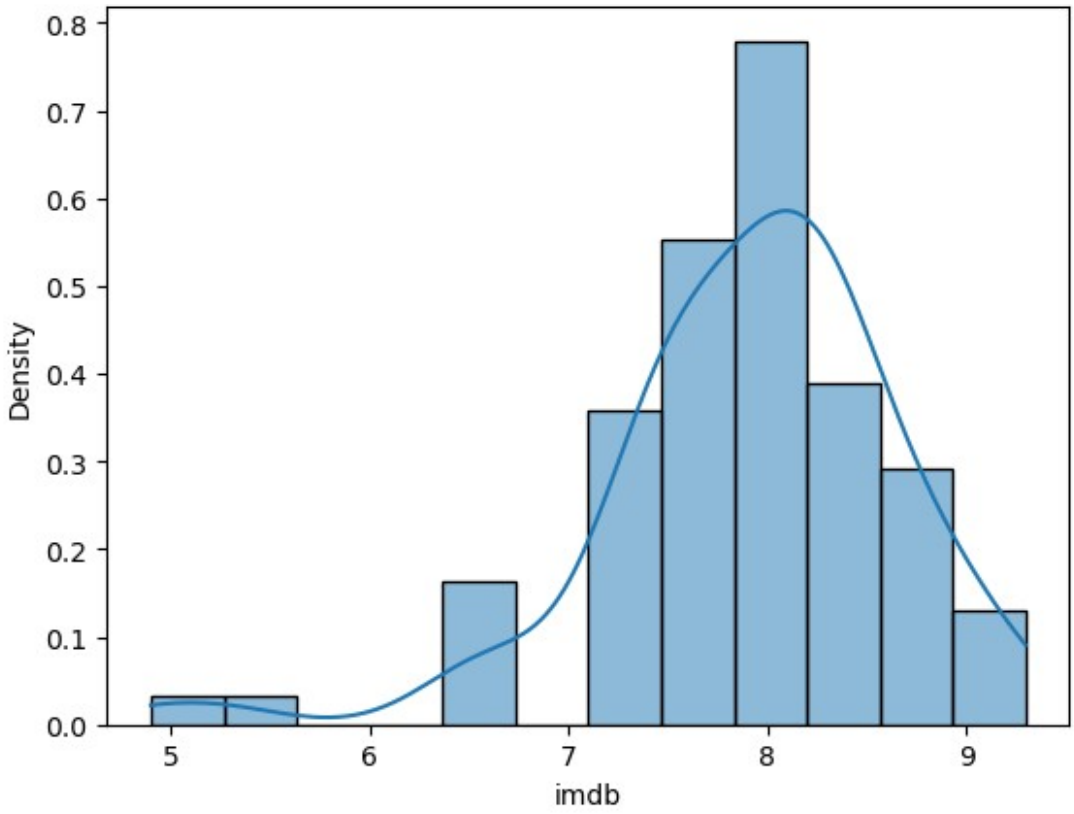



```
ss.normaltest(data[data['network'] == 'Boomerang']['imdb'], nan_policy
= 'omit', axis = None)
```

```
NormaltestResult(statistic=0.8213122644320303,
pvalue=0.6632149506496635)
```

```
sb.histplot(data = data[data['network'] == 'Cartoon Network'], x =
'imdb', kde = True, stat = 'density', label = 'engagement')
```

```
<Axes: xlabel='imdb', ylabel='Density'>
```




```
ss.normaltest(data[data['network'] == 'Cartoon Network']['imdb'],
nan_policy = 'omit', axis = None)
```

