Devin Belden

C964 Capstone

Western Governors University

Section A: Letter of Transmittal and Project Proposal

Letter of Transmittal

December 21, 2021

Martin Akerfeldt
FHA Consulting
100 E. Millennia Way
Seattle, WA 98121

Mr. Akerfeldt,

The key to any successful investment is planning around the future, and the best way to predict the future is to look at past events. Unfortunately, there are so many factors of past problems that it is often difficult to take all of them into consideration when planning for the future. At FHA, we pride ourselves on creating the best outcomes for our shareholders, and one of the tools we use to do so is computer modeling. Computer models take the guesswork out of prediction, giving our clients more time to focus on planning ahead.

With a future project, we will be able to take our tried and tested techniques for computer modeling, and apply them to the film industry. We will look at past film projects spanning almost 50 years, and use a computer model to predict what that film will gross at the box office and home video. The model will consider factors such as writer, director, studio, cast, and budget in order to make its decisions, and its predictions should fall within 5% of the actual value. This accuracy will allow us and our clients to understand what changes, if any, to make to a film project in order to make it as profitable as possible.

The project will require work hours from engineers and QA specialists. The total cost for such a project is $87,200, with yearly maintenance costing $12,770. Preliminary research suggests that the project will pay for itself within eighteen months, and that the direct return on investment to FHA for this project will be 120% within the first two fiscal years.

To complete the project, I and other engineers will be relying on our collective decade of previous experience in creating similar projects for FHA. These techniques can be found in the

documentation for those solutions, and may be reviewed at your leisure in order to get an idea of how this project will come to fruition.

Sincerely,
Devin Belden

Project Proposal

Problem Description

FHA Consulting has expressed interest in providing our clients with accurate predictions for the gross return of film projects. However, it is difficult to rely on industry knowledge and human intuition as the sole tools to make such predictions. Furthermore, an incorrect prediction could result in a raw financial loss of up to nine figures for our client studios, depending on the budget, as well as the opportunity cost associated with choosing one film to finance over another film that would have been more successful. Therefore, to complement the skillset of our human clients, we propose using a computer model that will consider every facet of a film project in order to predict the film's profit.

Customer Benefit

Our computer model will have the ability to accurately predict the gross revenue of a film project, given factors such as cast, director, and budget, among others. Using this prediction, our clients will be able to make reliable and accurate business decisions, with regards to which film projects to finance, in order to maximize their return on investment. Filmmakers and producers will also be able to understand which key human or other resources to target for project attachment in order to produce the most profitable film.

Data Product Outline

The forward-facing part of the application will ask a user for inputs, such as cast, director, and budget. Once all of the required information is collected from the user, the data product will be able to feed that

information into the machine learning model, which will return an accurate prediction of the dollar amount that the proposed film project will gross. Continuing to tweak the variables asked for by the application will further change the predicted gross amount, allowing the customer to see, in real time, what human or other resources would be most beneficial to the film project as a whole.

Description of Data

For the initial proof of concept, the project will be using data that spans over seven thousand films over forty years. The dataset was originally collected from Kaggle.com and contains twenty columns, such as budget, studio, director, and so on. The dataset will likely contain errors and/or missing values, which will be dealt with via imputing or elimination. Outliers will also be considered carefully, with some being kept and others being eliminated in the pursuit of accuracy. The goal throughout the data cleaning process will be to lean on the conservative side, i.e. to minimize losses before focusing on maximizing profit. Therefore, high outliers will be eliminated before low outliers.

Hypotheses and Objectives

The primary objective of this project is to assign numerical weights to each factor of filmmaking, where each weight represents its contribution or importance to the model's decision making. The higher the value of the factor's weight, the more important that factor is deemed by the model. It is the hypothesis of the engineering team that budget will possess the highest weight among all the individual factors, making it the best predictor of a film's gross.

Project Methodology

Since there is an iterative element to the model creation process that was briefly touched upon in the Data Description section of this proposal, it is the recommendation of the engineering team that this project's development follows the Agile methodology. This methodology welcomes iterative development, and supports revisiting earlier steps in the process with the discovery of new information. As is often the case, model creation begins with a set amount of knowledge regarding the data on the part of the engineering team, and as the process continues, more information about the data is

revealed. Agile has room for improvement of each step built into its methodology, making this the recommended choice for such a project.

Funding Requirements

Given that the development environments used for this project are open-source, and can be run on existing company hardware, neither software nor hardware will incur any cost on the part of FHA. Therefore, the only cost associated with this project is engineering and quality assurance time, along with server bandwidth for continuing support. The total estimated cost for this project is $87,200, with yearly maintenance costing $12,770. This cost includes initial development hours, quality assurance hours, maintenance hours, and server bandwidth service.

