NEURAL NETWORK FOR MOVIE BOX OFFICE PREDICTION

Final Capstone

Jupyter Notebook

Presentation by Kyle Millar Corliss for Thinkful Data Science Program

How much money will a movie make its opening weekend?

Issues / Motivation:

1. Forecasting box office is a challenging problem with many variables relevant to the analysis

2. Existing consumer interest measurement methods exist but do not match to box office
    1. Consumer survey “first choice” metric provided 3 weeks in advance

3. Large portion (15-20%) of a movie’s marketing budget spent on digital marketing

Proposal

1. Capture a movie's "digital footprint" and scrape box office information from public sources

2. Determine important inputs, design additional features, identify structure in the data, and where necessary conduct clustering and dimension reduction techniques

3. Build neural network to predict opening weekend box office performance and movie multiple
    1. Compare to other regression models (LR, RFR)

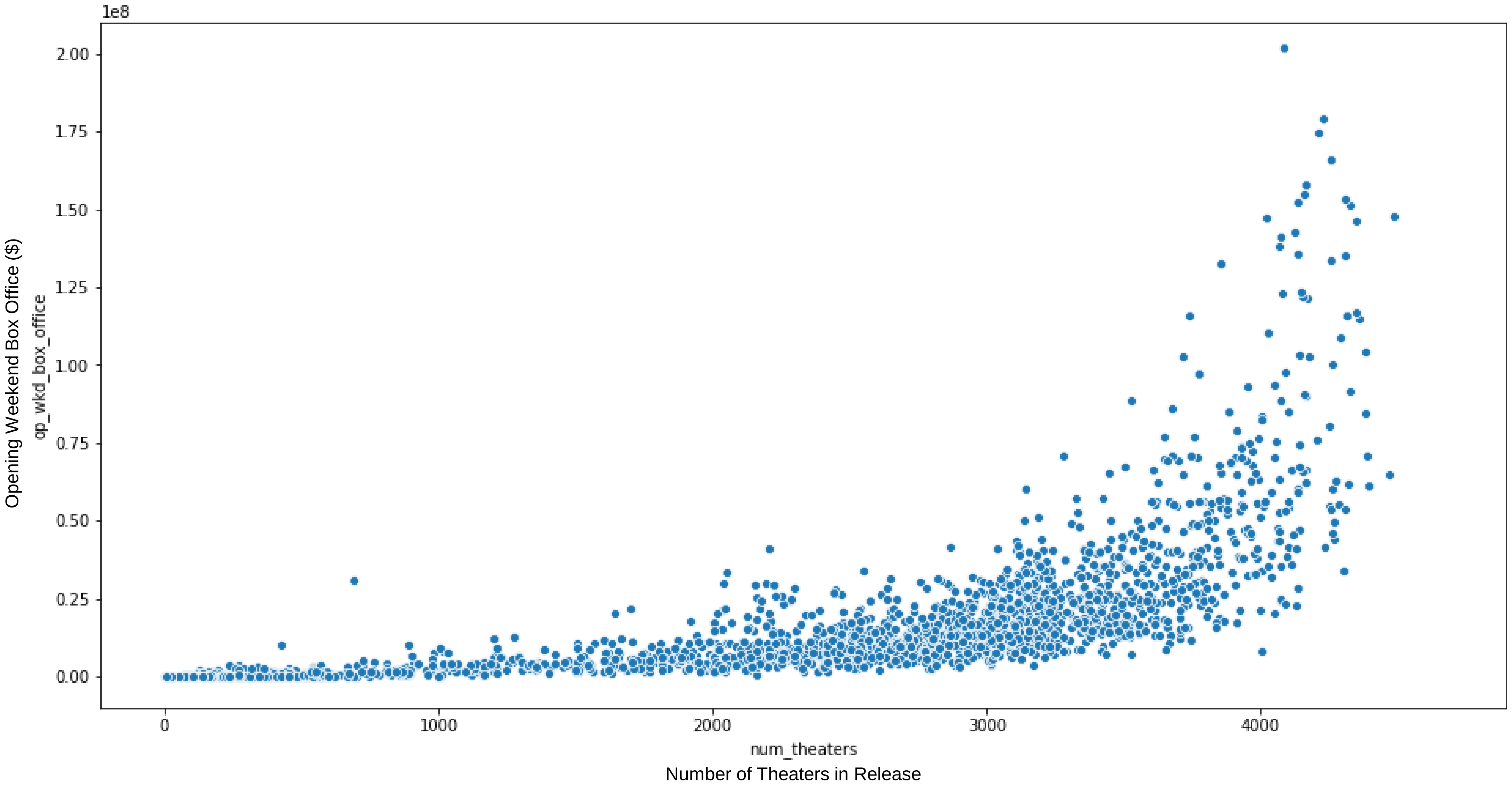
Data sources

Source	Raw Data	Data Rate Limits	Data Collection Duration
TheMovieDB API	- Genres - Keywords - Production cos./countries - Actor credits - Popularity *(link to YouTube trailers)*	40 requests per every 10 seconds	<1 hour
Youtube API	- Trailer video stats: i. Views ii. Likes iii. Comments iv. Dislikes	10k units per day (each trailer costs 3 units)	3 days
Box Office Mojo	- Box office metrics - Distributor - Rating - Number of theaters	--	<1 hour
RottenTomatoes	- Critic scores - Critic counts ~~- Audience scores~~	--	<1 hour
Metacritic	- Critic scores - Critic counts	--	<1 hour

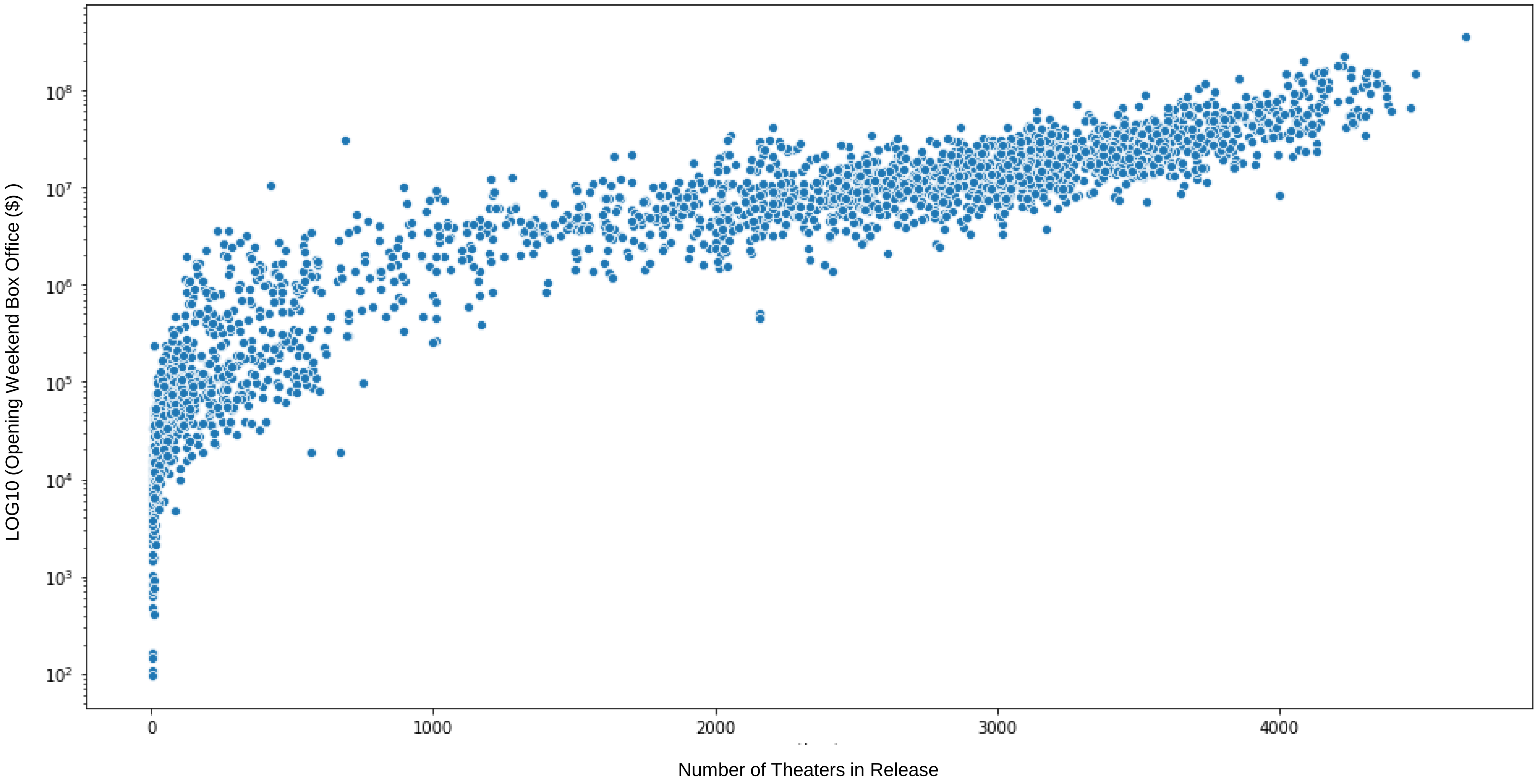
NOTE: Additional data sources were considered (Twitter, Facebook, Google search data), but were prohibitively expensive (Twitter), inaccessible (Facebook), or not scalable (Google).

ANALYSIS / FEATURE ENGINEERING

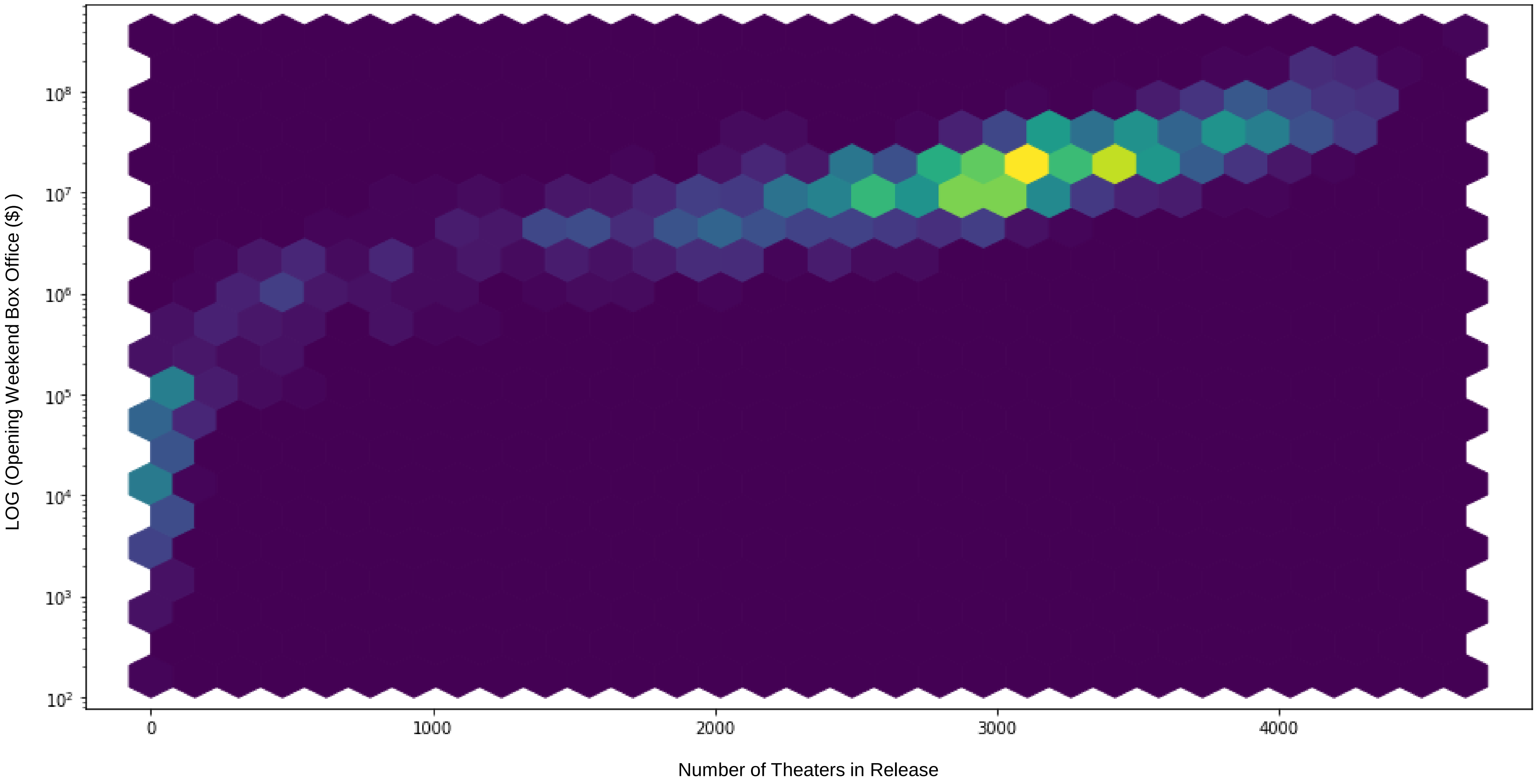
Number of theaters released opening weekend