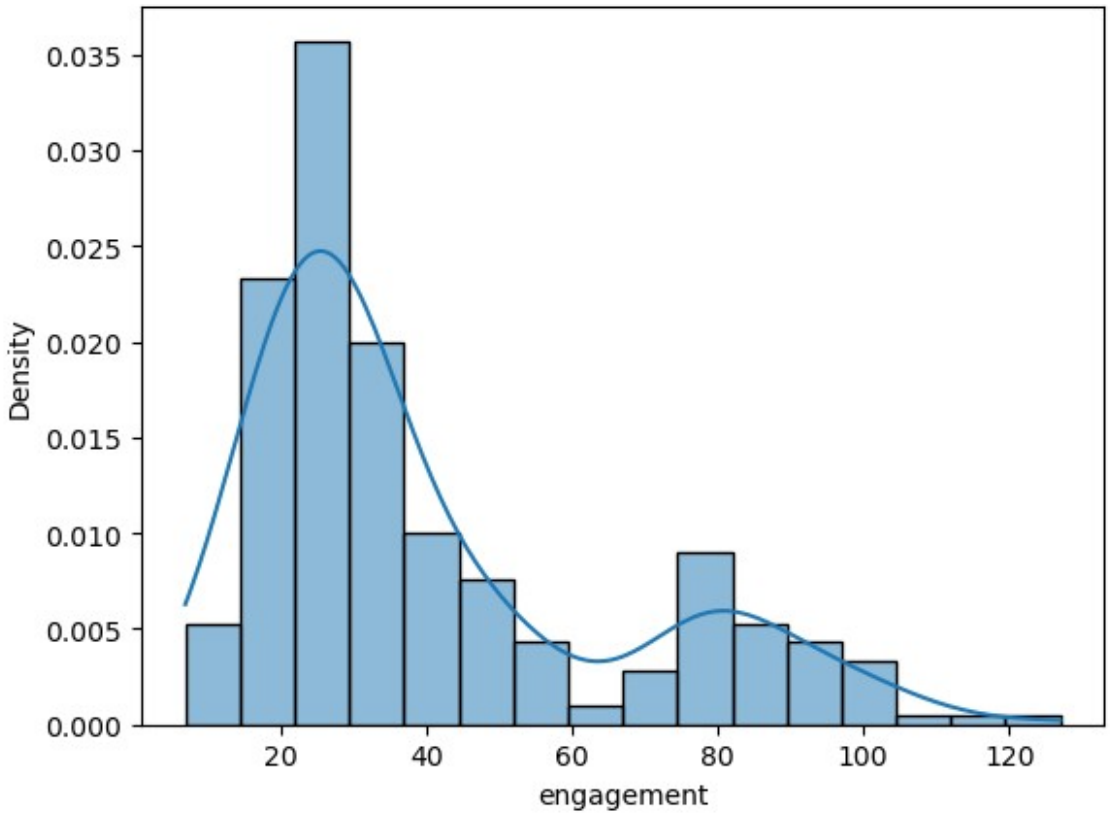
```
NormaltestResult(statistic=29.376823826268517,
pvalue=4.177377836382621e-07)
```

Zostały też przeprowadzone przykładowe testy normalności dla cechy 'engagement' w pewnych podkategoriach 'network'. Wartości zwracane przez testy przeprowadzone dla danych zgromadzonych osobno dla stacji Boomerang oraz ABC pozwalają na odrzucenie hipotezy zerowej H_0 - rozkład normalny w danej podkategorii, ponieważ p-wartości są bardzo bliskie zeru.

```
sb.histplot(data = data[data['network'] == 'ABC'][data['engagement'] <
500], x = 'engagement', kde = True, stat = 'density', label =
'engagement')
```

```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\196895213.py:1:
UserWarning: Boolean Series key will be reindexed to match DataFrame
index.
  sb.histplot(data = data[data['network'] == 'ABC'][data['engagement']
< 500], x = 'engagement', kde = True, stat = 'density', label =
'engagement')
```

```
<Axes: xlabel='engagement', ylabel='Density'>
```




```
ss.normaltest(data[data['network'] == 'ABC'][data['engagement'] < 500]
['engagement'], nan_policy = 'omit', axis = None)
```

