Hit Movie Project

Group 8

- John McDowell - “Square”
- Jeremy Ocain - “Circle”
- Latifou Amoussa - “Triangle”
- Jose Mendez - “X”

Why Hit Movies?

- We all enjoy movies, but what makes them hits?
- What is our definition of a “hit”?
- This analysis attempts to dissect what is a critical hit based on Metascore data and a Monetary hit based on Net Profit.

Data Sources and Tools Utilized

1. CSV file from Kaggle - IMDb movies.csv
2. Pandas / Python / Sqlalchemy / Scikit-learn
3. SQLite to clean and integrate data
4. Tableau for visualizations and final presentation

Purpose

- Is a movie a hit based on simply Metascore, Gross Income or a mixture of these outputs?
- Is there a seasonality effect in place?
- Does higher budget increase hit probability?
- Has there been a significant change in profitability for movies over the decades analyzed?
- Is there any correlation by genres?

metacritic
MUST-SEE

Definitions / Terminology Used

Critically Acclaimed Movies (Based off of Metascore)

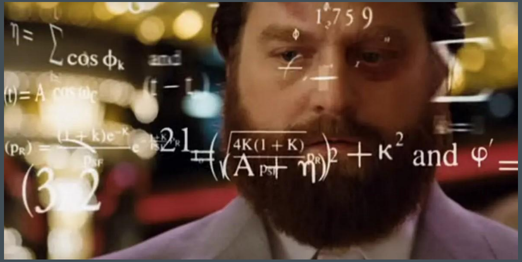
Our group considers a movie to be critically acclaimed (meta_hit) if the movie achieves a metascore of 75 or greater.

Blockbuster movies

Our group considers a blockbuster movie by two stipulations:

- A movie with a budget less than 7 million and having a gross profit greater than or equal to 500 percent.
- A movie with a budget over 7 million and having a gross profit greater than or equal to 250 percent.

The movie data utilized in this project span from the 1980’s to the beginning of 2020



Total Movie Count and Net Profits

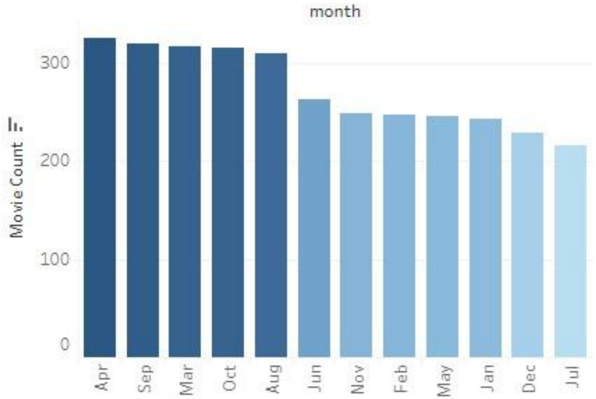
What makes a Hit Movie?

‹ 1 2 3 4 5 6 ›

Year (date) (group)

(All)