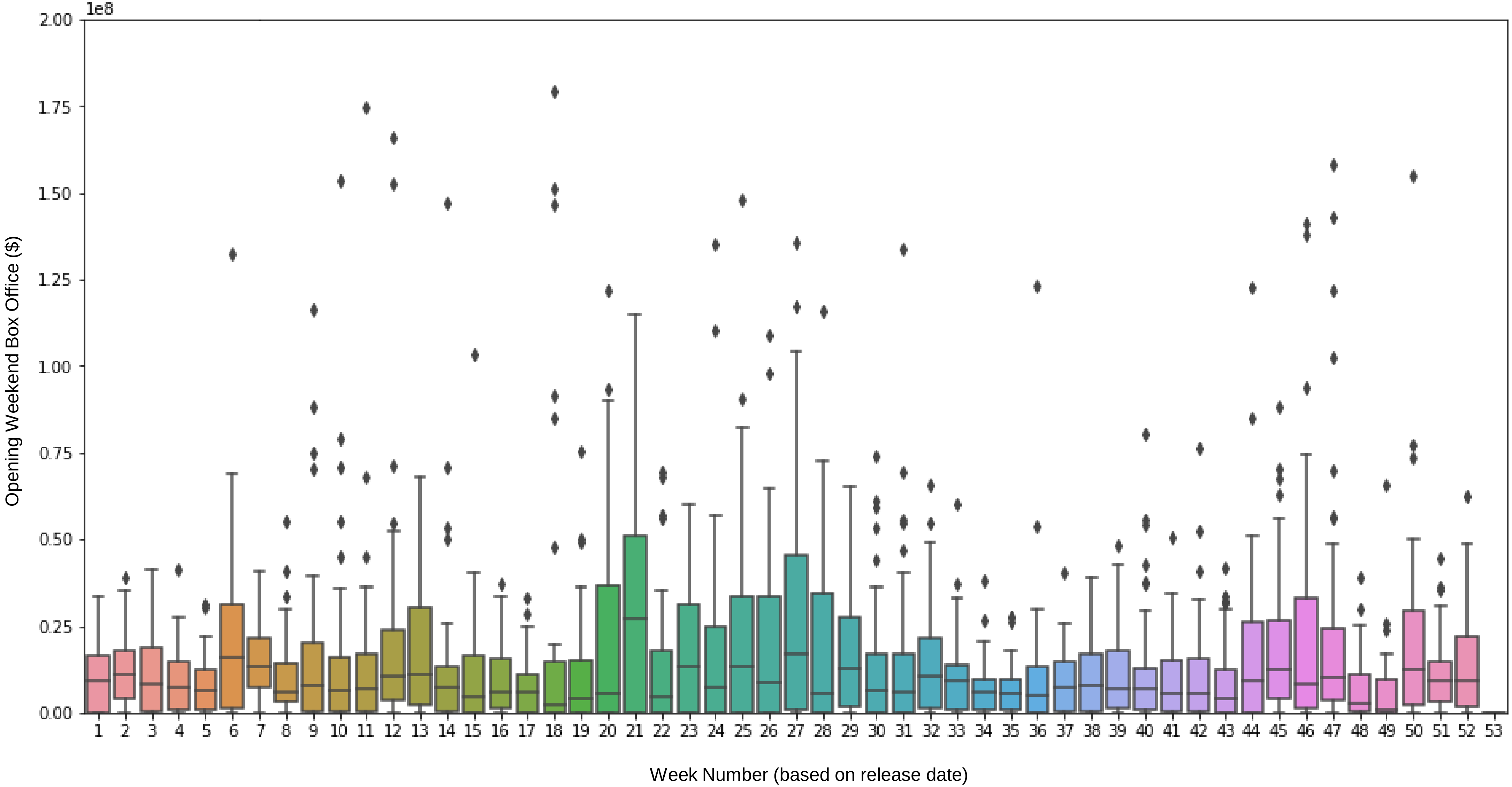
Number of theaters versus log of opening weekend



Density of number of theaters versus log of opening weekend



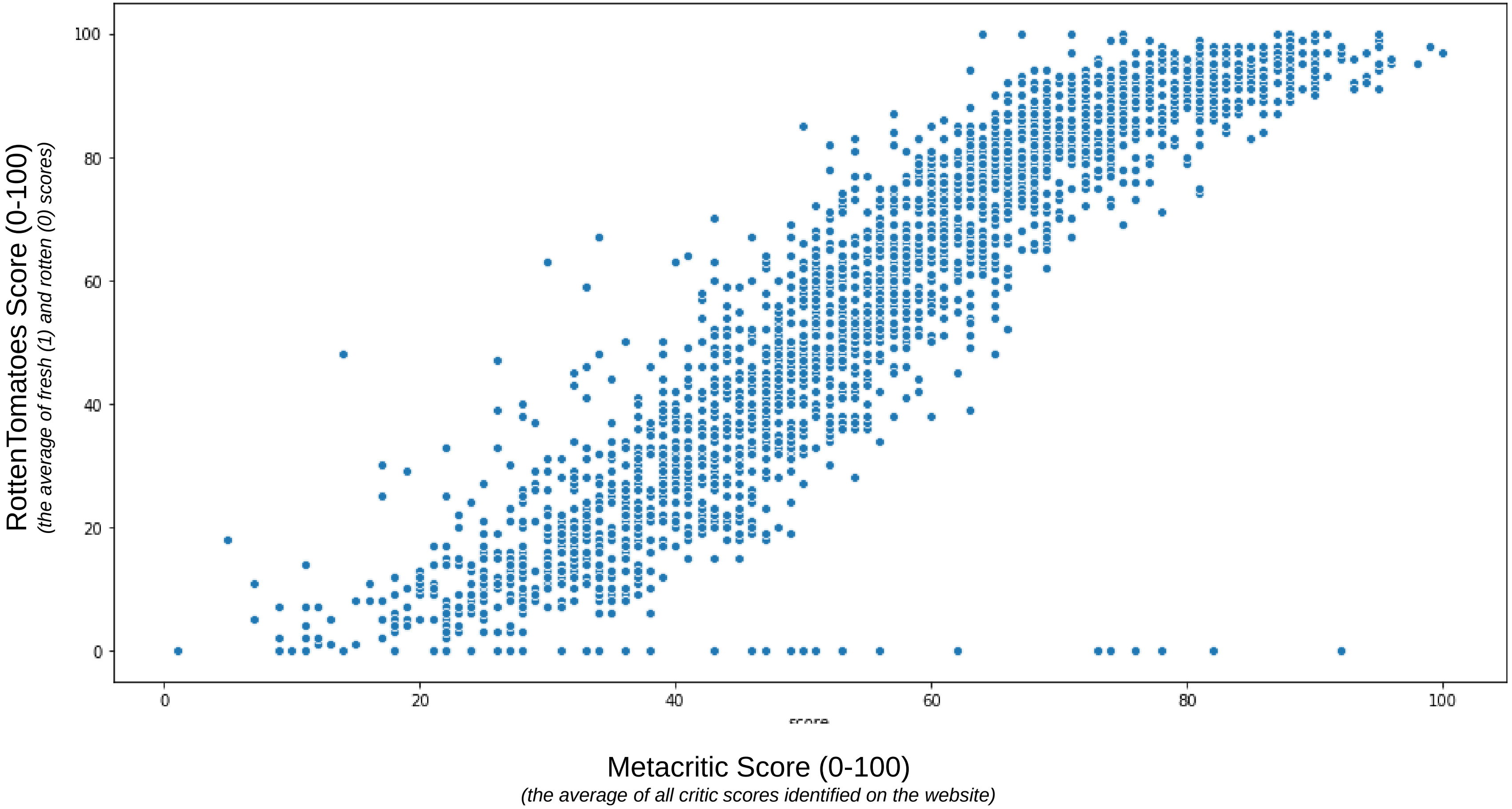
Capture movie seasonality differences via movie release date



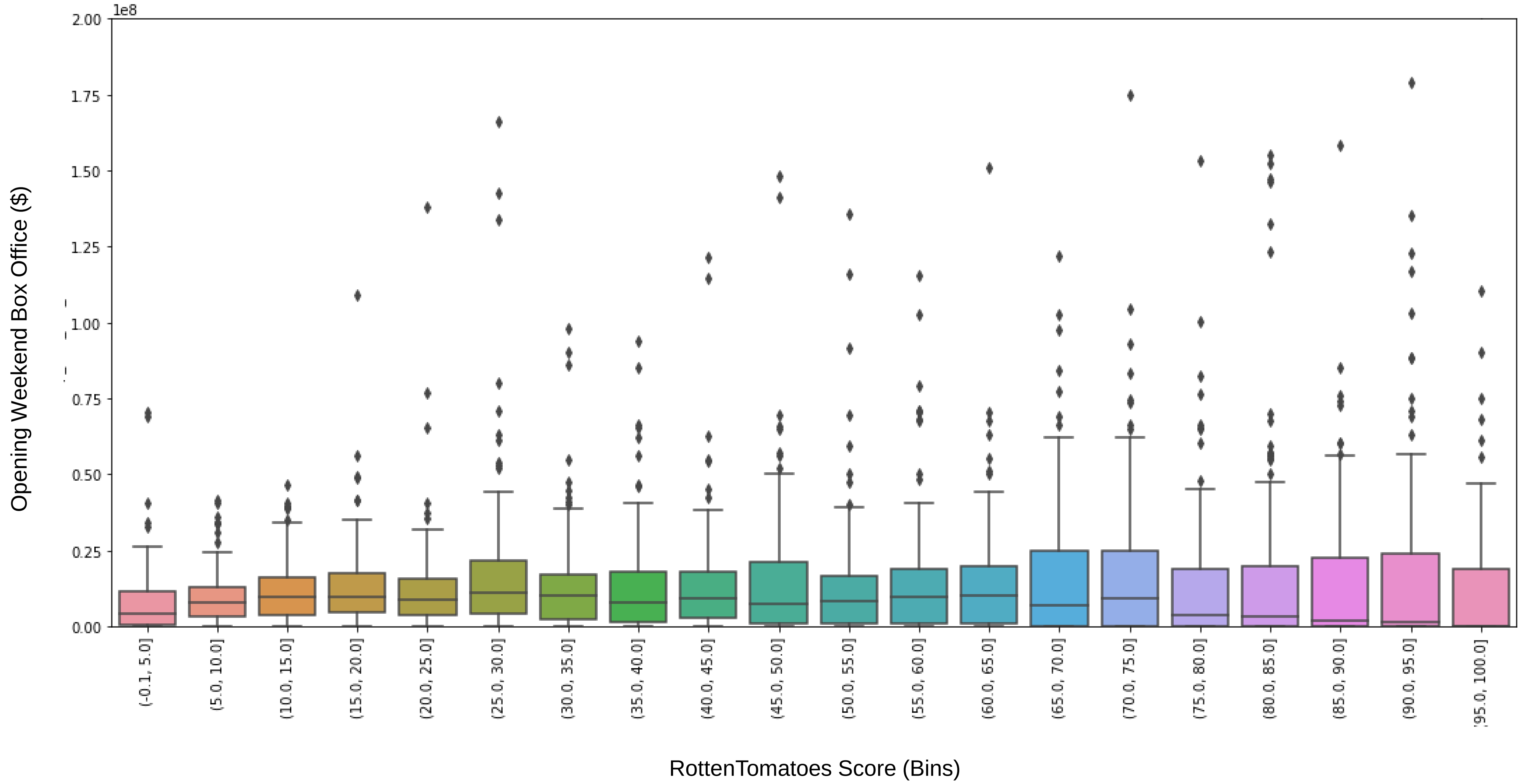
RottenTomatoes vs. Metacritic critic scores

Strong relationship between the two variables, with a slight S-curve identified.

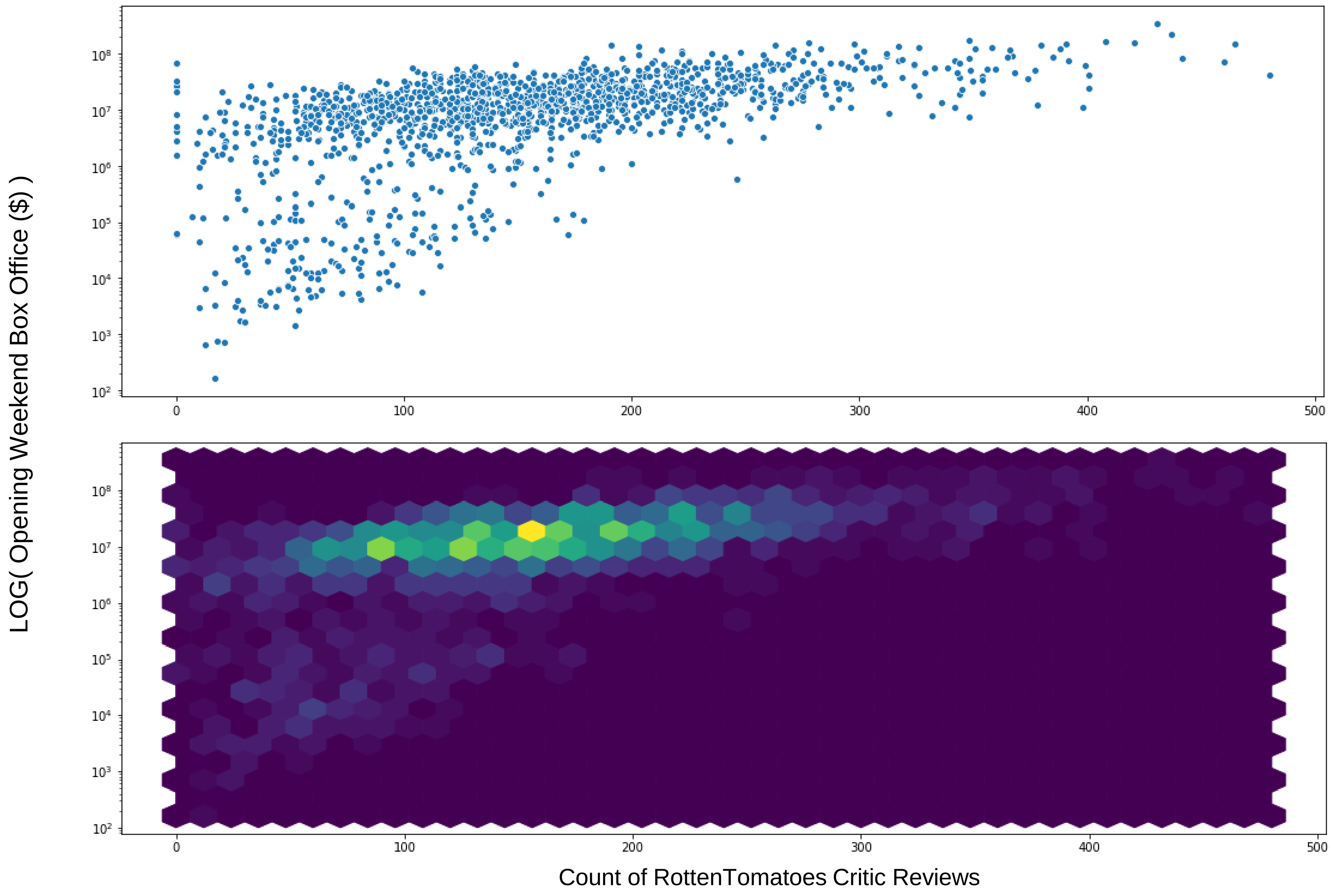
A few outliers that show a divergence between RT and Metacritic scores (i.e. strong score from one source but a much different score from the other source).