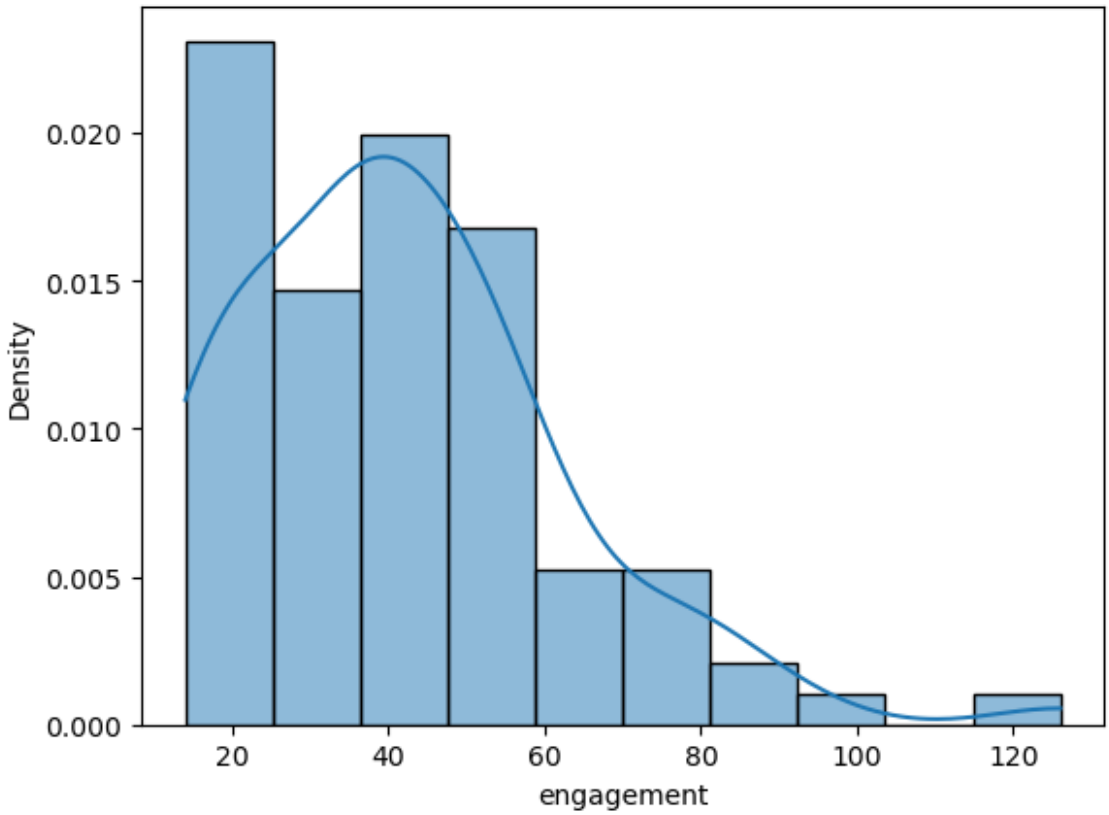
```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\1274314662.py:1:
UserWarning: Boolean Series key will be reindexed to match DataFrame
index.
  ss.normaltest(data[data['network'] == 'ABC'][data['engagement'] <
500]['engagement'], nan_policy = 'omit', axis = None)
```

```
NormaltestResult(statistic=48.250671584246504,
pvalue=3.3304260235562615e-11)
```

```
sb.histplot(data = data[data['network'] == 'Boomerang']
[data['engagement'] < 500], x = 'engagement', kde = True, stat =
'density', label = 'engagement')
```

```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\2766402827.py:1:
UserWarning: Boolean Series key will be reindexed to match DataFrame
index.
  sb.histplot(data = data[data['network'] == 'Boomerang']
[data['engagement'] < 500], x = 'engagement', kde = True, stat =
'density', label = 'engagement')
```

```
<Axes: xlabel='engagement', ylabel='Density'>
```




```
ss.normaltest(data[data['network'] == 'Boomerang'][data['engagement']
< 500]['engagement'], nan_policy = 'omit', axis = None)
```