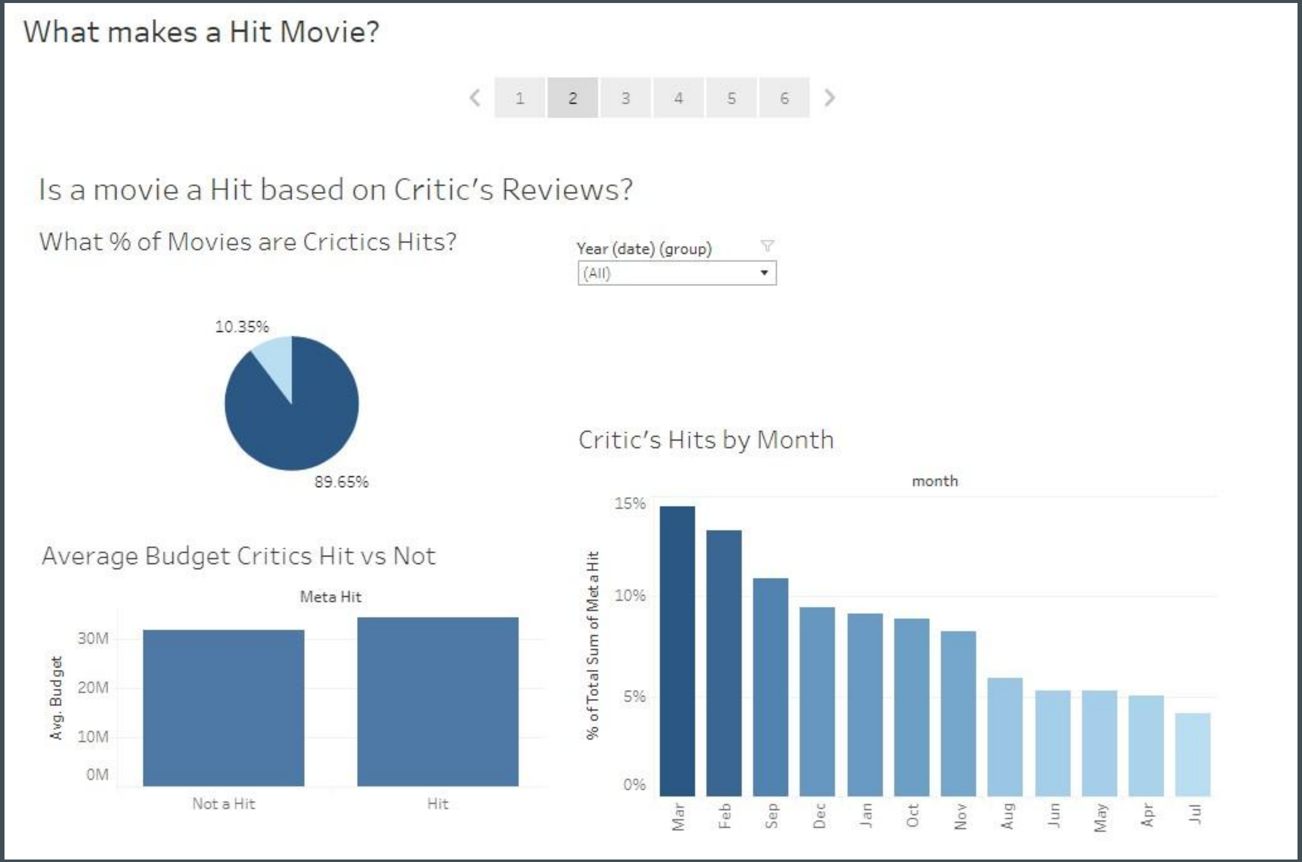
Total movies by month



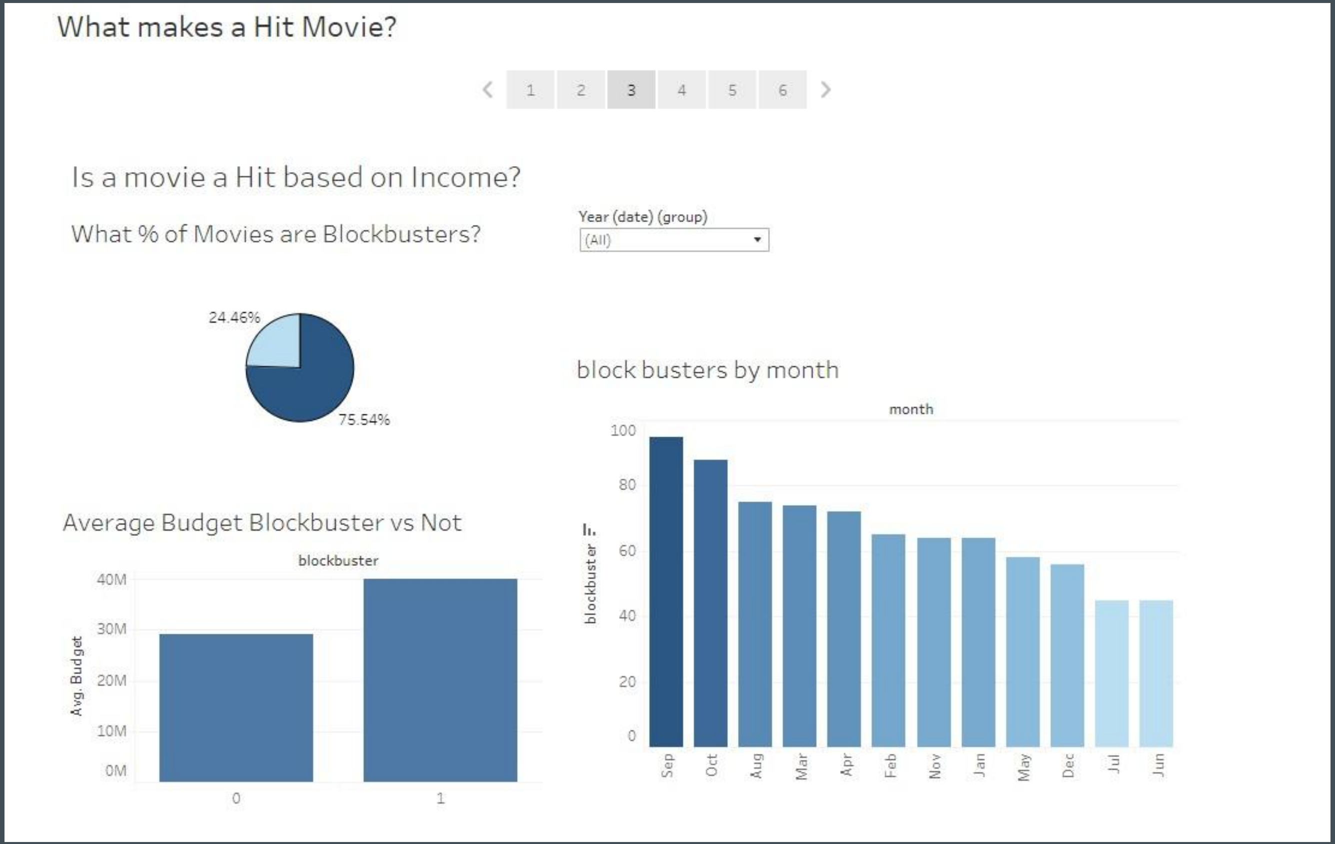
Net Profits by Month

month	
Dec	24,444,711,152
Sep	24,048,720,563
Oct	22,937,406,371
Apr	19,982,549,257
Nov	17,548,030,944
Aug	16,923,874,503
Mar	15,771,014,830
Jan	15,260,219,345
May	15,185,191,105
Feb	13,914,909,765
Jun	13,790,053,036
Jul	8,158,954,406