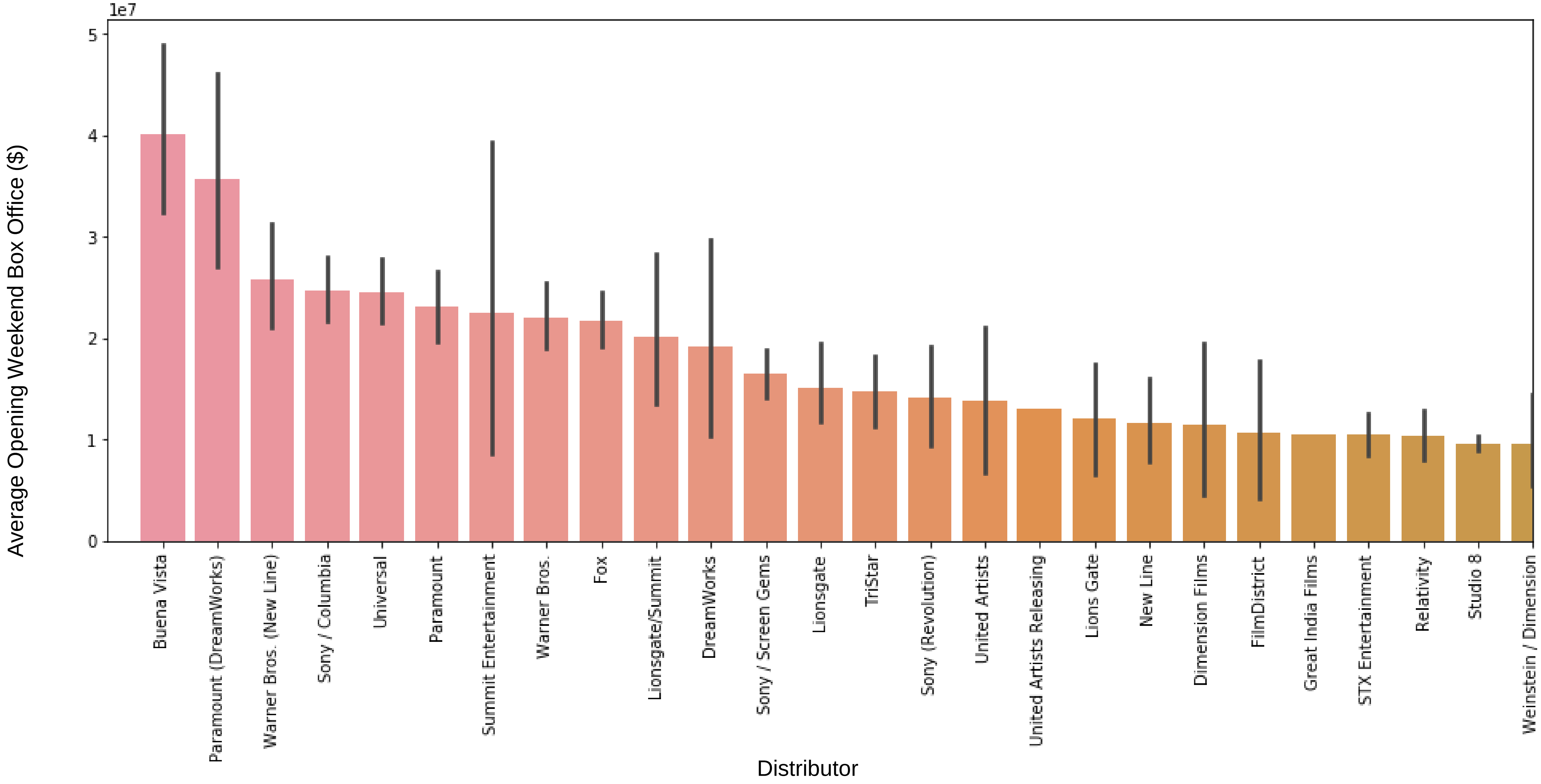
Middling RottenTomatoes scores average higher opening weekend box office



Number of critic reviews another strong indicator of box office performance



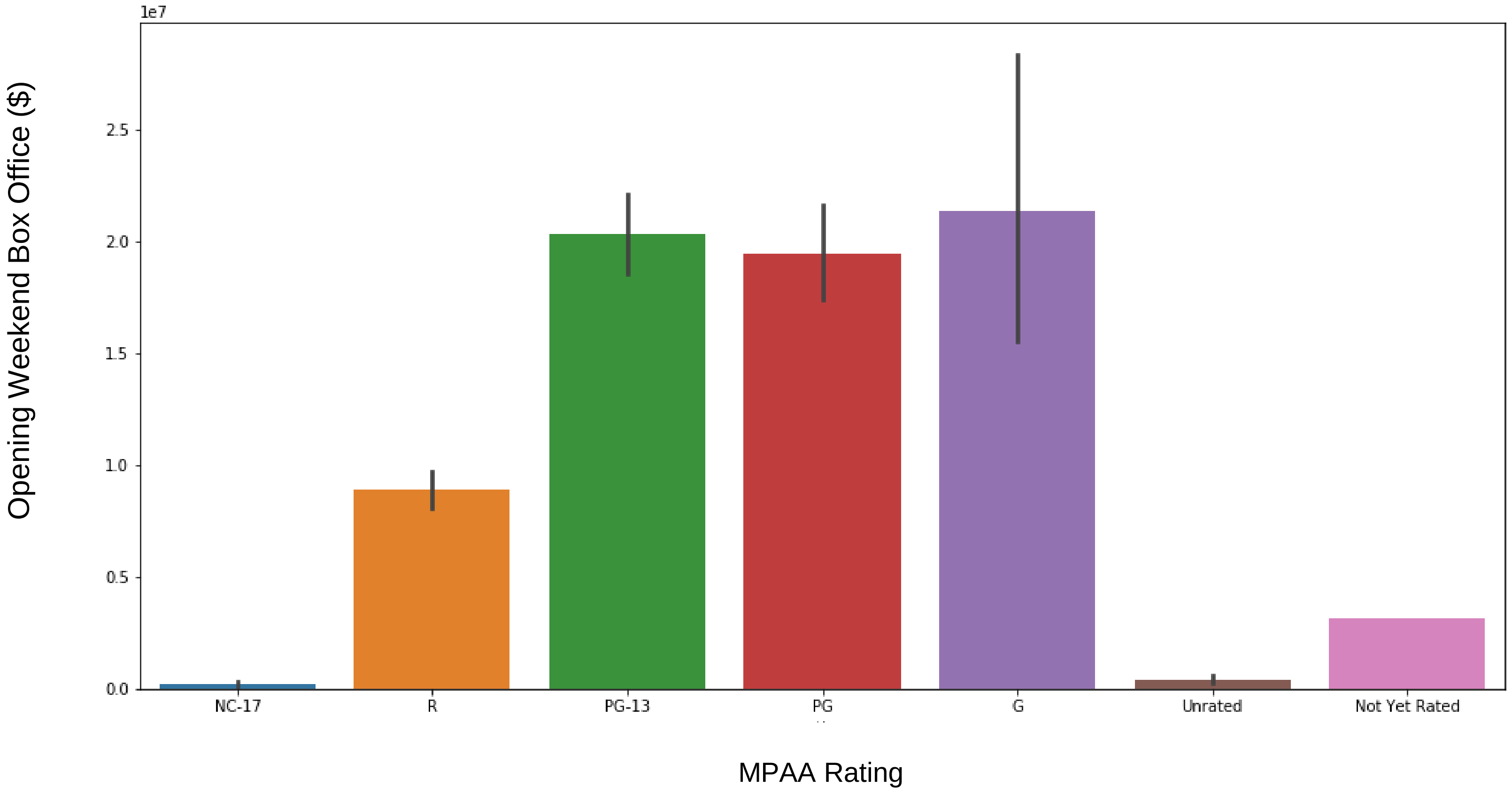
Average box office by distributor – Top 25



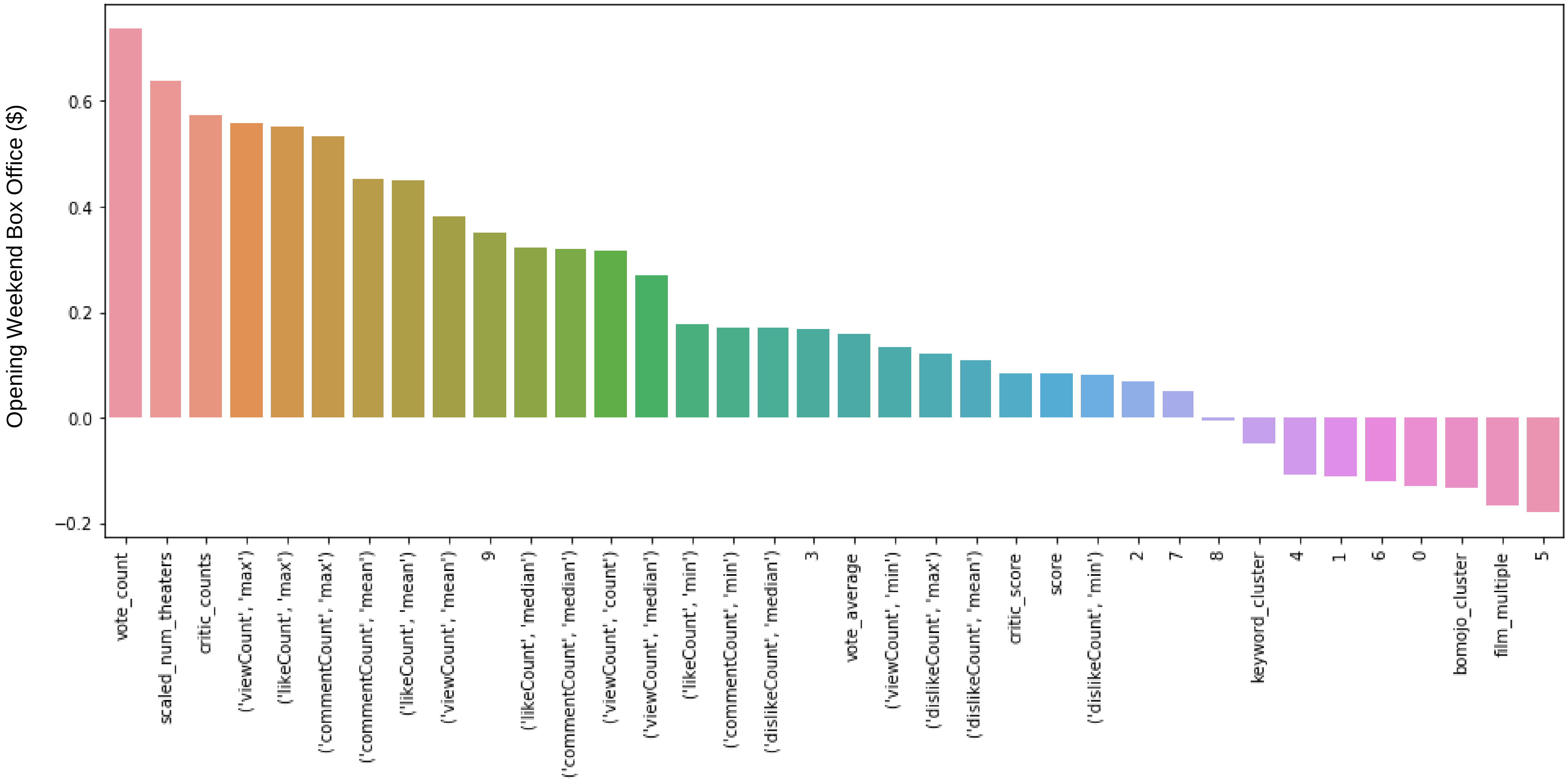
Average box office by rating

Rating is an important categorical variable for some movies

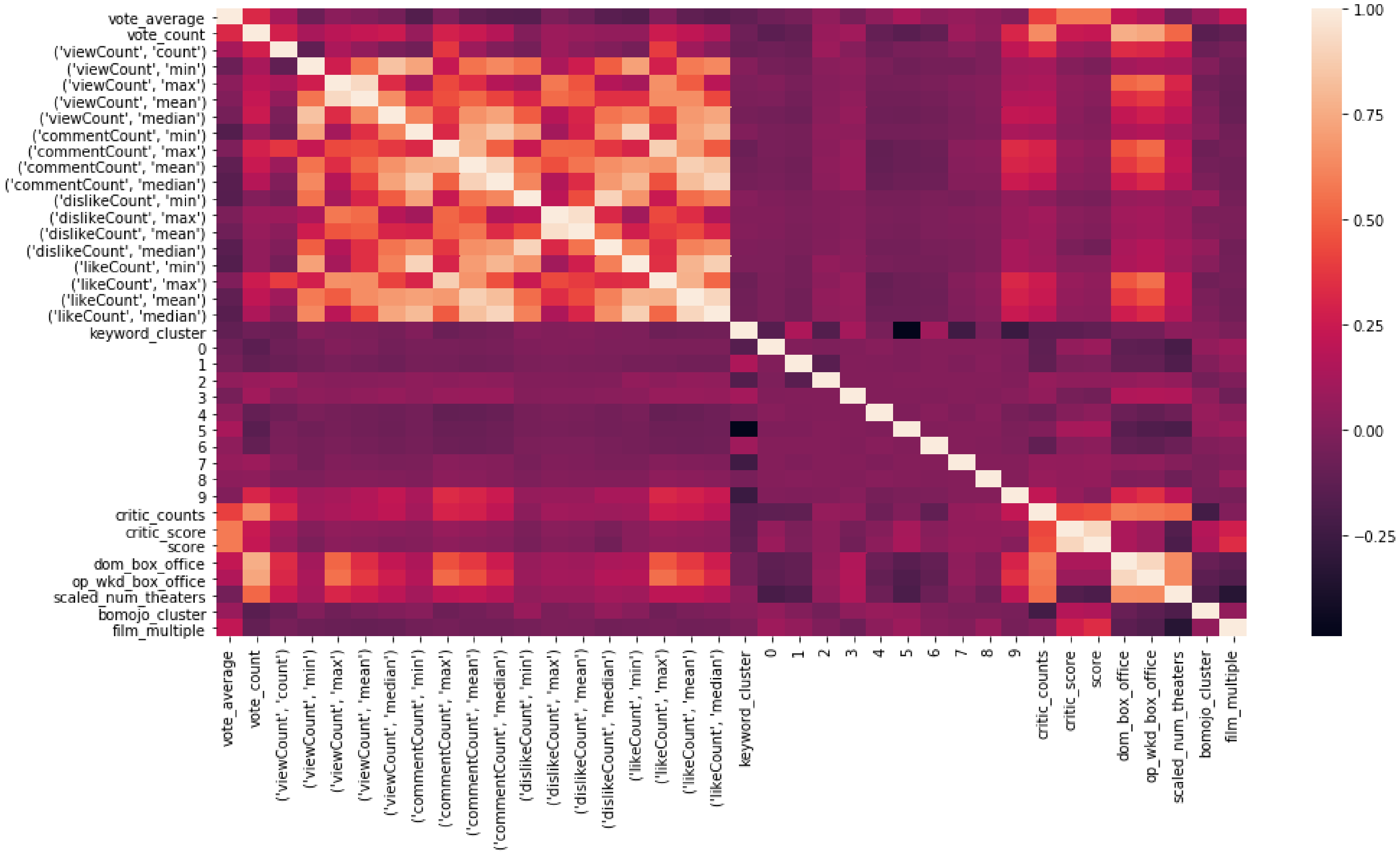
Not surprising to see animated titles (PG and G) having high average box office albeit with a lot of variance