Stakeholder Impact

This project, once completed, will allow our clients to input the relevant factors of a film proposal into the user facing application, and a dollar amount representing the proposal's projected gross revenue will be returned to the user. In a future iteration of this project, the completed application will recommend project changes to the client, such as hiring different cast or crew, that will make the predicted value of the proposal increase.

Ethical and Legal Considerations

All the data being collected and used for this project is open source, requiring no security access or prior authorization to obtain. Cast, crew, budget, and studio are all pieces of information that can be obtained from either the credits of a film, or other publicly available sources. However, the marketing budget of a film is less available, and is not distinguished within the dataset chosen for this proof of concept.

Relevant Expertise

The development team will consist of three junior developers led by a senior developer. Additionally, there will be a need for a Quality Assurance analyst. Together, this team has a collective 35+ years of experience with the ideation, development, and maintenance of machine learning models.

Section B: Executive Summary

Problem Statement

FHA's clients have expressed an interest in optimizing revenue streams from film projects. This undertaking, of course, requires an understanding of how to relate a matrix of features of a film project to a vector of gross revenue gained by the film. Recently, our partners and clients have requested that we approach this problem using machine learning techniques on data acquired from external, i.e. public, sources. This data collection includes credits, fan-made databases, and public funding records, including tax information within the public domain. The objective is to use the data to train an ensemble of machine learning algorithms, score the accuracies of the models, and select the highest performing algorithm for further tuning in order to achieve the most optimized result.

Summary of Customers

FHA's customer base includes various studios in charge of approving and financing film projects. Specifically, we will be appealing to producers, underwriters, tax authorities, and up to C-suite executives, in the case of particularly high-budget proposals. There is also room to appeal to the cast and crew of a film project, such as actors and directors, so that they may understand how best to drive their project selection criteria in order to command higher salaries in future projects within their respective careers.

Analysis of Existing Tools

FHA, being a consulting firm, has already proven the use of many of the machine learning techniques and algorithms that will be utilized in this project. Hosting of the project is nothing that needs figuring out, as we can leverage our existing partnerships with cloud service providers. All software will be written using open source languages and packages, complete with the appropriate licenses to ensure that our project particulars, such as model weights, will remain proprietary.

Data Requirements

As this project is specifically designed to be a proof of concept, we will be leveraging existing data from Kaggle.com. Kaggle is a large repository of datasets, which many data scientists and aspiring researchers use to provide demonstrations of techniques, best practices, and even to enter into data science competitions for real-world prizes. Almost no dataset is perfect, however, and many such datasets require extensive exploration and cleaning before running them through a model. Data engineering hours will therefore be allocated appropriately in order to handle this set of tasks.

Project Methodology

As previously mentioned, FHA will be utilizing the Agile methodology for our project development. This method is preferred due to its cyclical nature, wherein each step is revisited as the need arises. This is perfect for data science projects, as running a dataset through a step or set of steps can, and most often does, reveal new information that was not previously accounted for in the previous steps within the data science pipeline. The project can, in the Agile methodology, revisit those previous steps to account for the new information, leading to a more optimized result.

Project Deliverables

The deliverables for the project will include the following:

- Raw and cleaned datasets for client verification and validation
- All trained models used in the process, including the final tuned model
- Source code for the project
- Visual interface for users to interact with the model
- Visual dashboard
- Installation and user manual

Product Implementation and Outcomes

The future plan for this project is for the user interface to be hosted on a cloud service, wherein our clients would receive a persistent link to the product, and they would be able to interact with the product at their leisure and convenience. The target outcome for this project is for it to maximize the profitability of future film projects for our clients.

Project Validation

During the project development lifecycle, the model selected for tuning will be assessed based on its accuracy. More specifically, the model's R-squared score will be observed, and its adaptability to new data will be assessed as well. A cross-validation technique will be utilized to ensure that the model is not

overfit to one “lucky” subsection of data. For the user-centric part of the project, a satisfaction survey will be offered for our users to take in the first iterations of the front end of the project. Later iterations will utilize simple A/B testing in order to expedite this process. Lastly, the model’s accuracy will be continuously evaluated on future film projects, tweaking our model weights accordingly. Additionally, our in-house data research team will use this project to devise novel techniques for this specific regression type, in order to further optimize the prediction process.

Resource Requirements

Hardware

All data procurement and cleaning, model creation and tuning, and user interface design will be done on in house machines. The beta and release candidates will be hosted on a cloud service, which will likely be provisioned by our partners according to our existing contracts.