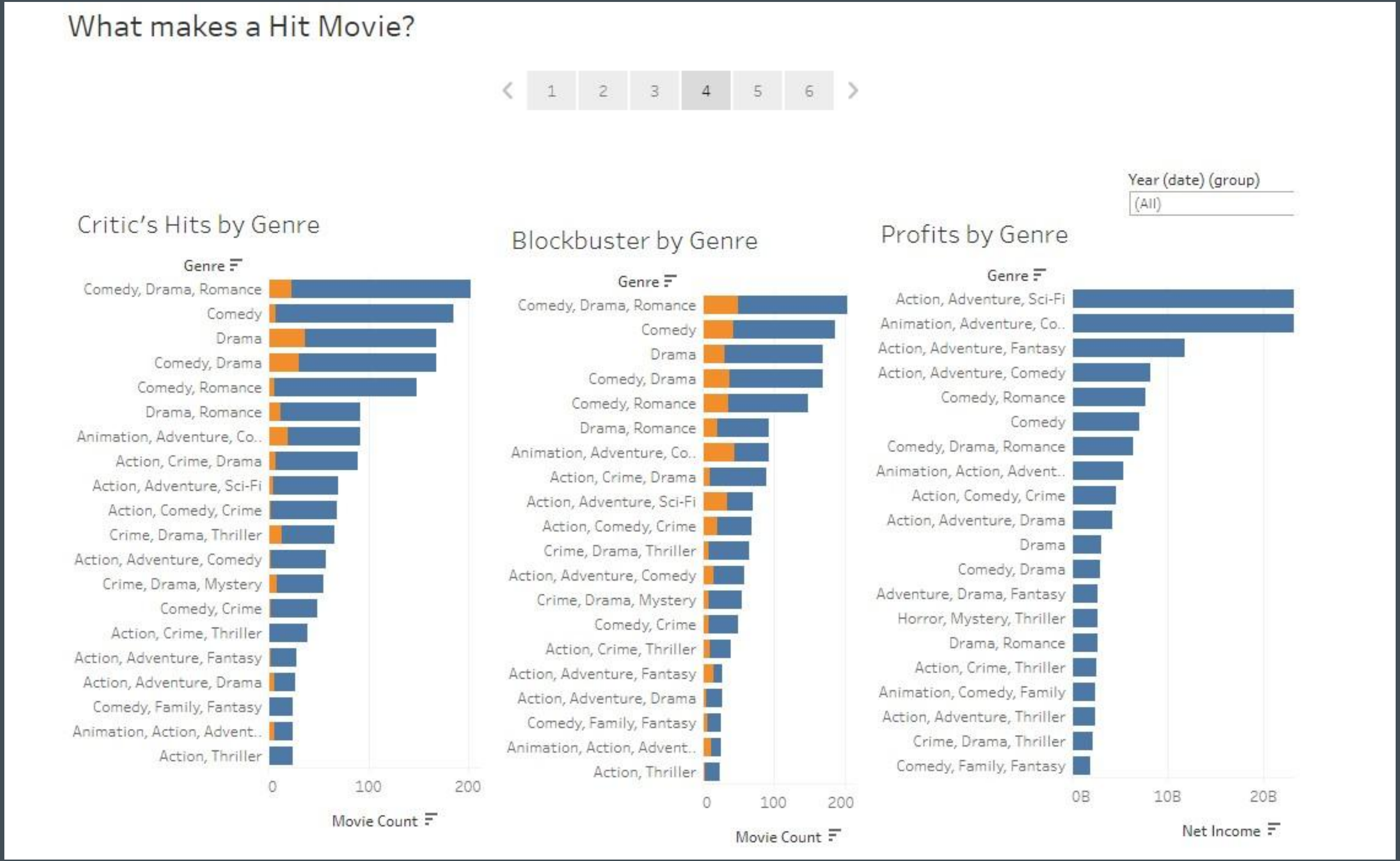
Critical Hits - Budget and Month Data



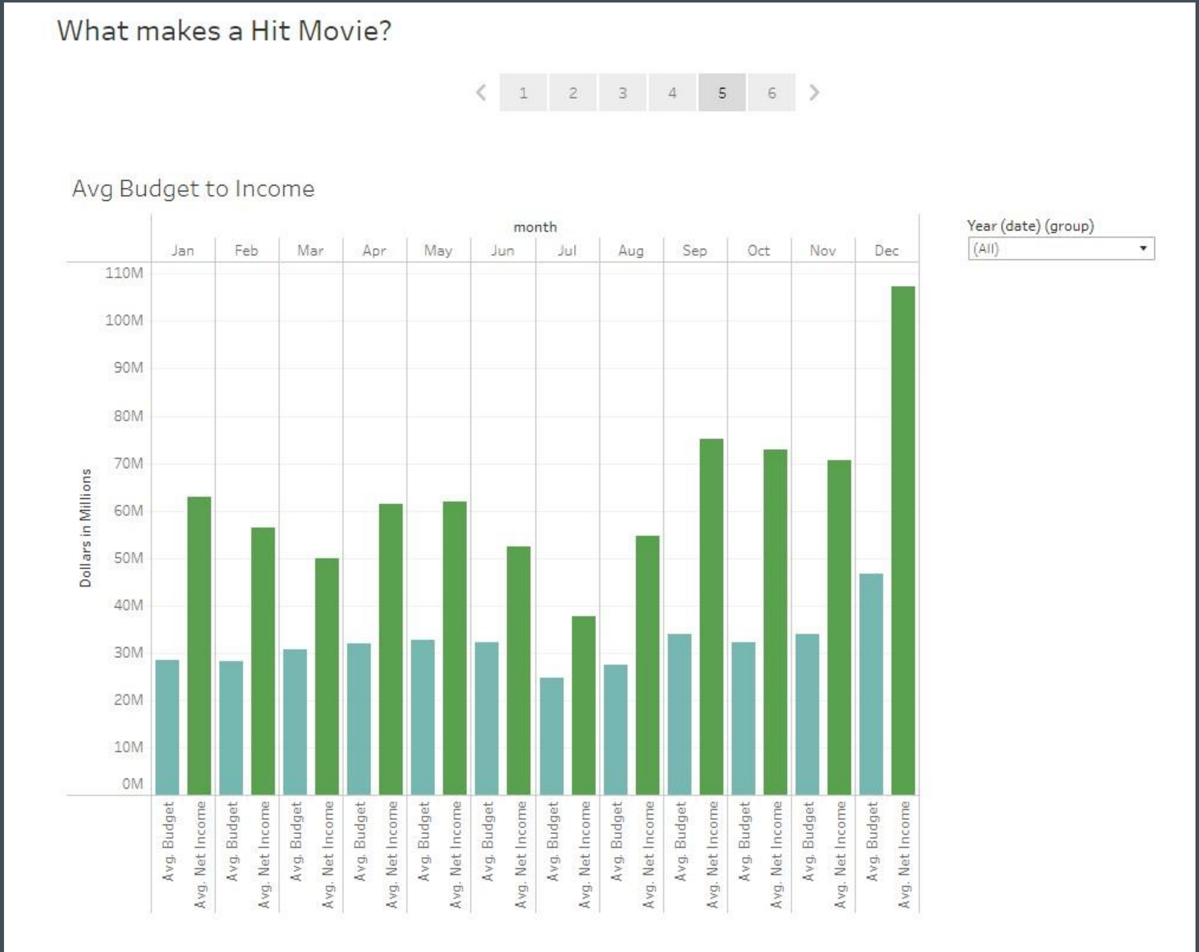
Block-Buster Analysis



Hit Movie by Genre Analysis



Budget Analysis



Machine Learning - Part 1

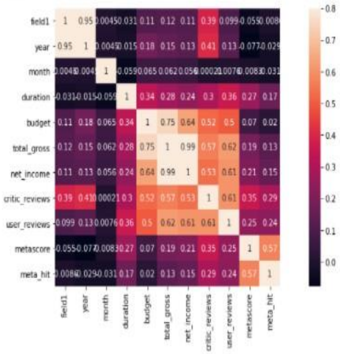
jupyter meta_&_blockbuster_models (autosaved)

```
In [3]: # open a connection to database
        conn = sql.connect('movies.db')
```

```
In [4]: # plot a sns heat map to see what data correlates.
        plt.figure(figsize=(18,8),dpi=100,)
        plt.subplots(figsize=(15,6))
        sns.heatmap(data=imdb_df.corr(),square=True,vmax=0.8,annot=True)
```

```
Out[4]: <AxesSubplot:>

        <Figure size 1800x800 with 0 Axes>
```



Critically Acclaimed Movies (Based off of Metascore)

Our group considers a movie to be critically acclaimed (meta_hit) if the movie achieves a metascore of 75 or greater.

Split data into training and testing

```
In [5]: # seperate the features(X) from the target (y)
        y = imdb_df['meta_hit']
        X = pd.get_dummies(imdb_df.drop(columns='meta_hit'))

        # split data into training and testing
        X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 1, stratify = y)
        X_train.shape
```

```
Out[5]: (2498, 3925)
```

jupyter meta_&_blockbuster_models (autosaved)

Create a MetaHit Logistic Regression Model

```
In [6]: classifier = LogisticRegression(solver='lbfgs', max_iter = 200, random_state = 1)
```

```
In [7]: classifier.fit(X_train, y_train)
```

```
Out[7]: LogisticRegression(max_iter=200, random_state=1)
```