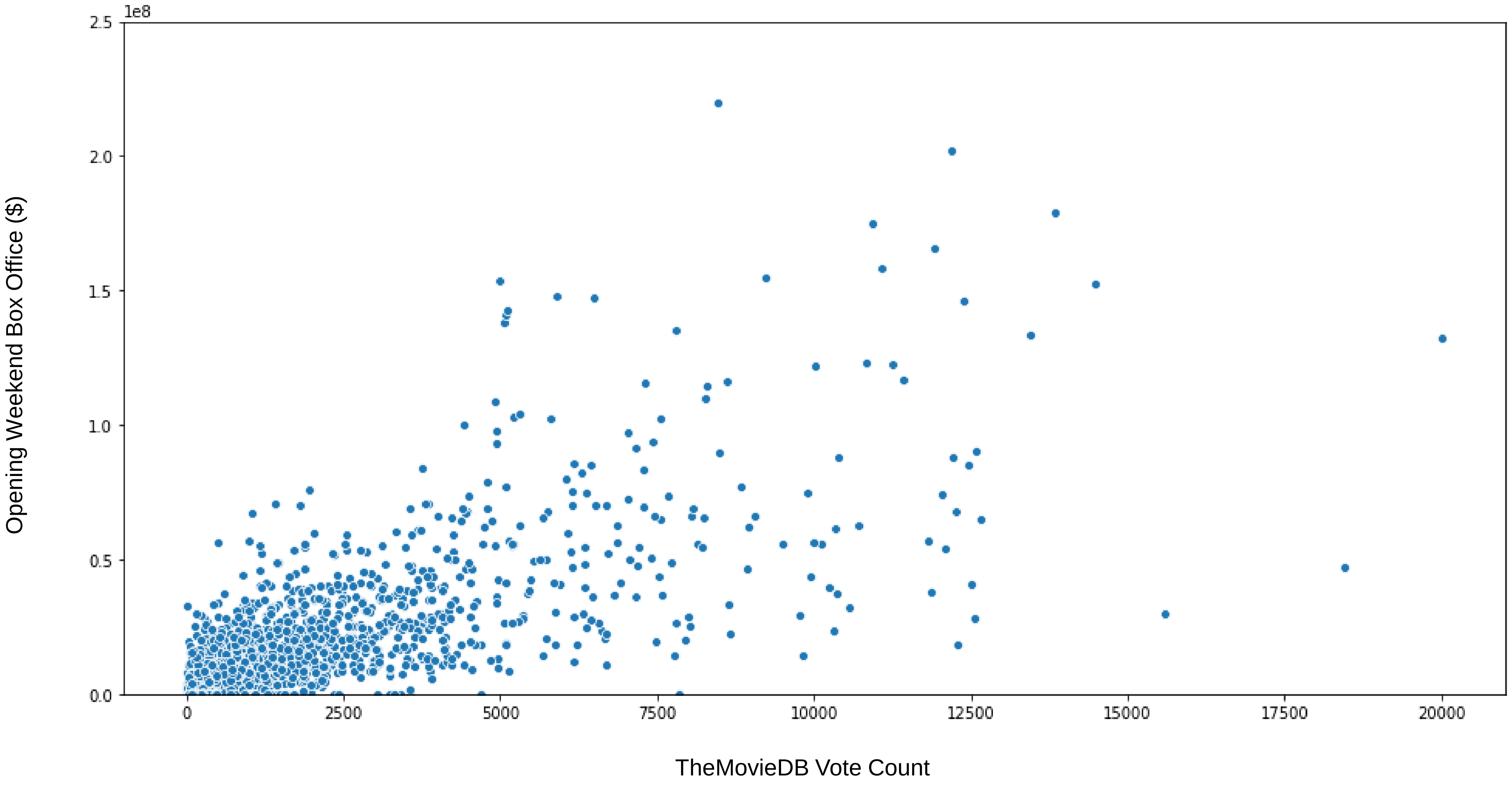
Feature correlation to target variables



Heatmap for feature correlation



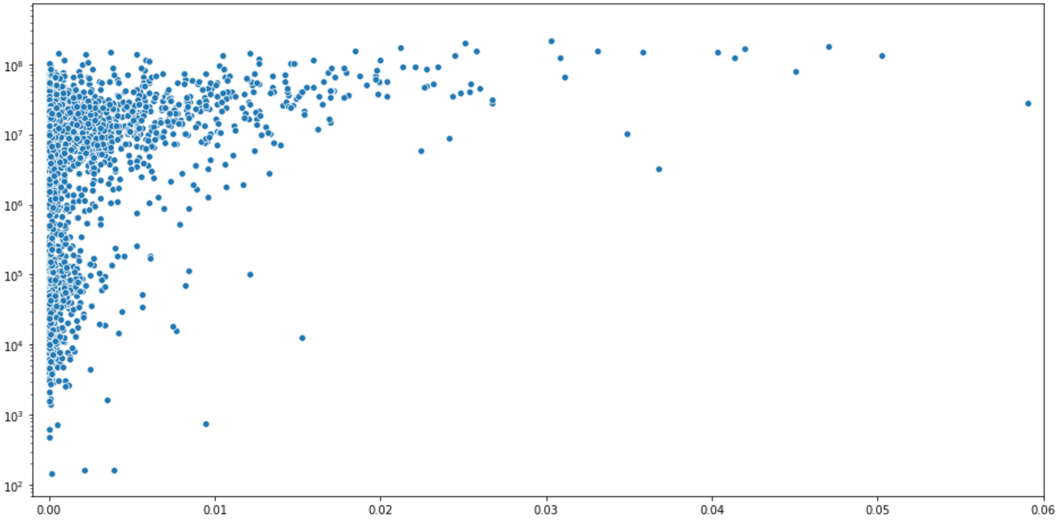
Vote count highly correlated to opening weekend box office



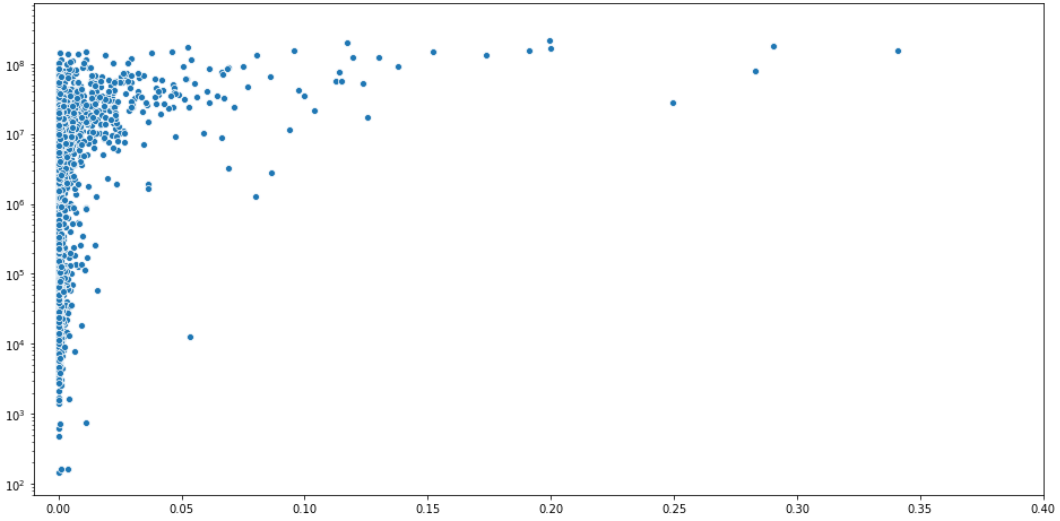
Four most highly correlated variables to box office point to the most popular trailer

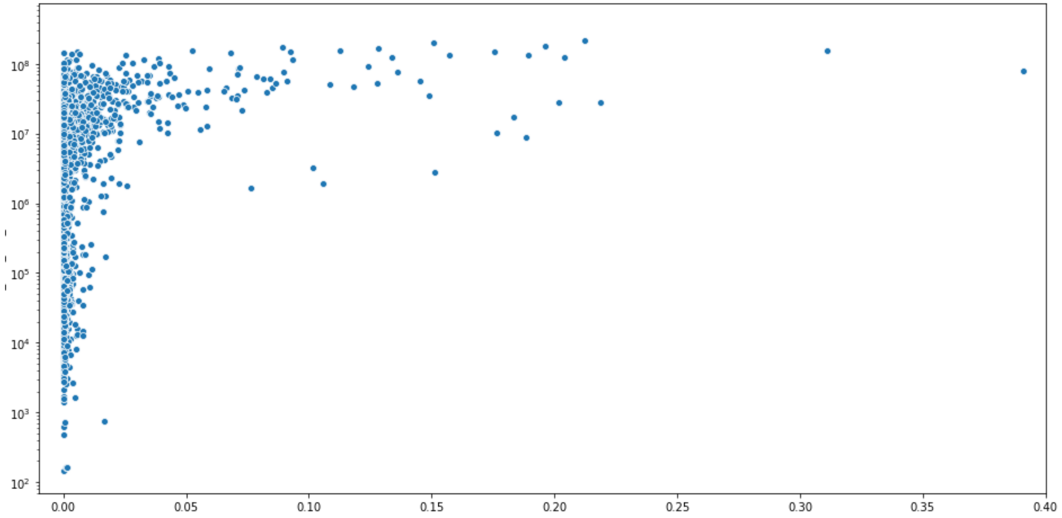
LOG (Opening Weekend Box Office ($))

Max of all trailers – View Count (scaled)

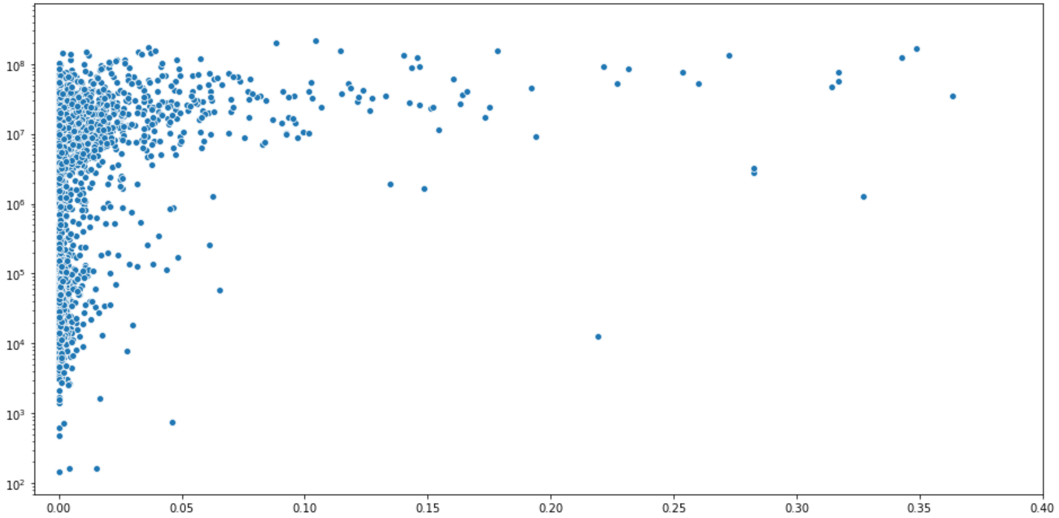


Max of all trailers – Comment Count (scaled)





Max of all trailers – Like Count (scaled)



MEAN of all trailers – Comment Count (scaled)

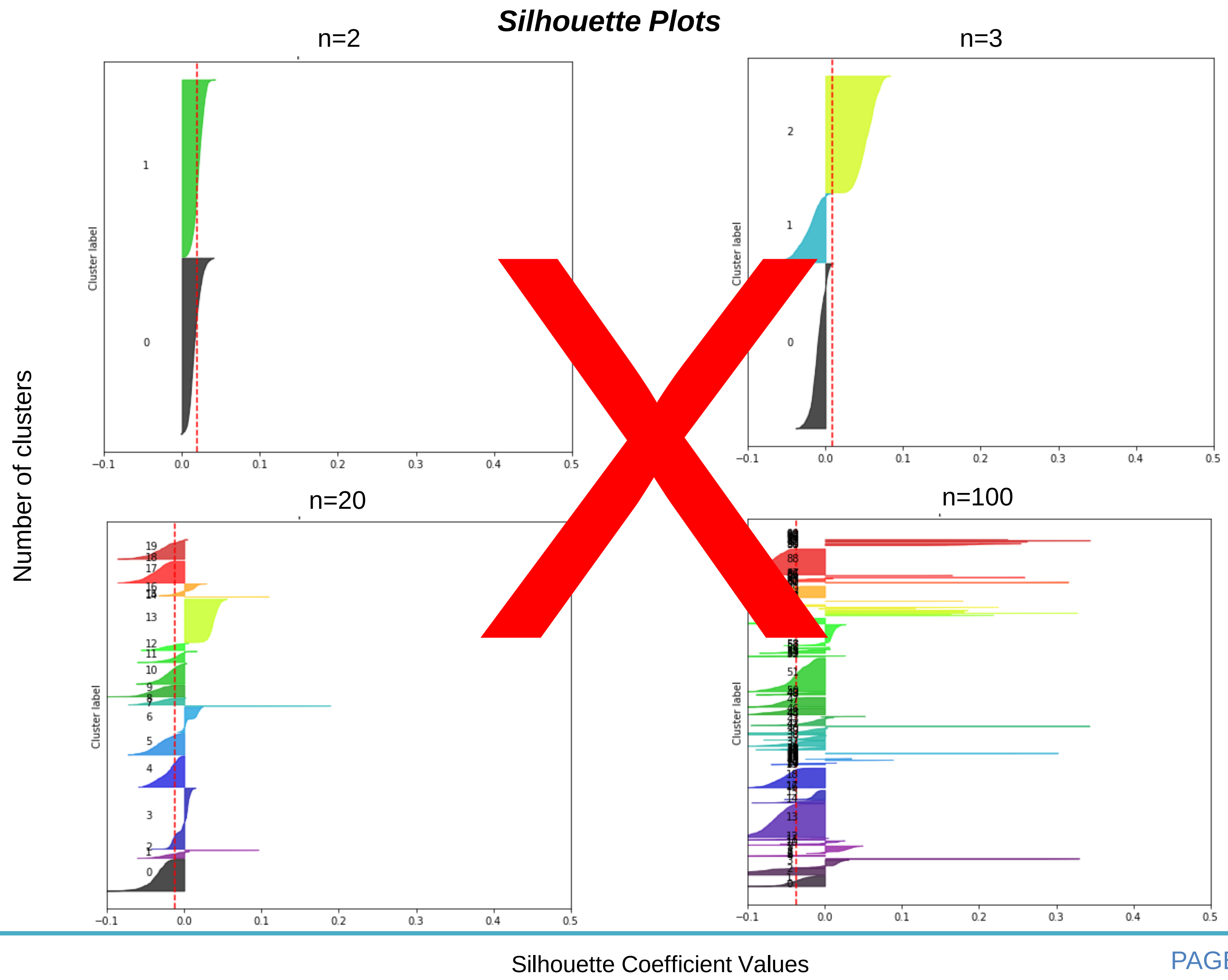
Clustering for feature selection / simplification

Clustering TheMovieDB info based on:

- *Genre*
- *Movie keywords*
- *Actor credits*
- *Production company*
- *Production country*

Silhouette scores per n clusters:

N_clusters	Avg. Silhouette Score
2	0.018689
3	0.008345
4	0.0016428
5	0.001270
10	-0.007432
20	-0.012462
100	-0.037647



Clustering for feature selection / simplification (cont.)