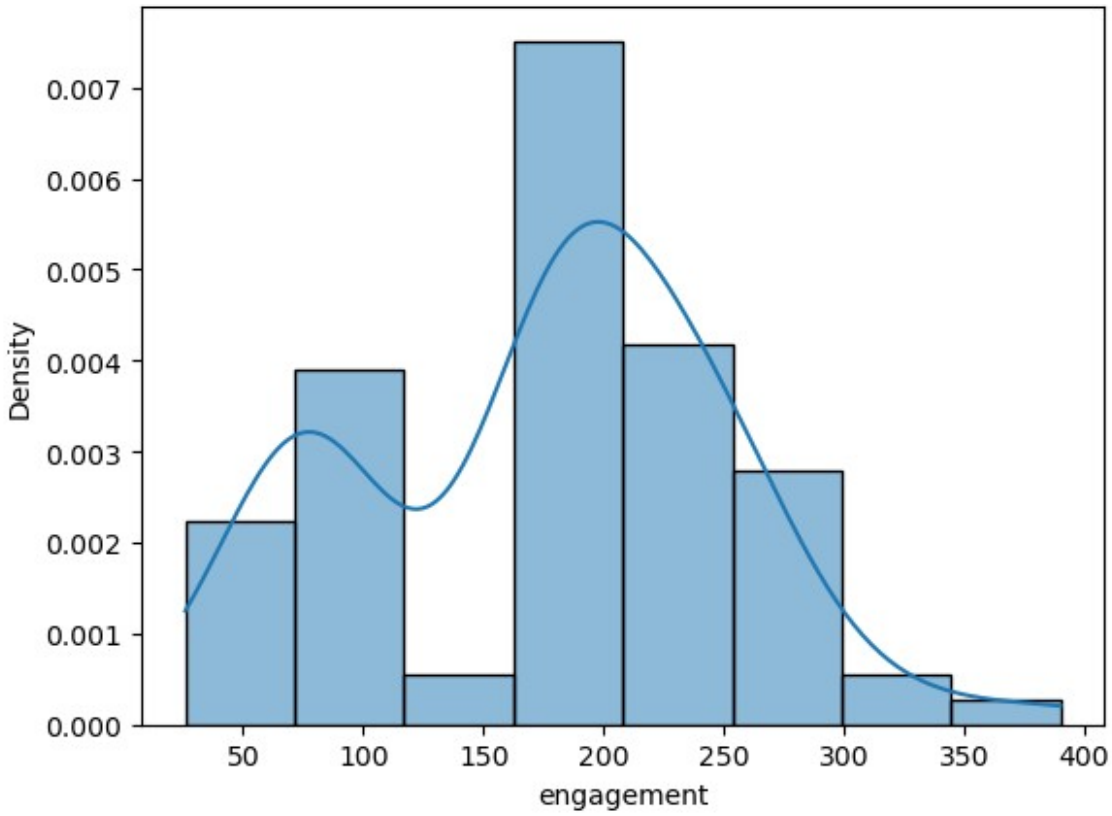
```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\2893371324.py:1:
UserWarning: Boolean Series key will be reindexed to match DataFrame
index.
  ss.normaltest(data[data['network'] == 'Boomerang']
[data['engagement'] < 500]['engagement'], nan_policy = 'omit', axis =
None)
```

```
NormaltestResult(statistic=21.714226710374255,
pvalue=1.926706579457682e-05)
```

```
sb.histplot(data = data[data['network'] == 'Cartoon Network']
[data['engagement'] < 500], x = 'engagement', kde = True, stat =
'density', label = 'engagement')
```

```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\3730040760.py:1:
UserWarning: Boolean Series key will be reindexed to match DataFrame
index.
  sb.histplot(data = data[data['network'] == 'Cartoon Network']
[data['engagement'] < 500], x = 'engagement', kde = True, stat =
'density', label = 'engagement')
```

```
<Axes: xlabel='engagement', ylabel='Density'>
```




```
ss.normaltest(data[data['network'] == 'Cartoon Network']
[data['engagement'] < 500]['engagement'], nan_policy = 'omit', axis =
None)
```

```
C:\Users\mateu\AppData\Local\Temp\ipykernel_3880\1780968463.py:1:
UserWarning: Boolean Series key will be reindexed to match DataFrame
index.
  ss.normaltest(data[data['network'] == 'Cartoon Network']
[data['engagement'] < 500]['engagement'], nan_policy = 'omit', axis =
None)

NormaltestResult(statistic=0.30813393316441023,
pvalue=0.8572146143563755)
```

