What makes a film successful?

Predicting total domestic gross
via linear regression

Alexander Sigrist

Project Outline

1. Data
2. Model
3. Predictions
4. Conclusion & Future Steps

Data Collected

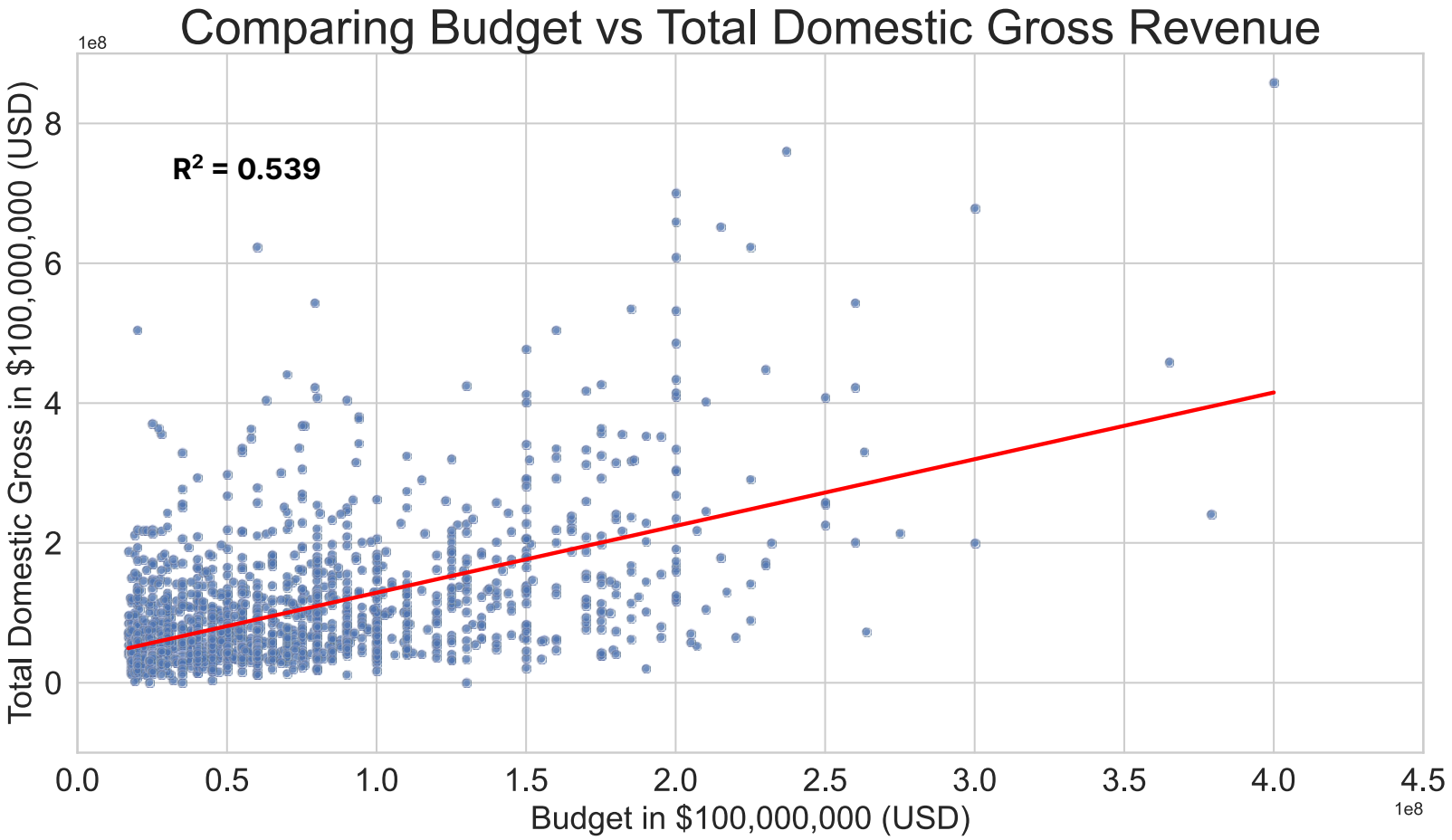
- Box Office Mojo

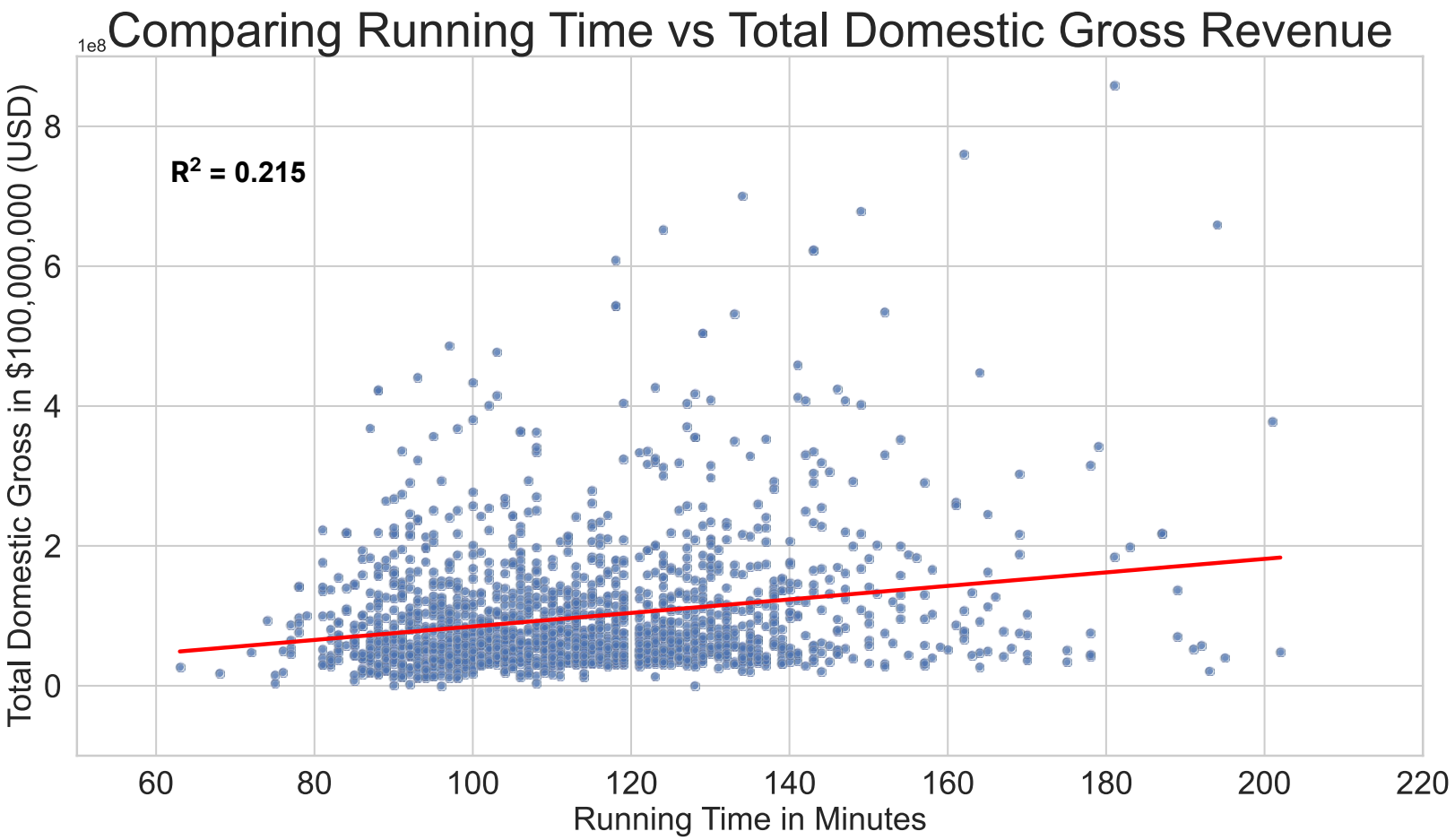
- The Numbers

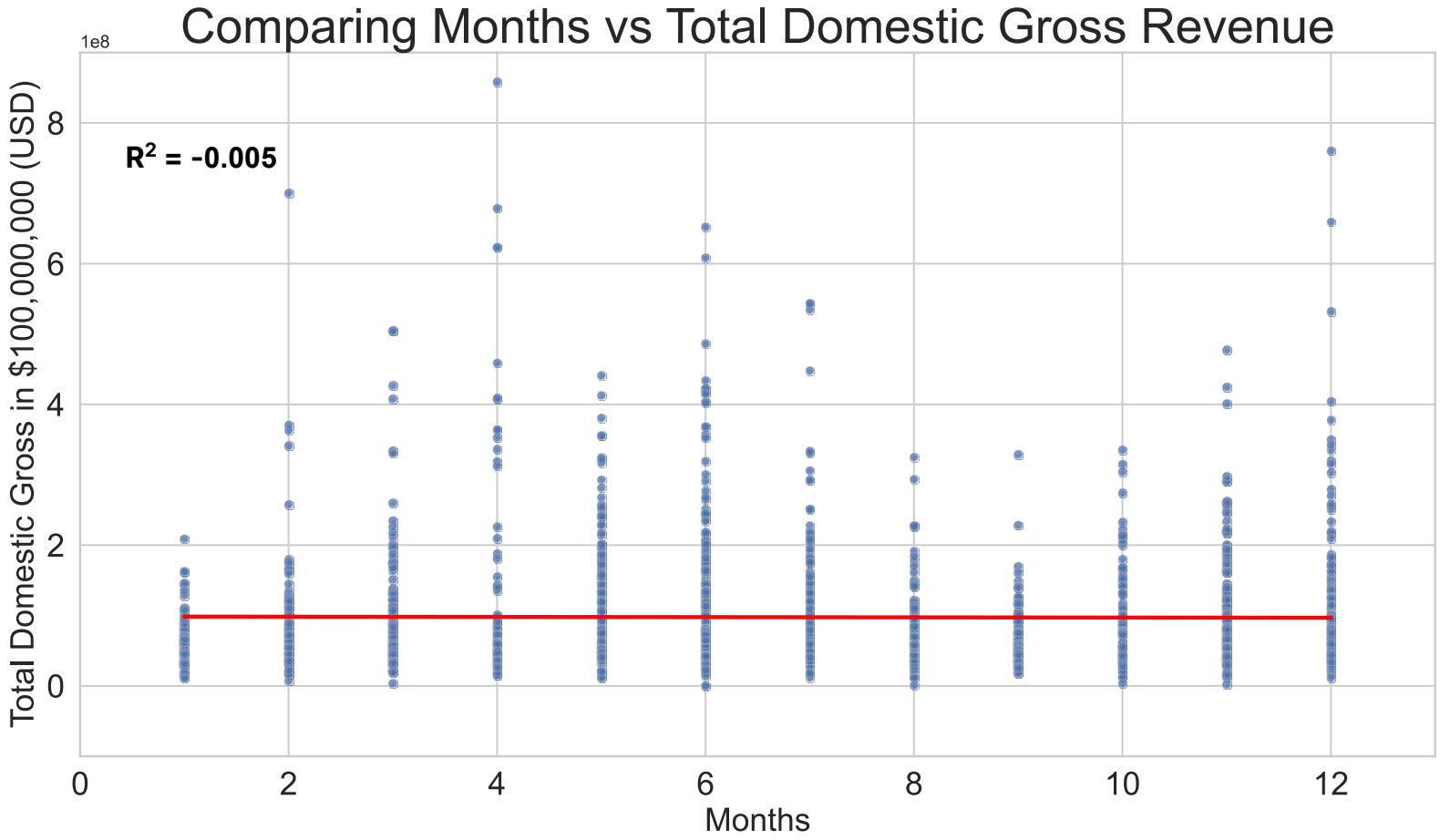
What makes a film domestically successful