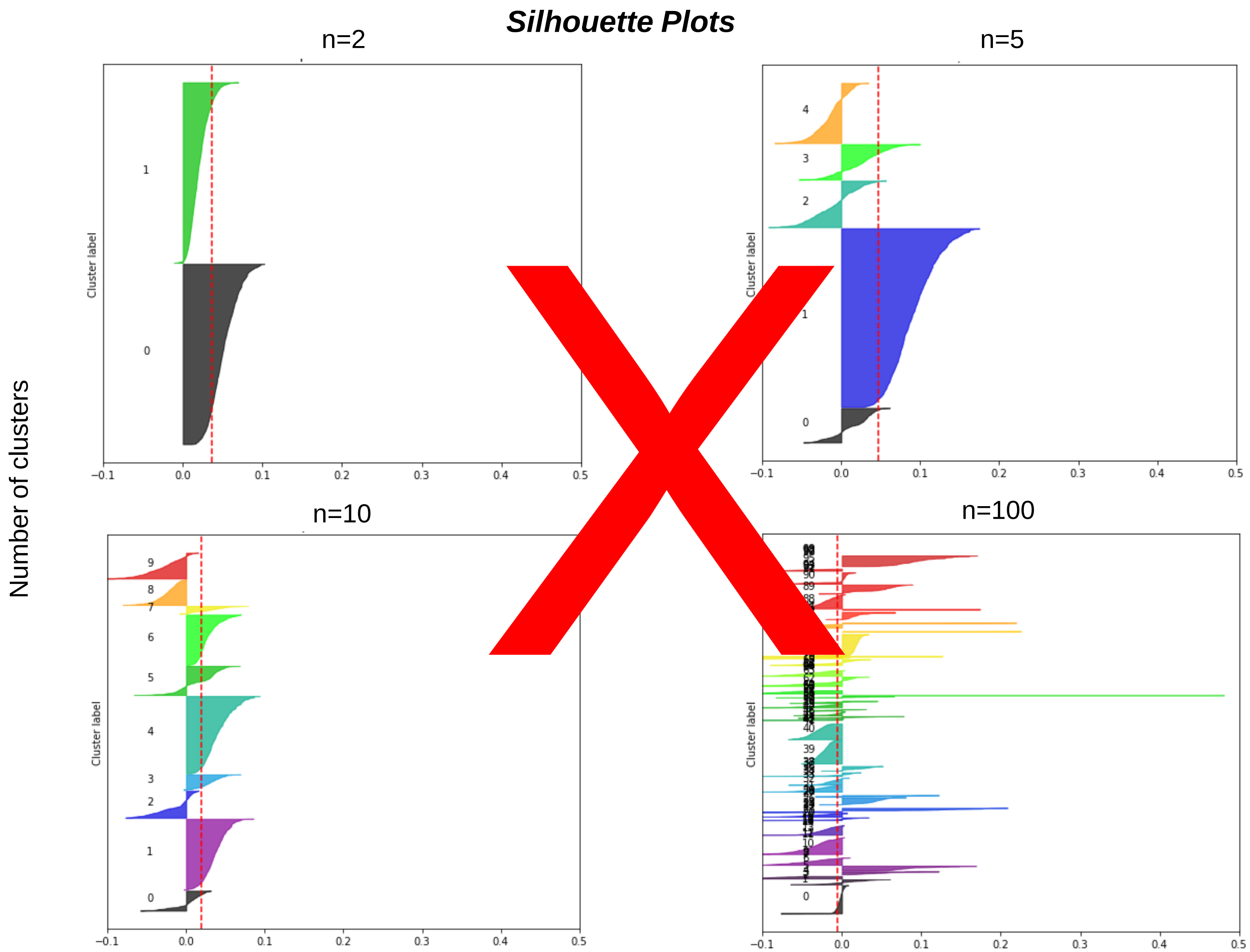
Clustering TheMovieDB info based on:

- *Genre*
- *Movie keywords*

(removed production companies and countries)

Silhouette scores per n clusters:

N_clusters	Avg. Silhouette Score
2	0.035881
3	0.023160
4	0.021300
5	**0.046275**
10	0.018956
20	0.001371
30	-0.007013
100	-0.005524



Silhouette Coefficient Values

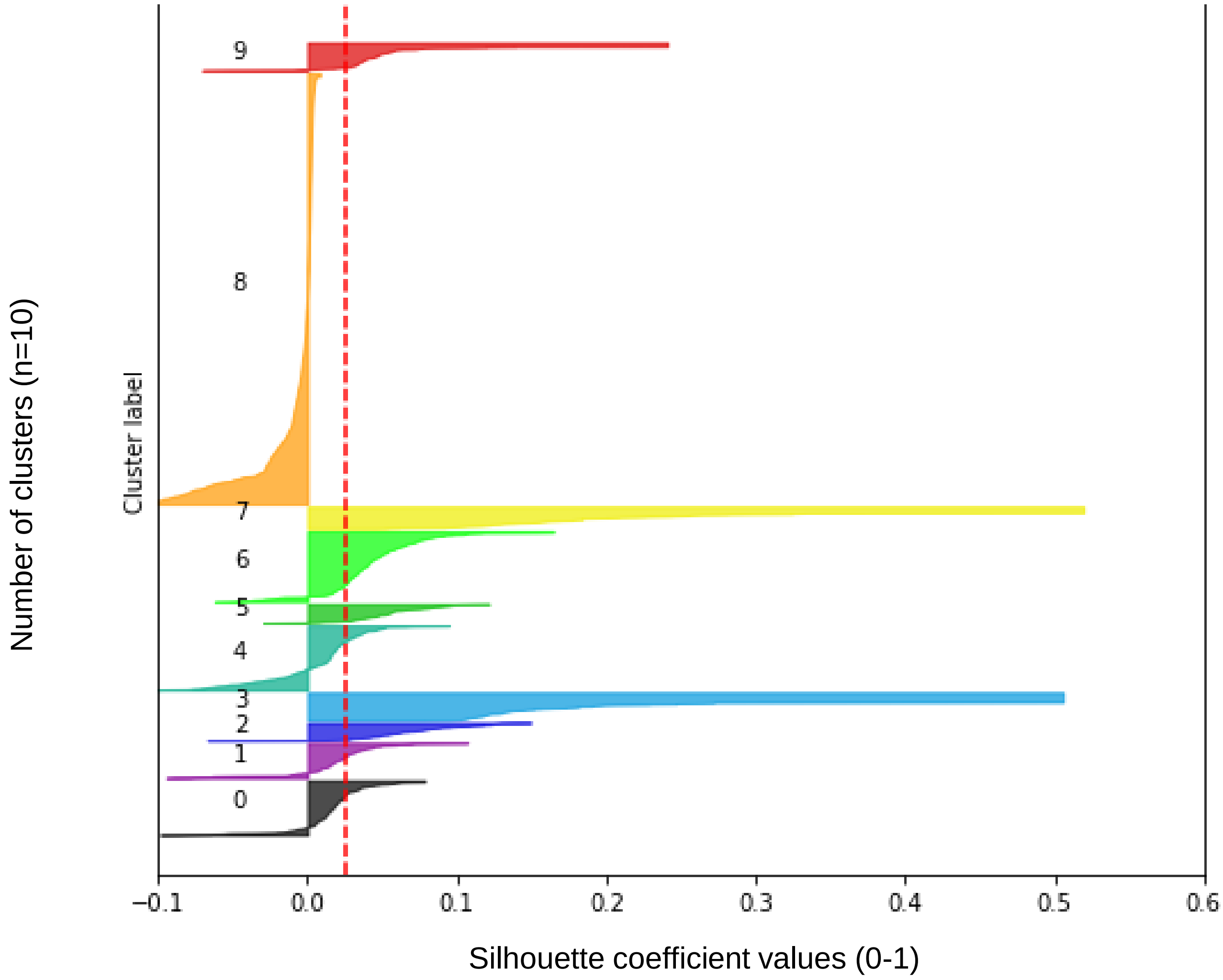
TF-IDF plus clustering of movie keywords for feature reduction

First, TF-IDF the movie keywords, and then try to cluster into similar movies
(removed genre, production companies and countries)

Silhouette scores per n clusters:

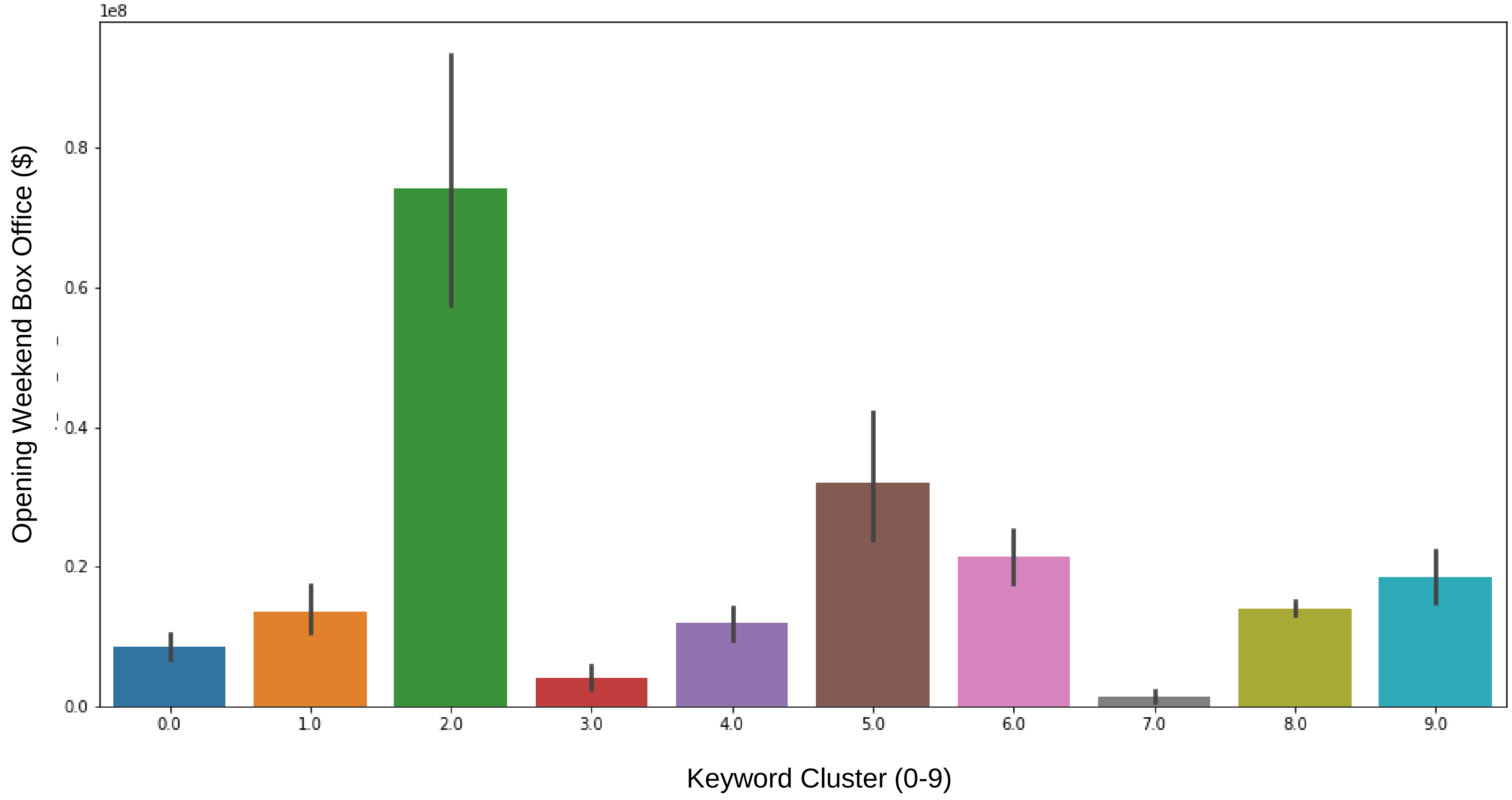
N_clusters	Avg. Silhouette Score
2	0.011560
3	0.013441
4	0.019353
5	0.019337
10	**0.025076**
20	0.030117
30	0.032828
40	0.033914
60	0.039980
80	0.042031
100	0.007253

Silhouette Plot



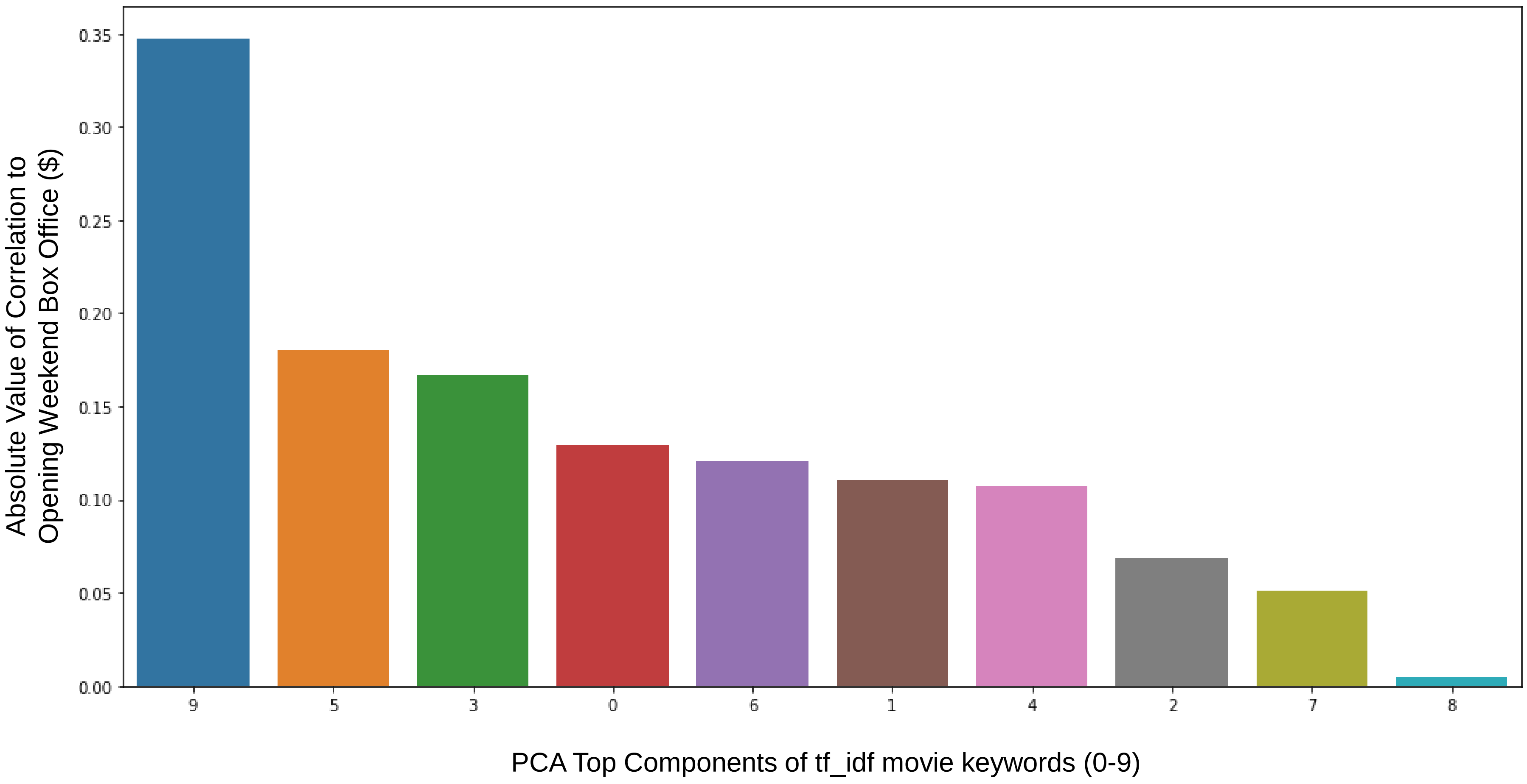
Keyword clustering identifies differences in box office performance

Despite relatively low silhouette scores for keyword clustering, the clusters do seem to have identified structure in the data because the box office performance in the clusters all seem quite different with also differing levels of variance.