Software

All software being used for this project will be open source, incurring no additional cost for FHA. This includes the Python language and Jupyter notebook environments. All Python packages planned for use are also open source, including NumPy, Pandas, and Seaborn for visualizations.

Human Resources

Hourly work will comprise the bulk of the cost of this project. As mentioned in the previous sections, the total salary cost for this project is $87,200, with yearly maintenance costs totaling $12,770. Given that this project will be rolled out in episodic fashion, our early estimates say that this project will become net green in 1.5 fiscal years.

Project Timeline

Milestone	Start Date	End Date	Duration	Dependencies	Resources
Data Procurement/ Cleaning	12/13/2021	12/24/2021	2 weeks	None	Engineers
Final Dataset Approval	12/27/2021	12/31/2021	1 week	Data Procurement and Cleaning Complete	QA Specialist
Model Construction and Selection	1/3/2021	1/7/2021	1 week	Final Dataset Approved	Engineers

Model Tuning and Application Development	1/10/2021	1/14/2021	1 week	Model Constructed	Engineers
Final Model and Application Approval	1/17/2021	1/21/2021	1 week	Model Tuned and Application Built	QA Specialist, Cloud Hosting Service
Final Delivery and Verification	1/24/2021	1/31/2021	1 week	Complete Data Product Finished	Client

Section D: Documentation

Business Vision and Requirements

FHA, a company dedicated to providing the latest business intelligence for its corporate clients, has tasked itself with pioneering new technology that will allow it to provide evidence-based financial advice for companies in the film and television industries. Such a partnership with film companies would result in an expansion of FHA's client list, and would represent a diversification of its portfolio. Therefore, FHA aims to leverage its years of proven experience with machine learning and data solutions to propose a method for predicting the future value of such an investment vehicle.

It is the aim of this document to create a data product capable of examining many years of film releases, their budgets, cast, crew, and other information, and predicting that film's gross revenue based on those factors. As this is a proof of concept, only public data will be utilized as a matter of availability. At first, this model will be exploratory in nature, providing valuable insight into the feasibility of such an undertaking, as well as laying bare the need for proprietary and confidential data.

The methodology behind this data product is to observe over forty years of film data, clean the collection, and use it to train one or more machine learning models in order to create a fully-functioning product capable of accurately predicting the gross revenue generated by a hypothetical film project with user-specified parameters.

Raw And Cleaned Datasets

The raw dataset itself was located on Kaggle.com. Click on the following link to view the dataset directly:

[Click here to view the dataset]

The raw dataset is also located within this repository as “movies.csv”. The cleaned dataset is located within this repository under “data_cleaned_and_scaled.csv”. Below are samples of each:

	name	rating	genre	year	released	score	votes	director	writer	star	country	budget	gross	company	runtime
0	The Shining	R	Drama	1980	June 13, 1980 (United States)	8.4	927000.0	Stanley Kubrick	Stephen King	Jack Nicholson	United Kingdom	19000000.0	46998772.0	Warner Bros.	146.0
1	The Blue Lagoon	R	Adventure	1980	July 2, 1980 (United States)	5.8	65000.0	Randal Kleiser	Henry De Vere Stacpoole	Brooke Shields	United States	4500000.0	58853106.0	Columbia Pictures	104.0
2	Star Wars: Episode V - The Empire Strikes Back	PG	Action	1980	June 20, 1980 (United States)	8.7	1200000.0	Irvin Kershner	Leigh Brackett	Mark Hamill	United States	18000000.0	538375067.0	Lucasfilm	124.0
3	Airplane!	PG	Comedy	1980	July 2, 1980 (United States)	7.7	221000.0	Jim Abrahams	Jim Abrahams	Robert Hays	United States	3500000.0	83453539.0	Paramount Pictures	88.0
4	Caddyshack	R	Comedy	1980	July 25, 1980 (United States)	7.3	108000.0	Harold Ramis	Brian Doyle-Murray	Chevy Chase	United States	6000000.0	39846344.0	Orion Pictures	98.0