Jako ostatni element naszej analizy danych, wykonamy przykładowe t-testy dla średniej oraz wariancji dla zmiennej 'imdb'. T-testy dla zmiennej 'engagement' zostaly celowo pominięte, jako że wykres tej zmiennej nawet wizualnie nie przypomina rozkładu normalnego (oprócz stacji Cartoon Network - według testu normalności nie możemy odrzucić hipotezy, że jest to w tym przypadku rozkład normalny)

1. Równość średniej w próbce i w całej populacji

$H_0 : m_0 = m$

$H_1 : m_0 \neq m$

Jako kolejne próbki przyjmiemy produkcje emitowane przez stacje ABC, Boomerang, Cartoon Network oraz losowo wybrane próbki pełnej populacji.

```
abc_data = data[data['network'] == 'ABC']['imdb']

population_mean = 7.29

t, p = ss.ttest_1samp(abc_data, population_mean)
t, p
```

```
(-5.08136989660142, 6.858307181518568e-07)
```

```
b_data = data[data['network'] == 'Boomerang']['imdb']

population_mean = 7.29

t, p = ss.ttest_1samp(b_data, population_mean)
t, p
```

```
(3.0148647850977714, 0.003399339154011681)
```

```
cn_data = data[data['network'] == 'Cartoon Network']['imdb']

population_mean = 7.29

t, p = ss.ttest_1samp(cn_data, population_mean)
t, p
```

```
(7.619362163310269, 3.706810291652266e-11)
```

```
rnd_data = data['imdb'].sample(n = 100)

population_mean = 7.29

t, p = ss.ttest_1samp(rnd_data, population_mean)
t, p
```

```
(-0.6928114042300255, 0.4900499475452196)
```

```
rnd_data = data['imdb'].sample(n = 30)

population_mean = 7.29

t, p = ss.ttest_1samp(rnd_data, population_mean)
t, p
```

```
(0.5748295003073361, 0.5698396798842545)
```

Uzyskane wyniki dla poszczególnych stacji (p_wartości bliskie zeru, żadna nie przekracza nawet 0.05) sugerują odrzucenie hipotezy zerowej H_0. Można założyć prawdziwość hipotezy H_1 oraz przyjąć, że średnie poszczególnych próbek różnią się od średniej dla pełnej populacji.

Dla losowo wybranych próbek, wysoka p_wartość nie sugeruje odrzucenia hipotezy zerowej.

1. Równość średniej w poszczególnych próbkach

$H_0 : m_1 = m_2$

$H_1 : m_1 \neq m_2$

Jako kolejne próbki przyjmiemy produkcje emitowane przez stacje ABC, Boomerang, Cartoon Network oraz losowo wybrane próbki pełnej populacji.

```
abc_data = data[data['network'] == 'ABC']['imdb']
b_data = data[data['network'] == 'Boomerang']['imdb']

t, p = ss.ttest_ind(abc_data, b_data)
t, p
```

```
(-5.590132927027362, 4.457470582668716e-08)
```

```
abc_data = data[data['network'] == 'ABC']['imdb']
cn_data = data[data['network'] == 'Cartoon Network']['imdb']

t, p = ss.ttest_ind(abc_data, cn_data)
t, p
```

```
(-11.83759825294803, 1.4605826504973318e-27)
```

```
abc_data = data[data['network'] == 'ABC']['imdb']
b_data = data[data['network'] == 'Boomerang']['imdb']

t, p = ss.ttest_ind(abc_data, cn_data)
t, p
```

```
(-11.83759825294803, 1.4605826504973318e-27)
```

Uzyskane wyniki dla poszczególnych par stacji (p_wartości bliskie zeru, żadna nie przekracza nawet 0.05) sugerują odrzucenie hipotezy zerowej H_0. Można założyć prawdziwość hipotezy H_1 oraz przyjąć, że średnie poszczególnych próbek różnią się pomiędzy sobą.