PCA on tf_idf of movie keywords

Graph shows the absolute value of the correlation of each of the top 10 components to target variable (opening weekend box office)



Clustering box office mojo information

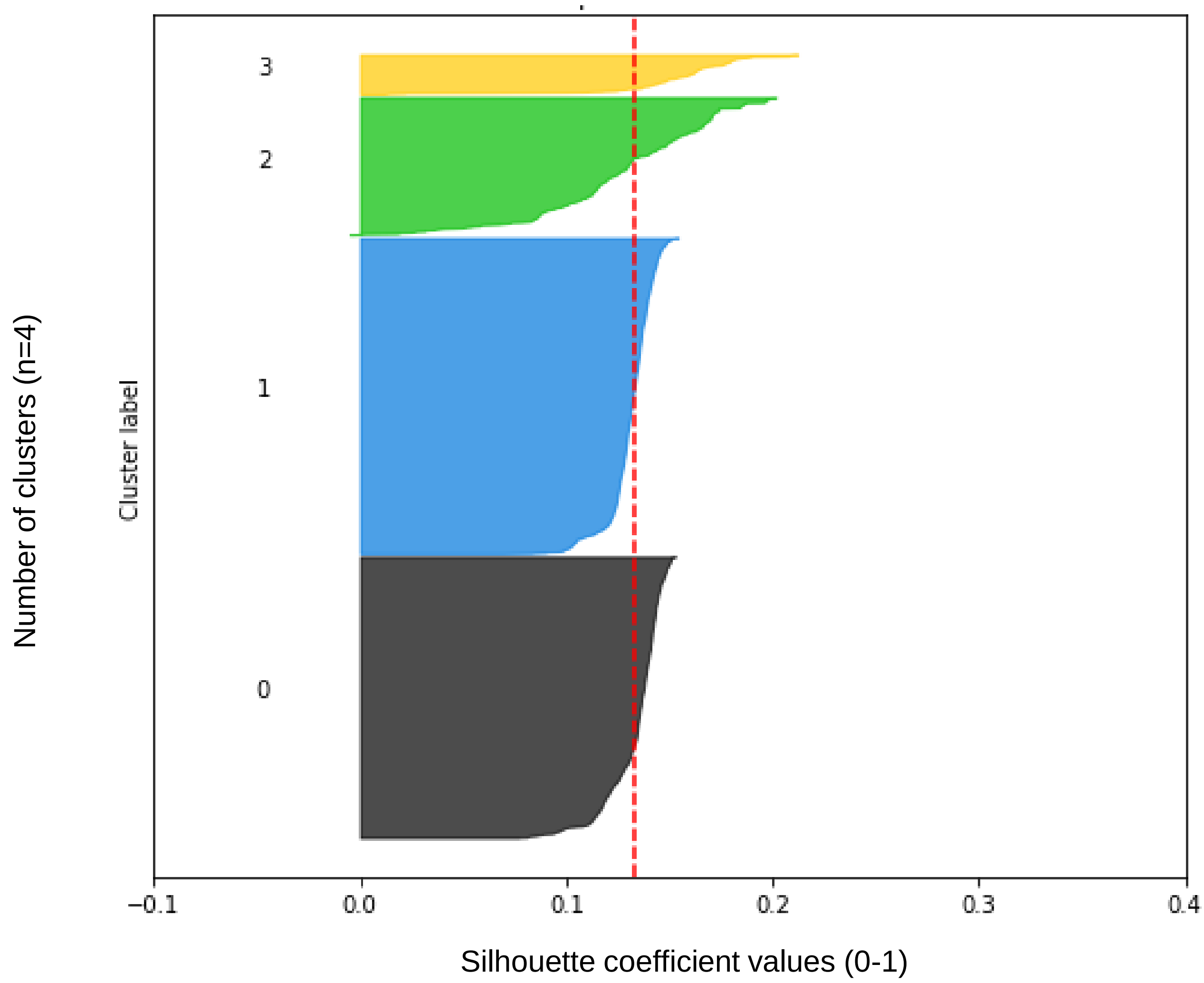
Clustering BOMOJO info based on:

- *Genre*
- *Rating*
- *Distributor*
- *Week of release*
- *Number of theaters (scaled)*

Silhouette scores per n clusters:

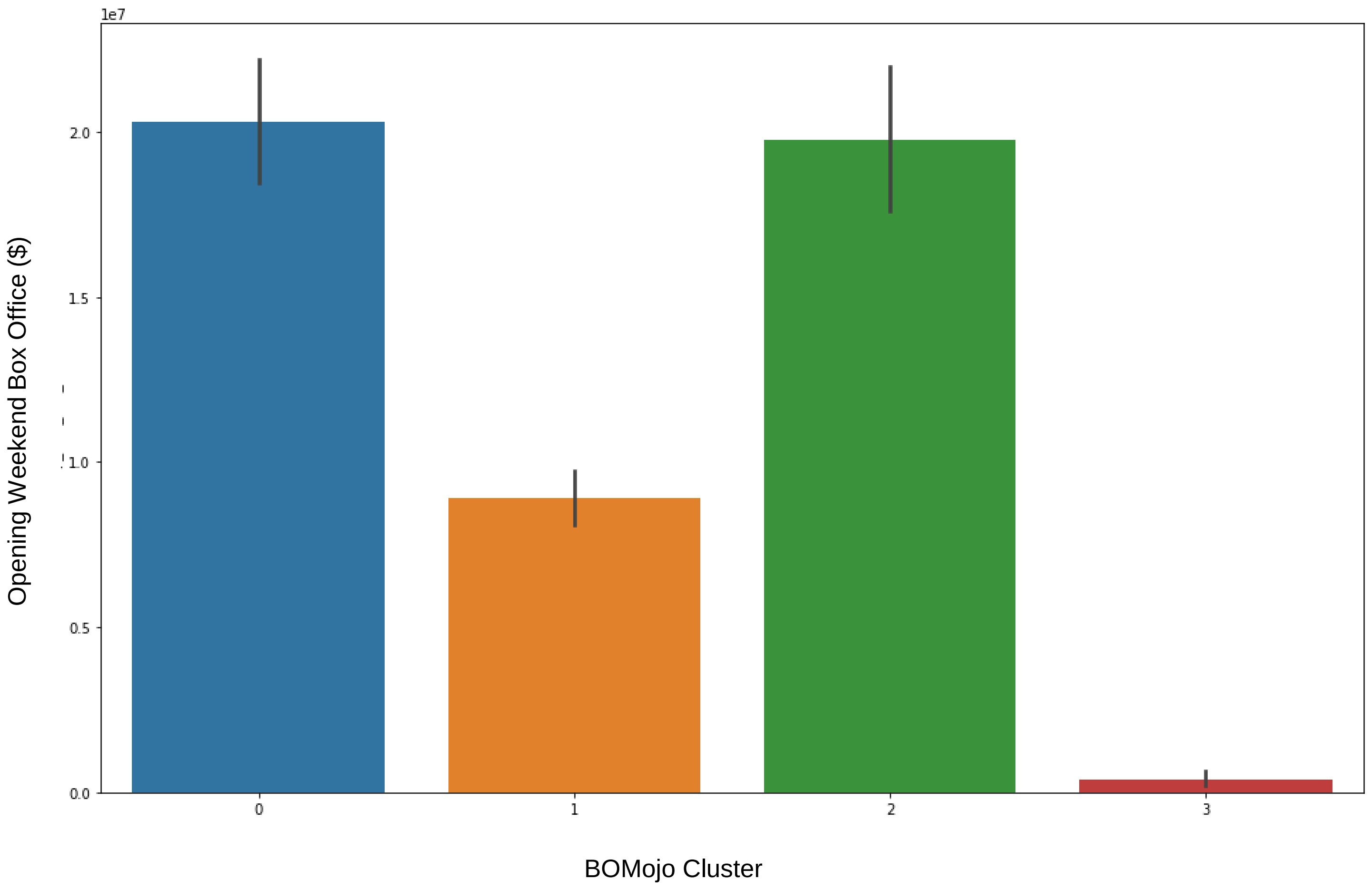
N_clusters	Avg. Silhouette Score
2	0.096590
3	0.125122
4	**0.132554**
5	0.089168
6	0.058371
7	0.065418
8	0.070180
9	0.073333
10	0.065304

Silhouette Plot

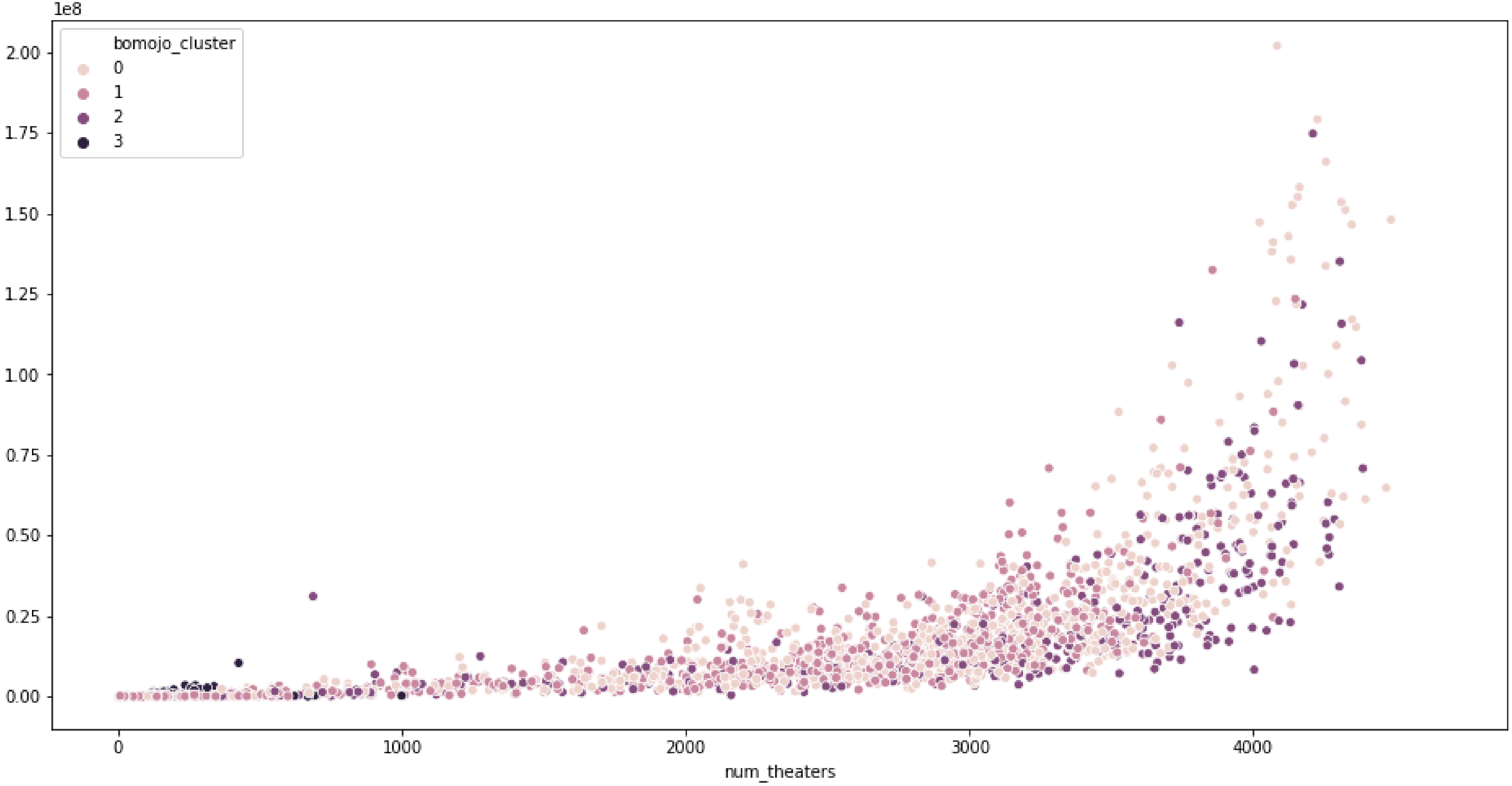


Comparing box office profiles of BOMojo clusters

Some differentiation between clusters 1 and 3, not much differentiation between clusters 0 and 2