Raw Dataset

	name	rating	genre	released	score	director	writer	star	country	gross	company	release_date	day_of_year	day_of_week	votes	runtime	budget	scaled_gross
0	The Shining	R	Drama	June 13, 1980	2.080918	Stanley Kubrick	Stephen King	Jack Nicholson	United Kingdom	46998772.0	Warner Bros.	1980-06-13	165	5	2.113362	2.029667	-0.199786	-0.195006
1	The Blue Lagoon	R	Adventure	July 2, 1980	-0.618832	Randal Kleiser	Henry De Vere Stacpoole	Brooke Shields	United States	58853106.0	Columbia Pictures	1980-07-02	184	3	0.462460	-0.125294	-1.251281	-0.123860
2	Star Wars: Episode V - The Empire Strikes Back	PG	Action	June 20, 1980	2.392428	Irvin Kershner	Leigh Brackett	Mark Hamill	United States	538375067.0	Lucasfilm	1980-06-20	172	5	2.273710	0.992100	-0.249042	2.754115
3	Airplane!	PG	Comedy	July 2, 1980	1.354062	Jim Abrahams	Jim Abrahams	Robert Hays	United States	83453539.0	Paramount Pictures	1980-07-02	184	3	1.222679	-1.186551	-1.388390	0.023786
4	Friday the 13th	R	Horror	May 9, 1980	0.004187	Sean S. Cunningham	Victor Miller	Betsy Palmer	United States	39754601.0	Paramount Pictures	1980-05-09	130	5	0.858665	-0.700309	-2.110760	-0.238484

Cleaned Dataset

Data Analysis

The entirety of the code used to construct this data product is located within this repository under “Predicting Movie Revenue.ipynb”. Below are selected highlights, along with explanations of methodology:

```
df_temp = df.copy()
df_temp.dropna(subset=['gross','rating','writer','country','company','runtime'], inplace=True)
df_temp.reset_index(drop=True, inplace=True)
df_temp.isna().sum()
```

Dropping null values. The sum total of nulls in each column involved in the code above was fairly low, so dropping rows was deemed more efficient than imputing missing values, or dropping the columns themselves.

```
date_error = []
dates = df_temp['released'].values
for i in range(len(dates)):
    dates[i] = dates[i][:dates[i].index('(') - 1]
    if len(dates[i].split(' ')) < 3:
        date_error.append(1)
    else:
        date_error.append(0)

df_temp['date_error'] = date_error
df_temp = df_temp[df_temp['date_error'] != 1]
df_temp.head()
```

Identifying and eliminating rows with date errors. There were a select few (<20) rows that did not conform to the traditional YYYY-MM-DD style of release date. Rather than programmatically correct these errors, the rows were simply detected and dropped.

```
df_temp_transform = pd.DataFrame([])

df_temp_transform['votes'] = np.log(df_temp['votes'])
df_temp_transform['runtime'] = np.log(df_temp['runtime'])
df_temp_transform['budget'] = df_temp['budget'] ** (1/3)

df_temp_transform.hist(figsize=(15,15));
```

Applying log and cube root transformations to numerical data. Votes and runtime were heavily skewed, to the point where a log transform was the only transformation aggressive enough to bring the skew sufficiently low enough. Budget, on the other hand, was not so heavily skewed, so a cube root transform was sufficient.

```
scaler = StandardScaler()
df_scaled = df_temp.copy()
```

```
scaling_cols = ['score','votes','runtime','budget']
df_scaled[scaling_cols] = scaler.fit_transform(df_scaled[scaling_cols].values)
df_scaled['scaled_gross'] = scaler.fit_transform(df_scaled['gross'].values.reshape(-1,1))
df_scaled.describe().round(3)
```

Scaling numerical data. The data was scaled based on the column's mean and standard deviation. In the next step, outliers, defined as lying outside three standard deviations from the mean, were eliminated.

```
for col in scaling_cols + ['scaled_gross']:
    df_scaled[col+'_outliers'] = df_scaled[col].apply(lambda x: abs(x) > 3)

df_scaled = df_scaled[(df_scaled['runtime_outliers'] == False) &
                      (df_scaled['budget_outliers'] == False) &
                      (df_scaled['votes_outliers'] == False) &
                      (df_scaled['score_outliers'] == False) &
                      (df_scaled['scaled_gross_outliers'] == False)]
df_scaled.describe().round(3)
```