1. Równość wariancji w próbce i całej populacji

$H_0 : \sigma_1 = \sigma$

$H_1 : \sigma_1 \neq \sigma$

Jako kolejne próbki przyjmiemy produkcje emitowane przez stacje ABC, Boomerang, Cartoon Network oraz losowo wybrane próbki pełnej populacji.

```
abc_data = data[data['network'] == 'ABC']['imdb']

t, p = ss.levene(abc_data, data['imdb'])
t, p
```

```
(45.690988737248745, 2.5024910538498836e-11)
```

```
b_data = data[data['network'] == 'Boomerang']['imdb']

t, p = ss.levene(b_data, data['imdb'])
t, p
```

```
(1.3752438534803142, 0.24131532128174424)
```

```
cn_data = data[data['network'] == 'Cartoon Network']['imdb']

t, p = ss.levene(cn_data, data['imdb'])
t, p
```

```
(0.08628516346147656, 0.7690419666684933)
```

```
rnd_data = data['imdb'].sample(n = 30)

t, p = ss.levene(rnd_data, data['imdb'])
t, p
```

```
(0.7345500689092708, 0.39173360496984755)
```

Uzyskane wyniki dla stacji 'ABC' (p_wartości bliska zeru) sugerują odrzucenie hipotezy zerowej H_0. Można założyć prawdziwość hipotezy H_1 oraz przyjąć, że wariancja dla stacji 'ABC' różni się od wariancji dla pełnej populacji.

Uzyskane wyniki dla stacji 'Boomerang' oraz 'Cartoon Network' nie pozwalają na odrzucenie hipotezy zerowej, ponieważ p-wartość jest odpowiednio wysoka (przekracza 0.2 dla stacji 'Boomerang', a dla stacji 'Cartoon Network' wynosi aż 0.77)

Dla losowo wybranej próbki, wysoka p-wartość również nie sugeruje odrzucenia hipotezy zerowej.

1. Równość wariancji w poszczególnych próbkach

$H_0 : \sigma_1 = \sigma_2$

$H_1 : \sigma_1 \neq \sigma_2$

Jako kolejne próbki przyjmiemy produkcje emitowane przez stacje ABC, Boomerang, Cartoon Network.

```
abc_data = data[data['network'] == 'ABC']['imdb']
b_data = data[data['network'] == 'Boomerang']['imdb']

t, p = ss.levene(abc_data, b_data)
t, p
```

```
(16.67013539519269, 5.4712885596956534e-05)
```

```
abc_data = data[data['network'] == 'ABC']['imdb']
cn_data = data[data['network'] == 'Cartoon Network']['imdb']
```

```
t, p = ss.levene(abc_data, cn_data)
t, p
```

```
(28.903313930722245, 1.3647034379335253e-07)
```

```
b_data = data[data['network'] == 'Boomerang']['imdb']
cn_data = data[data['network'] == 'Cartoon Network']['imdb']

t, p = ss.levene(b_data, cn_data)
t, p
```

```
(1.4080392603618523, 0.23706747978094497)
```

Podsumowanie

Wbrew oczekiwaniom, jednoznaczna odpowiedź na pytanie o aspekty czyniące produkcje z uniwersum Scooby-Doo atrakcyjnymi dla widza nie jest możliwa. Zmienne dotyczące oceny użytkowników oraz rozgłosu i zaangażowania poszczególnych filmów w większości przypadków nie korelują w znacznym stopniu z innymi, w związku z czym ciężko jest przewidzieć ocenę oraz zaangażowanie społeczności. W trakcie analizy danych oraz ich reprezentacji graficznej na wykresach, zauważone zostały cechy wpływające na ocenę oraz zaangażowanie w różnym stopniu, takie jak:

1. Zaangażowanie oraz popularność:
   - wytwórnia
   - producent
   - długość produkcji
   - ilość podejrzanych
   - ilość użytych powiedzonek
2. Ocena produkcji:
   - ilość potworów
   - występowanie faktycznego potwora
   - niepowodzenie śledztwa
   - dobór aktorów (choć niektórzy aktorzy wzięli udział w produkcjach zbyt mało razy, by móc mówić o reprezentatywnych danych)

Ze względu na słabą korelację pomiędzy oceną a zaangażowaniem, ciężko stwierdzić w jaki sposób zachowałyby się poszczególne wskaźniki przy próbie uwzględnienia wszystkich powyższych faktów. W celu lepszego zbadania zależności zaangażowania oraz oceny od wielu zmiennych konieczne jest obliczenie regresji dla wielu zmiennych.