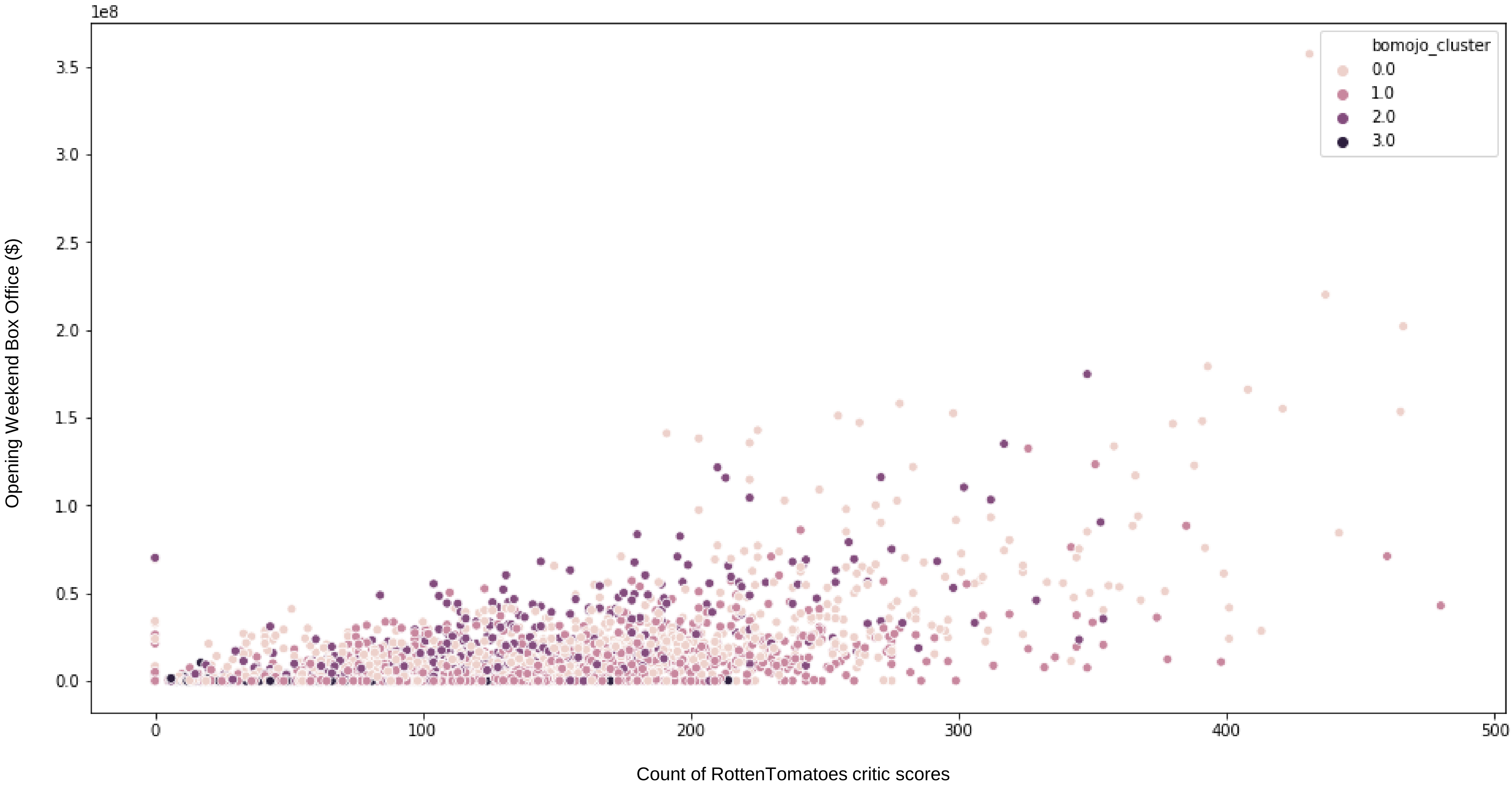
Number of theaters released versus opening weekend by BOMojo cluster



Opening Weekend Box Office ($)

Number of Theaters in Release

Critic counts versus opening weekend by BOMojo cluster



Pipeline recap

Data Sources

TheMovieDB API

Google /Youtube API

RottenTomatoes

Metacritic

Box Office Mojo

Exploration

Bar plots to visualize categorical against output

Scatter plots for feature to output comparison

Correlation matrices / graphs

Hexbin diagrams to identify density

Features

Scaling continuous features (theaters, etc.)

Seasonality (week number)

Genre / Keyword / Production Co. Clustering

Genre / Keyword Clustering

Keyword tf_idf clustering

Keyword tf_idf PCA

Modeling

Predictors:

Opening weekend box office

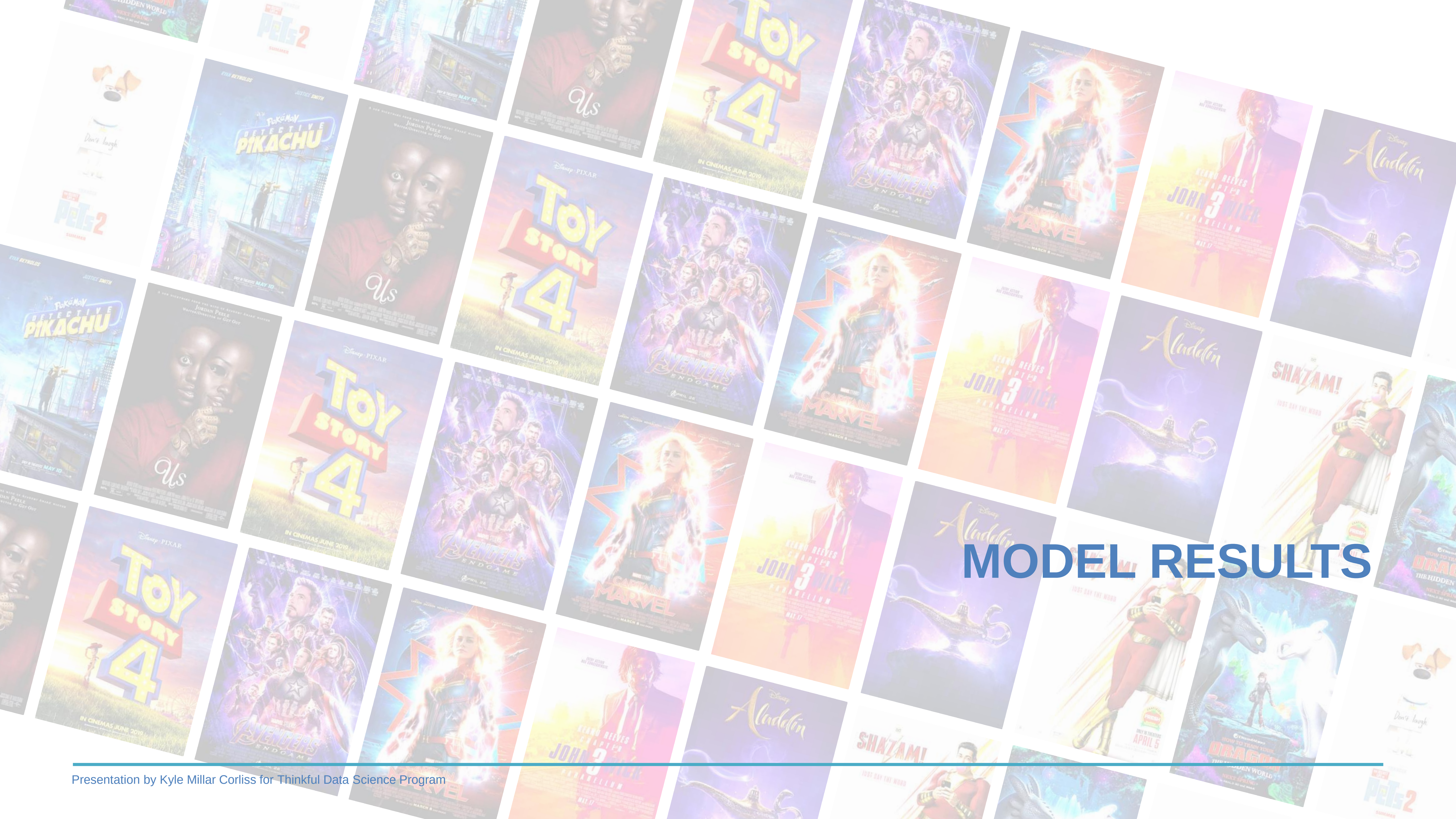
Movie multiple

Algorithms:

Neural Network

Random Forest Regression

Linear Regression



MODEL RESULTS

Neural network build

Train, test split of 15%

Did not have enough data (~1.8k movies) to do a train, test and holdout split

Input code:

```
from keras.layers import LeakyReLU, PReLU

model = Sequential()

model.add(Dense(128,input_shape=input_shape,activation='relu'))
model.add(Dense(64))
model.add(PReLU(alpha_initializer='zeros'))
model.add(Dense(64))
model.add(LeakyReLU(alpha=.03))
model.add(Dropout(0.1))
model.add(Dense(64))
model.add(LeakyReLU(alpha=.03))
model.add(Dense(64, activation='elu'))
model.add(Dense(64))
model.add(LeakyReLU(alpha=.03))
model.add(Dense(64))
model.add(LeakyReLU(alpha=.03))
model.add(Dropout(0.1))
model.add(Dense(64))
model.add(LeakyReLU(alpha=.03))
model.add(Dense(64))
model.add(LeakyReLU(alpha=.03))
model.add(Dense(64))
model.add(LeakyReLU(alpha=.03))
model.add(Dense(64, activation='elu'))
model.add(Dense(32, activation='elu'))
model.add(Dense(16, activation='elu'))
model.add(Dense(num_classes, activation='linear'))

model.summary()
# Compile the model to put it all together.
model.compile(loss='mean_absolute_error',
              optimizer=RMSprop(lr=0.0005),
              metrics=['mean_absolute_error'])
```

Model summary:

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 128)	44800
dense_2 (Dense)	(None, 64)	8256
p_re_lu_1 (PReLU)	(None, 64)	64
dense_3 (Dense)	(None, 64)	4160
leaky_re_lu_1 (LeakyReLU)	(None, 64)	0
dropout_1 (Dropout)	(None, 64)	0
dense_4 (Dense)	(None, 64)	4160
leaky_re_lu_2 (LeakyReLU)	(None, 64)	0
dense_5 (Dense)	(None, 64)	4160
dense_6 (Dense)	(None, 64)	4160
leaky_re_lu_3 (LeakyReLU)	(None, 64)	0
dense_7 (Dense)	(None, 64)	4160
leaky_re_lu_4 (LeakyReLU)	(None, 64)	0
dropout_2 (Dropout)	(None, 64)	0
dense_8 (Dense)	(None, 64)	4160
leaky_re_lu_5 (LeakyReLU)	(None, 64)	0
dense_9 (Dense)	(None, 64)	4160
leaky_re_lu_6 (LeakyReLU)	(None, 64)	0
dense_10 (Dense)	(None, 64)	4160
leaky_re_lu_7 (LeakyReLU)	(None, 64)	0
dense_11 (Dense)	(None, 64)	4160
dense_12 (Dense)	(None, 32)	2080
dense_13 (Dense)	(None, 16)	528
dense_14 (Dense)	(None, 1)	17

Total params: 93,185
Trainable params: 93,185
Non-trainable params: 0

NN epochs and overfitting

500 epochs for training

Average time per epoch: 152.8 us/step

Dropout layers added to reduce overfitting

Could also add additional regularization within the model build

NN Model output (500 epochs)

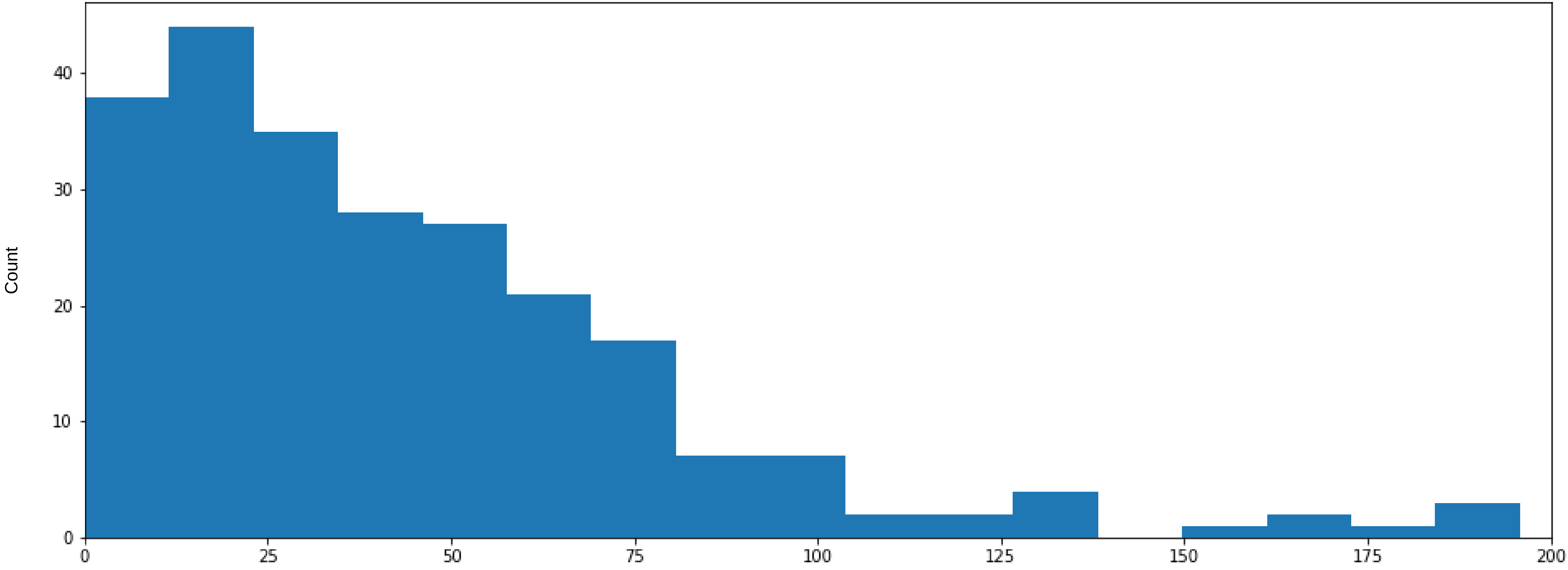
```
Train on 1453 samples, validate on 257 samples
Epoch 1/500
1453/1453 [==============================] - 1s 610us/step - loss: 13186327.6779 - mean_absolute_error:
13186327.6779 - val_loss: 9797664.3677 - val_mean_absolute_error: 9797664.3677
Epoch 2/500
1453/1453 [==============================] - 0s 155us/step - loss: 9765329.4123 - mean_absolute_error: 9765329.4123
- val_loss: 10204777.8881 - val_mean_absolute_error: 10204777.8881
Epoch 3/500
1453/1453 [==============================] - 0s 160us/step - loss: 9672081.7440 - mean_absolute_error: 9672081.7440
- val_loss: 9898101.1089 - val_mean_absolute_error: 9898101.1089

Epoch 496/500
1453/1453 [==============================] - 0s 249us/step - loss: 5524255.5988 - mean_absolute_error: 5524255.5988
- val_loss: 6777911.4786 - val_mean_absolute_error: 6777911.4786
Epoch 497/500
1453/1453 [==============================] - 0s 223us/step - loss: 5376074.5788 - mean_absolute_error: 5376074.5788
- val_loss: 7352383.8249 - val_mean_absolute_error: 7352383.8249
Epoch 498/500
1453/1453 [==============================] - 0s 252us/step - loss: 5629128.1889 - mean_absolute_error: 5629128.1889
- val_loss: 6844232.0136 - val_mean_absolute_error: 6844232.0136
Epoch 499/500
1453/1453 [==============================] - 0s 260us/step - loss: 5609150.8802 - mean_absolute_error: 5609150.8802
- val_loss: 7530954.5700 - val_mean_absolute_error: 7530954.5700
Epoch 500/500
1453/1453 [==============================] - 0s 228us/step - loss: 5649423.9317 - mean_absolute_error: 5649423.9317
- val_loss: 6685626.2354 - val_mean_absolute_error: 6685626.2354
```