- Distributor
- Season (Month)
- Year (Age)
- Runtime
- MPAA Rating
- Budget

?


Comparing Budget vs Total Domestic Gross Revenue
1e8
Total Domestic Gross in $100,000,000 (USD)
R² = 0.539
0
2
4
6
8
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
1e8
Budget in $100,000,000 (USD)


Comparing Running Time vs Total Domestic Gross Revenue
1e8
Total Domestic Gross in $100,000,000 (USD)
R² = 0.215
0
2
4
6
8
60
80
100
120
140
160
180
200
220
Running Time in Minutes


Comparing Months vs Total Domestic Gross Revenue
1e8
R^2 = -0.005
Total Domestic Gross in $100,000,000 (USD)
0
2
4
6
8
0
2
4
6
8
10
12
Months

Useful

- Budget
- Running Time
- Year

Less Useful

- Distributor
- MPAA Rating
- Month (Release Date)

Predictions

- Average error $48,084,222
- Scores well on lower revenue
- Terrible at top box-office earners

Film	Predicted	Actual	Error
The Call of the Wild	$62,342,370	$62,342,368	$1.12
Thomas and the Magic Railroad	$15,933,524	$15,933,506	$17.9
The Avengers	$160,012,700	$623,357,900	$463,345,200
Avatar	$262,016,200	$760,507,600	$498,491,400

Future Analysis

- Collect other features
- Examine predictor by a subset

Questions?

Appendix - Model

```
kf = KFold(n_splits=5, shuffle=True, random_state = 1)
cv_lm_r2s = [] #collect the validation results

for train_ind, val_ind in kf.split(X,y):

    X_train, y_train = X[train_ind], y[train_ind]
    X_val, y_val = X[val_ind], y[val_ind]

    #simple linear regression
    lm = LinearRegression()

    lm.fit(X_train, y_train)
    cv_lm_r2s.append(round(lm.score(X_val, y_val), 3))

print('Simple regression scores: ', cv_lm_r2s, '\n')

print(f'Simple mean cv r^2: {np.mean(cv_lm_r2s):.3f} +- {np.std(cv_lm_r2s):.3f}')
```

```
Simple regression scores:  [0.302, 0.185, 0.202, 0.457, 0.095]

Simple mean cv r^2: 0.248 +- 0.123
```

Appendix - Model reduced Residuals

```
kf = KFold(n_splits=5, shuffle=True, random_state = 9)
cv_lm_r2s = [] #collect the validation results

for train_ind, val_ind in kf.split(X,y):

    X_train, y_train = X[train_ind], y[train_ind]
    X_val, y_val = X[val_ind], y[val_ind]

    #simple linear regression
    lm = LinearRegression()

    lm.fit(X_train, y_train)
    cv_lm_r2s.append(round(lm.score(X_val, y_val), 3))

print('Simple regression scores: ', cv_lm_r2s, '\n')

print(f'Simple mean cv r^2: {np.mean(cv_lm_r2s):.3f} +- {np.std(cv_lm_r2s):.3f}')
```

```
Simple regression scores:  [0.295, 0.297, 0.355, 0.124, 0.343]

Simple mean cv r^2: 0.283 +- 0.083
```