Eliminating outliers based on Z-score.

```
top_500_directors = list(df_scaled.groupby('director')['gross'].median().sort_values(ascending=False).index[:500])
top_500_writers = list(df_scaled.groupby('writer')['gross'].median().sort_values(ascending=False).index[:500])
top_500_stars = list(df_scaled.groupby('star')['gross'].median().sort_values(ascending=False).index[:500])
top_500_companies = list(df_scaled.groupby('company')['gross'].median().sort_values(ascending=False).index[:500])
```

```
top_500_directors
```

```
'Eric Darnell',
'Corin Hardy',
'Tim Johnson',
'David Yates',
'Michael Patrick King',
'Clay Kaytis',
'Eric Leighton',
'Mike Gabriel',
'Noam Murro',
'Gary Trousdale',
'Paul Tibbitt',
'Jeff Fowler',
'Wes Ball',
'Chris McKay',
'Byron Howard',
'Andy Muschietti',
'Rob Letterman',
'Peter Ramsey',
'David F. Sandberg',
'Tony Bancroft',
```

```
df_filtered = df_scaled[(df_scaled['director'].isin(top_500_directors)) |
                        (df_scaled['writer'].isin(top_500_writers)) |
                        (df_scaled['star'].isin(top_500_stars)) |
                        (df_scaled['company'].isin(top_500_companies))]
df_filtered.reset_index(drop=True, inplace=True)
print(len(df_filtered))
df_filtered.head()
```

```
4082
```

Identifying top 500 grossing cast and crew, and eliminating rows which do not contain at least one of those. This was a difficult decision to make, but it was made owing to the capabilities of the machine hosting this project. Not having enough computing power to handle each unique cast and crew member in a timely manner is a pain point illustrated elsewhere in this document, along with plans to recreate this project with different, more accurate data, and a more powerful computer.

```
categorical_cols = ['rating','genre','director','writer','star','country','company']
df_categorical = pd.get_dummies(df_filtered, columns=categorical_cols)
df_categorical.head()
```

Creating one-hot encoded columns for categorical data. This is a standard practice when dealing with categorical data. It did, however, represent a challenge when creating the user-facing side of the data

product. One-hot encoding created several thousand extra columns, which is unwieldy when asking a user for data to feed into the model. This was solved by translating the user's input into a column name that might be found within the original dataframe, and putting a 1 as the column's value, leaving all other columns of that type as 0. This does not lend itself at all to typos or incorrect spelling of names, and plans for so-called "fuzzy matching", employed by all major search engines, should be employed if possible.

```
X = df_categorical.drop(['name','released','release_date','gross','scaled_gross'], axis=1).values
y = df_categorical['gross'].values

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.1, random_state=13)
```

```
from xgboost import XGBRegressor

xgb = XGBRegressor()
xgb.fit(X_train,y_train)
```

```
[18:23:45] WARNING: src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
```

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0,
             importance_type='gain', learning_rate=0.1, max_delta_step=0,
             max_depth=3, min_child_weight=1, missing=None, n_estimators=100,
             n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
             silent=None, subsample=1, verbosity=1)
```

```
y_pred = xgb.predict(X_test)
from sklearn.metrics import r2_score
score = r2_score(y_test, y_pred)
```

```
score
```

```
0.7098414445867061
```

Creating preliminary gridsearch model candidate. The highest scoring of the vanilla (untuned) models was chosen for a gridsearch methodology, commonly employed for hyperparameter tuning. This was the first step that painfully pointed out the weaknesses of the machine running this project. Taking several minutes to run just one model meant that the machine would have to run the model for several hours or longer for a gridsearch to be measurably effective.

```
from sklearn.model_selection import GridSearchCV

go_ahead = input("Cell will take several minutes/hours to run. Do you wish to run this cell (y/n)? ")

if go_ahead == 'y':

    xgb = XGBRegressor()
    grid = {'gamma': [0, 0.3, 0.7],
            'learning_rate': [0.1,0.01],
            'min_child_weight': [3, 5, 10],
            'subsample': [0.7, 0.8, 0.9]}

    gridsearch = GridSearchCV(xgb, param_grid=grid, cv=5)

    xgb_cv = gridsearch.fit(X_train, y_train)

    best_params = xgb_cv.best_params_

    print(best_params)

else:

    best_params = {'gamma': 0,
                   'learning_rate': 0.1,
                   'min_child_weight': 10,
                   'subsample': 0.8}

    print(best_params)
```

```
Cell will take several minutes/hours to run. Do you wish to run this cell (y/n)? n
{'gamma': 0, 'learning_rate': 0.1, 'min_child_weight': 10, 'subsample': 0.8}
```

```
xgb_grid = XGBRegressor(**best_params)
xgb_grid.fit(X_train, y_train)

y_pred = xgb_grid.predict(X_test)
score = r2_score(y_test, y_pred)
```

```
[18:28:08] WARNING: src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
```

```
score
```

```
0.73089157376432
```

Tuning best gridsearch candidate model and rescoring. After gridsearching for the best model hyperparameters, the model's accuracy was only increased by 3%. In real time, this gridsearch model took over two hours to run. As mentioned before, with a more powerful machine, this could be reduced by as much as 80%, which would allow this project to employ a much wider range of candidate values per hyperparameter.