Make Predictions

```
In [8]: y_pred = classifier.predict(X_test)
        results = pd.DataFrame({"Prediction": y_pred, "Actual": y_test}).reset_index(drop=True)
```

```
In [9]: results.head(20)
```

Out[9]:

	Prediction	Actual
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	0	1
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0
13	0	0
14	0	0
15	0	0
16	0	0
17	1	0
18	0	0
19	0	0

```
In [10]: from sklearn.metrics import accuracy_score
         print(accuracy_score(y_test, y_pred))

0.8955582232893158
```

Machine Learning - Part 2

jupyter meta_&_blockbuster_models (autosaved)

Blockbuster movies

Our group considers a blockbuster movie by two stipulations:

1. A movie with a budget less than 7 million and having a gross profit greater than or equal to 500 percent.
2. A movie with a budget over 7 million and having a gross profit greater than or equal to 250 percent.

In [13]:
```
# create a new gross profit column using the Gross profit margin formula

# Gross Profit Margin = (Revenue - Cost) / Revenue x 100
imdb_df['gross_profit'] = (imdb_df['total_gross'] - imdb_df['budget'])/imdb_df['budget'] * 100

# create new blockbuster column
imdb_df['blockbuster'] = 0
imdb_df.head()
```

Out[13]:

	field1	title	year	month	genre	duration	country	language	budget	total_gross	net_income	critic_reviews	user_reviews	metascore	meta_hit
0	4334	Kate & Leopold	2001	3	Comedy, Fantasy, Romance	118	USA	English, French	48000000	76019048	28019048	115.0	341.0	44.0	0
1	19759	Diritto di cronaca	1981	3	Drama, Romance, Thriller	116	USA	English, Spanish	12000000	40716963	28716963	27.0	115.0	64.0	0
2	19774	Arturo	1981	2	Comedy, Romance	97	USA	English	7000000	95461682	88461682	44.0	132.0	69.0	0
3	19790	Blow Out	1981	4	Crime, Drama, Mystery	108	USA	English	18000000	12000000	-6000000	123.0	199.0	86.0	1
4	19804	Libertà poco vigilata	1981	5	Comedy, Drama	94	USA	English	11000000	31261269	20261269	5.0	16.0	55.0	0

In [14]:
```
# split the budget data
BBmovie_lower = imdb_df[imdb_df['budget']<7000000]
BBmovie_over = imdb_df[imdb_df['budget']>7000000]

# Set the blockbuster conditions
BBmovie_lower['blockbuster'] = BBmovie_lower['gross_profit'].apply(lambda x: 1 if x>=500 else 0)
BBmovie_over['blockbuster'] = BBmovie_over['gross_profit'].apply(lambda x: 1 if x>=250 else 0)
```

In [15]:
```
# look at blockbuster counts below a $7,000,000 budget
BBmovie_lower.groupby('blockbuster')['blockbuster'].count()
```

Out[15]:
```
blockbuster
0    664
1    159
Name: blockbuster, dtype: int64
```

jupyter meta_&_blockbuster_models (autosaved)

In [11]:
```
from sklearn.metrics import confusion_matrix, classification_report
matrix = confusion_matrix(y_test, y_pred)
# create a dataframe from the confusion matrix
matrix_df = pd.DataFrame(matrix, index=['Actual Critical Hit', 'Actual Non-Critical Hit'], columns=['Predicted Critical Hit', 'Pr
matrix_df
```

Out[11]:

	Predicted Critical Hit	Predicted Non-Critical Hit
Actual Critical Hit	744	3
Actual Non-Critical Hit	84	2

In [12]:
```
report = classification_report(y_test, y_pred)
print(report)
```

```
              precision    recall  f1-score   support

           0       0.90      1.00      0.94       747
           1       0.40      0.02      0.04        86

    accuracy                           0.90       833
   macro avg       0.65      0.51      0.49       833
weighted avg       0.85      0.90      0.85       833
```

Machine Learning - Part 3

jupyter meta_&_blockbuster_models (autosaved)

In [17]:
```
# combine lower and over into imdb dataframe
imdb_df = pd.concat([BBmovie_lower, BBmovie_over])
imdb_df.head()
```