Neural network results and histogram of errors

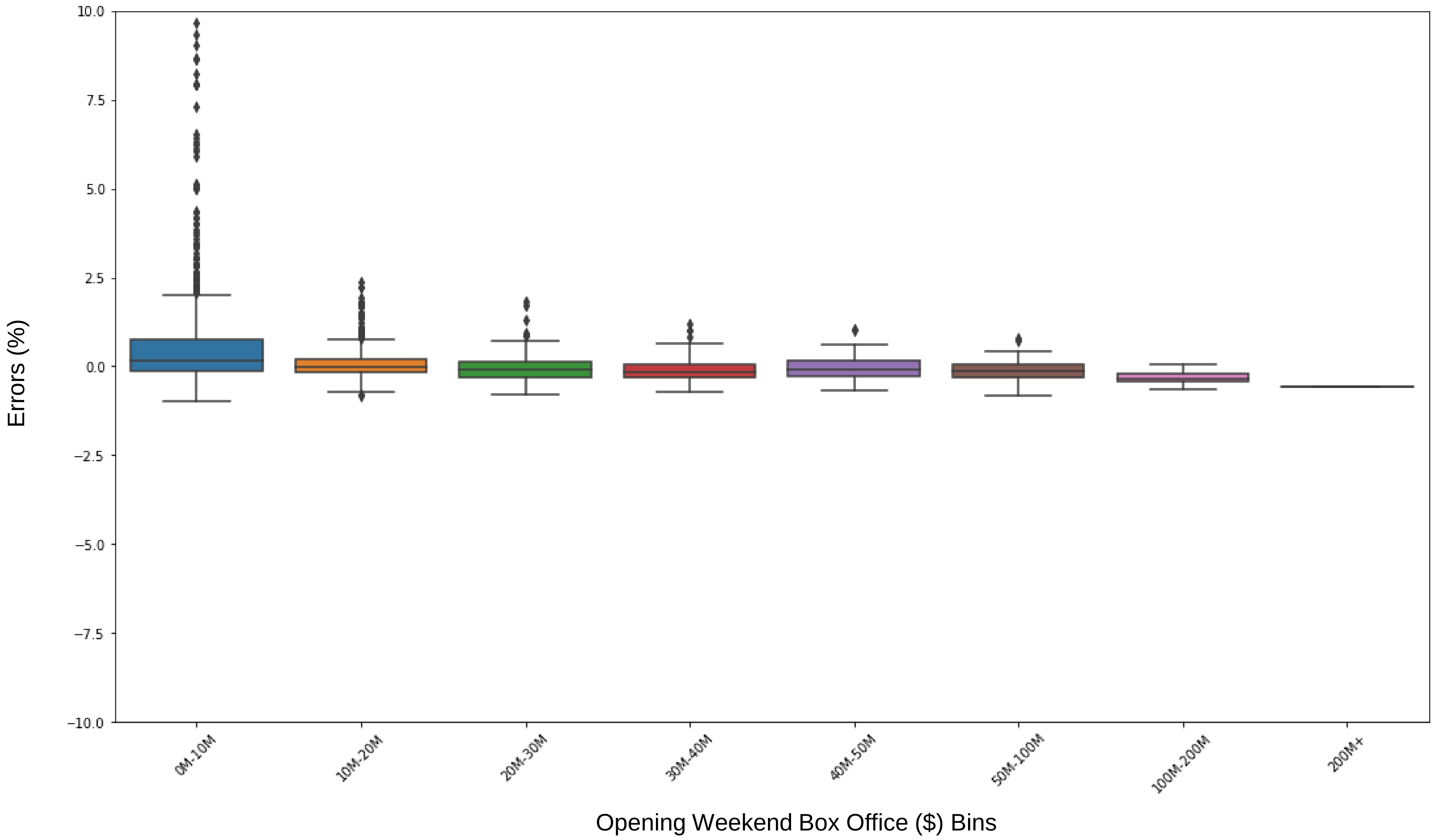
50% of predictions were within 39% of actual results…

Mean absolute error = 6,685,626

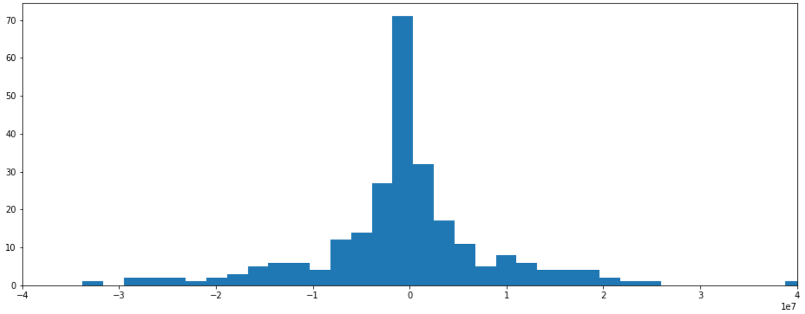
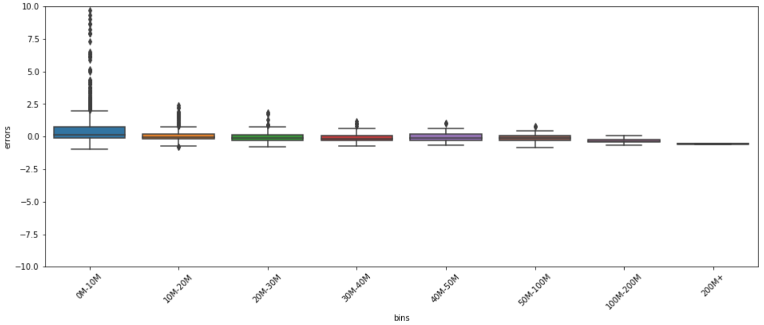
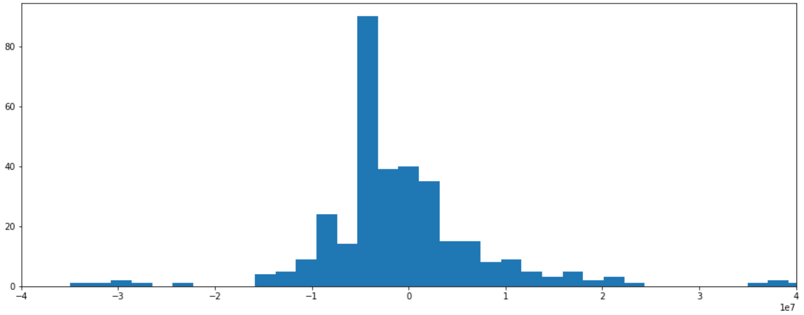
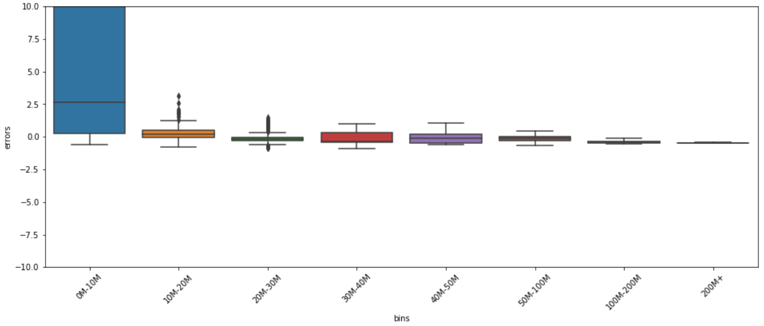
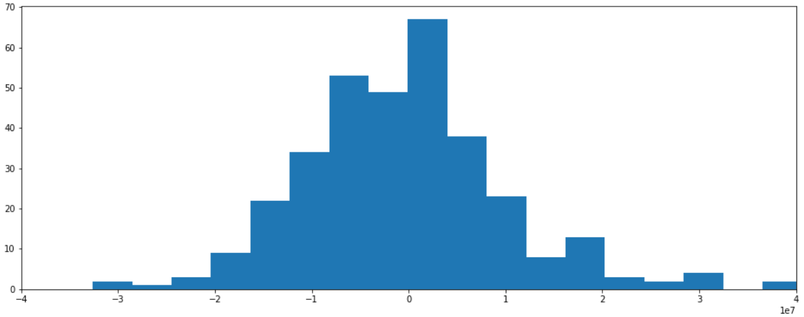
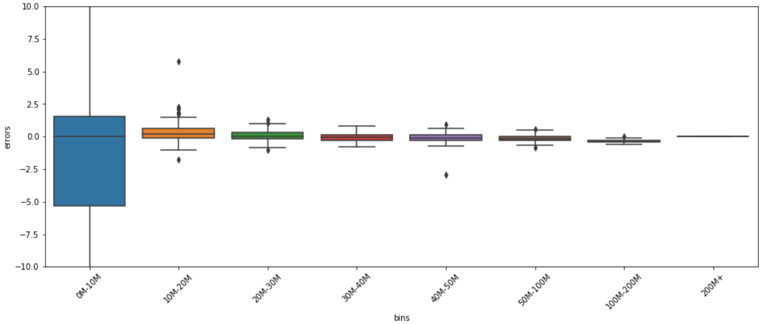


Absolute Value of Errors (predicted / actual – 1)

NN much better at predicting movies > $10M opening weekend



Model comparison – NN vs. Random Forest Regressor vs. Linear Regression

	Median Absolute Error	R^2	Histogram of Residuals	Errors by size of movie
NN	**6,685,626**	N.A.		
RFR	6,852,821	74.0%		
LR	9,527,990	83.9%		

Model comparison – quantiles of results

X% of predictions were within Y% of actual results…

	Errors (%)		
% of Predictions	**NN**	**RFR**	**LR**
10%	6.9%	7.5%	10.7%
20%	15.6%	14.6%	20.0%
25%	19.3%	20.2%	24.6%
50%	**39.0%**	**43.3%**	**58.6%**
75%	68.5%	146.3%	258.9%
80%	76.6%	874.8%	1030.7%
85%	93.6%	3537.2%	2928.6%
90%	134.5%	8665.1%	9311.6%

Moving forward

- More movies spanning more years – i.e. more data to train NN
  - Current process is skewed towards more popular and larger movies
  - Potentially find a search metric (similar to Wikipedia pageview data that was not used in the model)

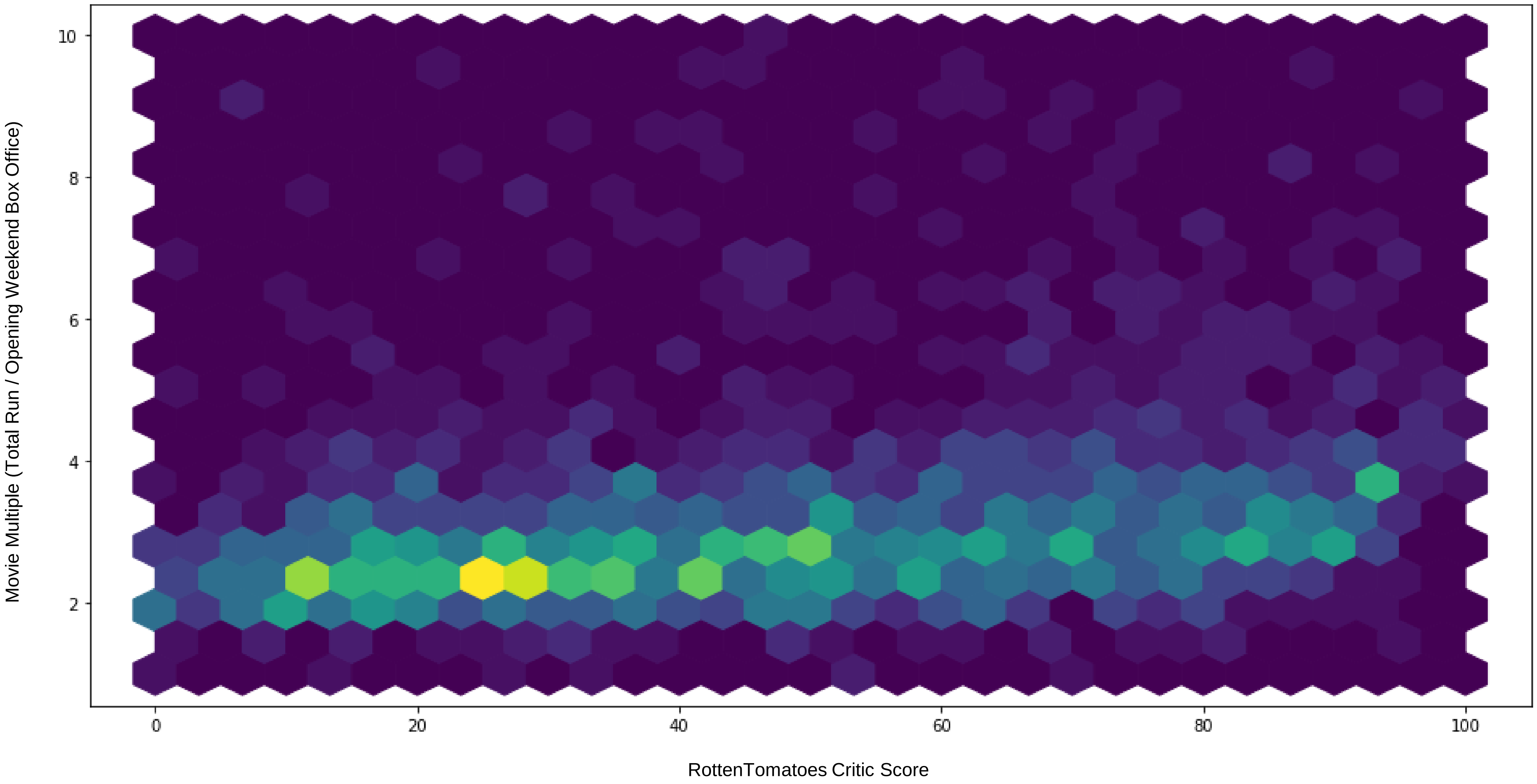
- Add actor credits and awards
  - Need to identify the talents' popularity (star power) and how it might contribute to the success of a movie

- Continue to fine-tune neural network parameters
- GridSearchCV for random forest or linear regression to try and match NN performance

THANK YOU

Presentation by Kyle Millar Corliss for Thinkful Data Science Program

Density of critic score versus movie multiple



Final model features

Input Variables:

- Vote average, count and popularity measure from TheMovieDB
- Trailer view stats:
  - View count, comment count, like count, dislike count
  - Max, min, mean, median
- [All Genres]
- [All Production Countries]
- Keyword cluster
- Keyword PCA features (top 10)
- RottenTomatoes critic review score and counts
- Metacritic score 'score'
- Movie studio / distributor
- TheMovieDB genre
- MPAA Rating
- Release date week number
- Number of theaters (scaled)
- Bomojo cluster (genre, distributor, rating, number of theaters (scaled))

Target Variable:

- Opening Weekend box office