```
X_columns = list(df_categorical.drop(['name','released','release_date','gross','scaled_gross'], axis=1).columns)
X_columns
```

```
import operator

rankings = list(zip(xgb_grid.feature_importances_, X_columns))
rankings.sort(key=lambda x: x[0], reverse=True)
rankings[:40]
```

```
[(0.18734801, 'budget'),
 (0.12513888, 'votes'),
 (0.07654247, 'rating_R'),
 (0.05185122, 'genre_Animation'),
 (0.049464196, 'rating_PG-13'),
 (0.042052303, 'rating_PG'),
 (0.029348886, 'genre_Crime'),
 (0.018825693, 'score'),
 (0.01849834, 'company_New Line Cinema'),
 (0.017731905, 'country_United States'),
 (0.016042145, 'genre_Comedy'),
 (0.015712176, 'genre_Action'),
 (0.01530857, 'genre_Adventure'),
 (0.014284567, 'day_of_year'),
 (0.013571254, 'runtime'),
 (0.013176132, 'star_Johnny Depp'),
 (0.01312132, 'company_DreamWorks Animation'),
 (0.012951748, 'star_Julia Roberts'),
 (0.012904819, 'country_China'),
```

Obtaining a list of features deemed most important to the model. This is a common practice, especially for datasets with large amounts of categorical data. It is beneficial to know what the computer model deems most important in its decision making, so that we humans can adjust our approaches accordingly.

Hypothesis Assessment

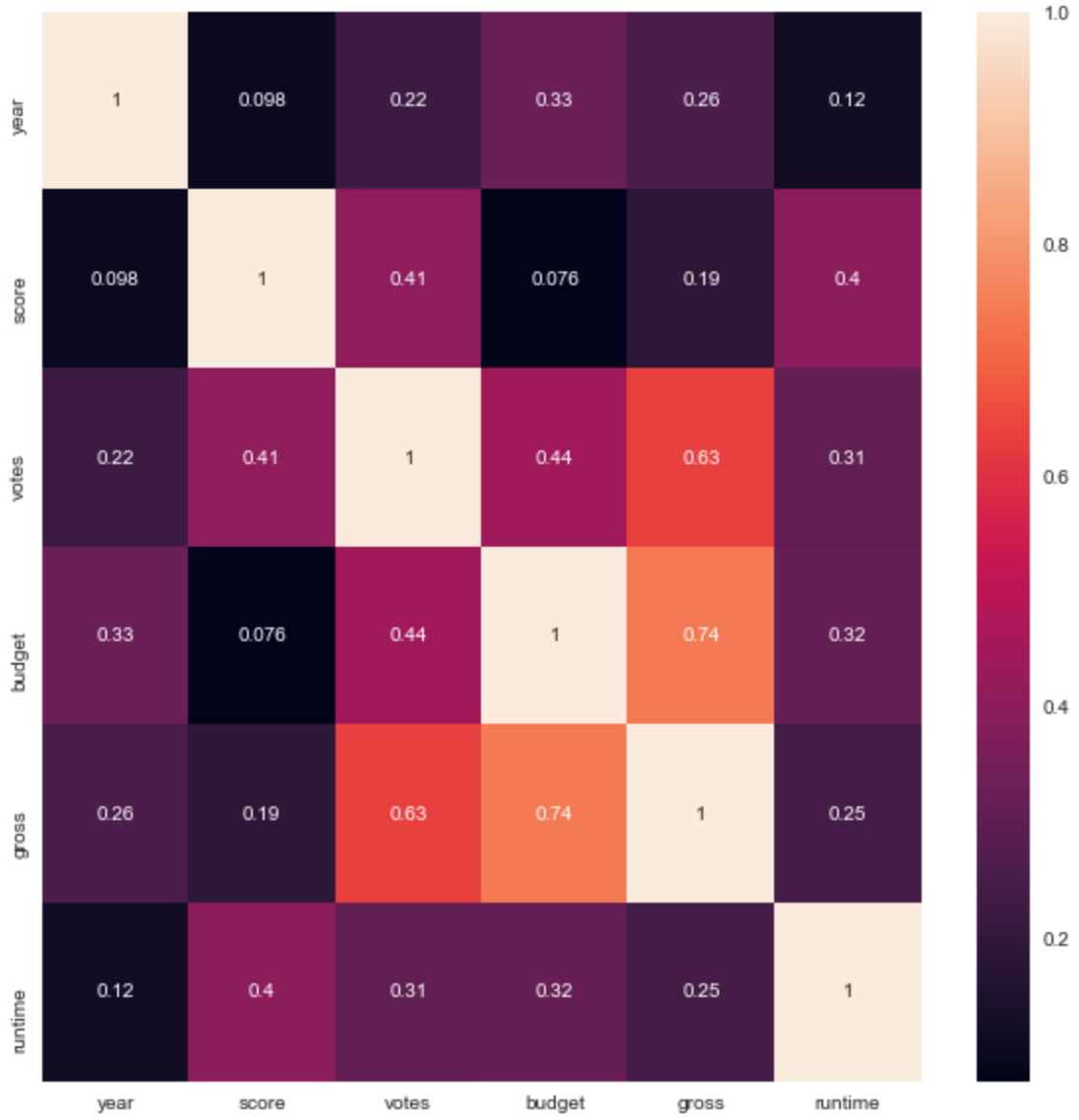