Out[17]:

	field1	title	year	month	genre	duration	country	language	budget	total_gross	net_income	critic_reviews	user_reviews	metascore	meta_
5	19837	Alla maniera di Cutter	1981	2	Crime, Drama, Mystery	109	USA	English	3000000	1752634	-1247366	43.0	60.0	70.0	
6	19847	Benedizione mortale	1981	8	Horror, Thriller	100	USA	English	2500000	8279042	5779042	114.0	73.0	56.0	
8	19883	1997: fuga da New York	1981	10	Action, Adventure, Sci-Fi	99	USA	English	6000000	25244626	19244626	250.0	343.0	76.0	
11	19918	L'assassino ti siede accanto	1981	4	Horror, Mystery, Thriller	87	USA	English	1250000	21722776	20472776	165.0	437.0	26.0	
12	19947	Il signore della morte	1981	10	Horror	92	USA	English	2500000	25533818	23033818	197.0	561.0	40.0	

Split the data into training and testing

In [18]:
```
# seperate the features(X) from the target (y)
y = imdb_df['blockbuster']
X = pd.get_dummies(imdb_df.drop(columns='blockbuster'))

# split data into training and testing
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 1, stratify = y)
X_train.shape
```

Out[18]: (2457, 3866)

In [19]:
```
# Check the balance of our target values
y.value_counts()
```

Out[19]:
```
0    2476
1     800
Name: blockbuster, dtype: int64
```

Create a Logistic Regression Model for Blockbuster Movies

In [20]:
```
classifier = LogisticRegression(solver='lbfgs', max_iter = 200, random_state = 1)
```

In [21]:
```
# Fit (train) the model using the training data
classifier.fit(X_train, y_train)
```

Out[21]: LogisticRegression(max_iter=200, random_state=1)

jupyter meta_&_blockbuster_models (autosaved)

Make Predictions

In [22]:
```
y_pred = classifier.predict(X_test)
results = pd.DataFrame({"Prediction": y_pred, "Actual": y_test}).reset_index(drop=True)
results.head(20)
```

Out[22]:

	Prediction	Actual
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	1	1
8	0	0
9	0	0
10	0	1
11	0	1
12	1	1
13	0	0
14	0	0
15	0	0
16	0	0
17	0	0
18	0	0
19	0	0

In [23]:
```
# Calculate the balanced accuracy score
from sklearn.metrics import accuracy_score
print(accuracy_score(y_test, y_pred))
```

0.9682539682539683

In [24]:
```
# Display the confusion matrix
from sklearn.metrics import confusion_matrix, classification_report
matrix = confusion_matrix(y_test, y_pred)

# create a dataframe from the confusion matrix
matrix_df = pd.DataFrame(matrix, index=['Actual Blockbuster Hit', 'Actual Non-Blockbuster'], columns=['Predicted Blockbuster Hit
matrix_df
```

Out[24]:

	Predicted Blockbuster Hit	Predicted Non-Blockbuster
Actual Blockbuster Hit	613	6
Actual Non-Blockbuster	20	180

In [25]:
```
report = classification_report(y_test, y_pred)
print(report)
```

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98       619
           1       0.97      0.90      0.93       200

    accuracy                           0.97       819
   macro avg       0.97      0.95      0.96       819
weighted avg       0.97      0.97      0.97       819
```

Conclusions

- Net Profits by Month - Although December shows to be the most profitable month historically spend, the trend for the last 20 years shows April to be the most profitable month.
- The most critical hits come from February and March months, near Academy Awards season.
- There are 33% less movies considered critical hits from the 1980's and 1990's to the 2000's and 2010's.
- The budget for blockbusters has ballooned by an average of 3-fold over the last 20 years (2000s / 2010s) when comparing movies from the (1980's / 1990's).
- The most critically acclaimed movies come out in March and February and are a combination of Drama, Comedy and Romance genres.
- Accuracy Score for MetaHit Logistic Regression Model is 0.8955582232893158
- Accuracy Score for Blockbuster Logistic Regression Model is 0.9682539682539683

Any Questions?
Thank you for your attention!