The hypothesis driving the team throughout this project was that a machine learning algorithm would be able to accurately predict the gross of a film project, given various parameters including cast, crew, and budget. As written before, the goal was to attain 95% accuracy with a machine learning model, whereas this proof of concept has achieved 73% accuracy using a wide variety of models. Higher accuracy could have been achieved by utilizing one or more machines with more computing power; this would eliminate the need to filter out such a large number of cast and crew members that are not already highly profitable. Additionally, the more powerful machine would be able to search for a much wider array of hyperparameters than were searched for in this project.

The next steps to be taken involve obtaining more data, perhaps through a different source, and employing the use of various other types of models, such as a neural network. For the data source, it would be beneficial to look for data including more than just one actor, or data that includes other crew members, such as the producer. More steps should be taken to validate the dataset chosen than were employed in the case of this project, as well.

Visualizations

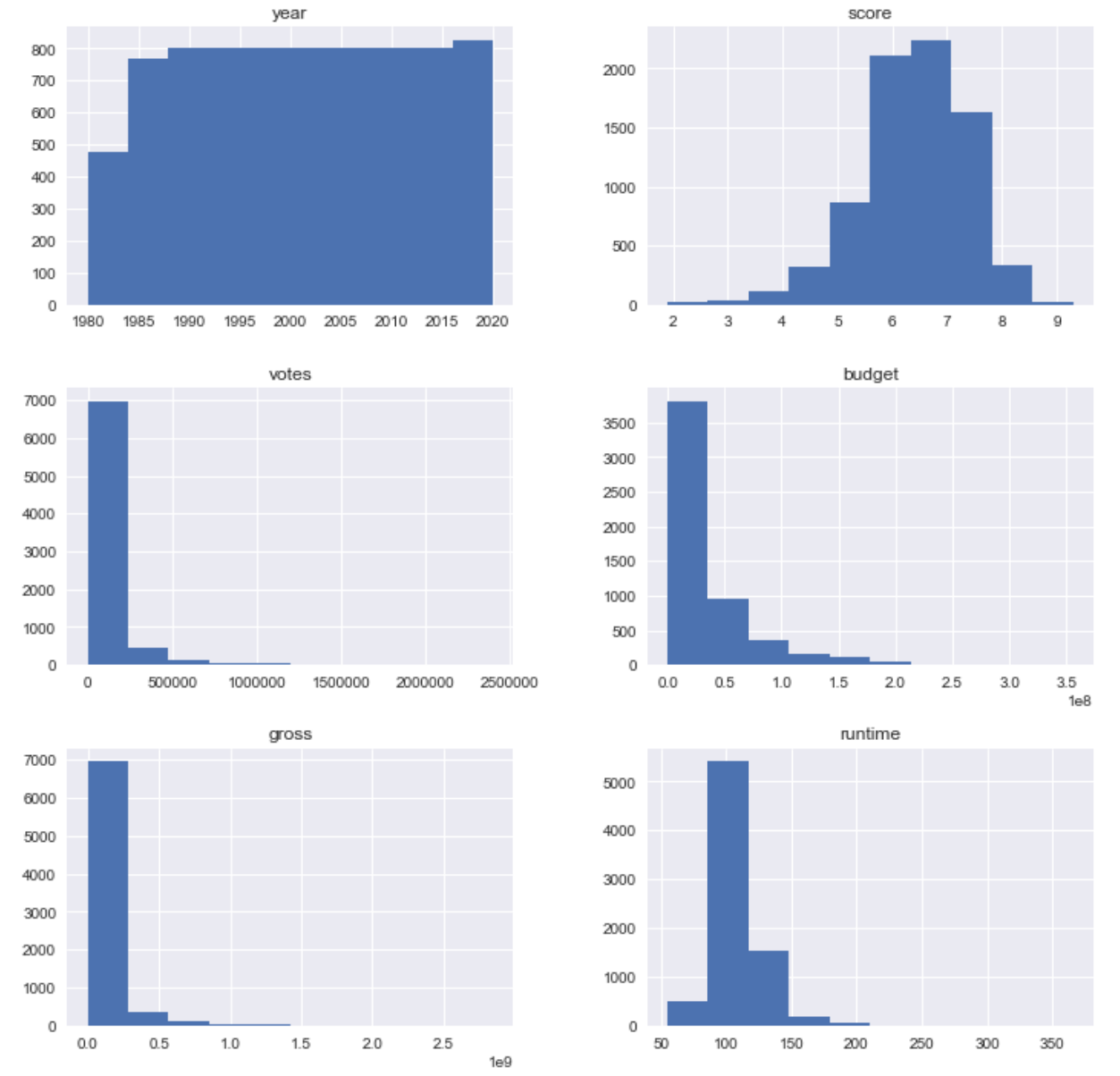
All visualizations, along with the code that created them, are as follows:

```
plt.figure(figsize=(10,10))
sns.heatmap(abs(df.corr()), annot=True);
```



Correlation Heatmap of the Numerical Columns of the Raw Data

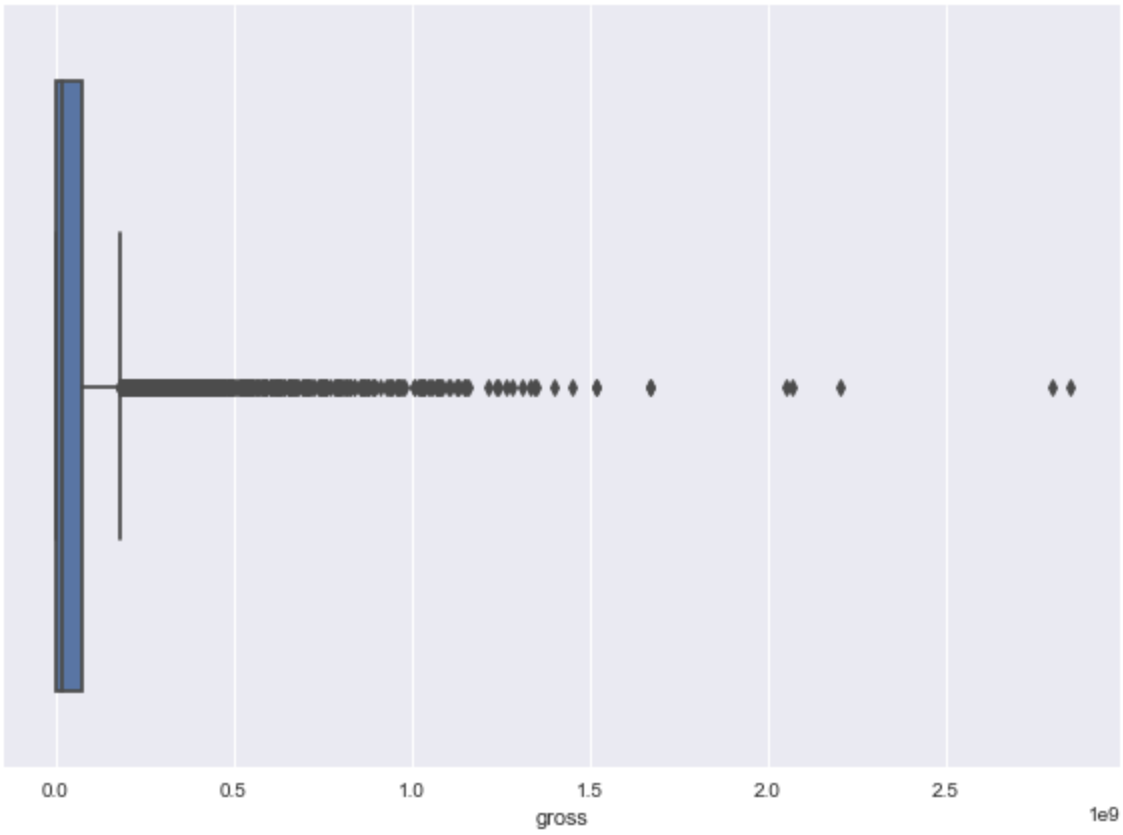
```
1  df.hist(figsize=(12,12));
```



Histograms of the Numerical Columns of the Raw Data to Identify Skew

```
1  plt.figure(figsize=(10,7))
2  sns.boxplot(x='gross', data=df)
```

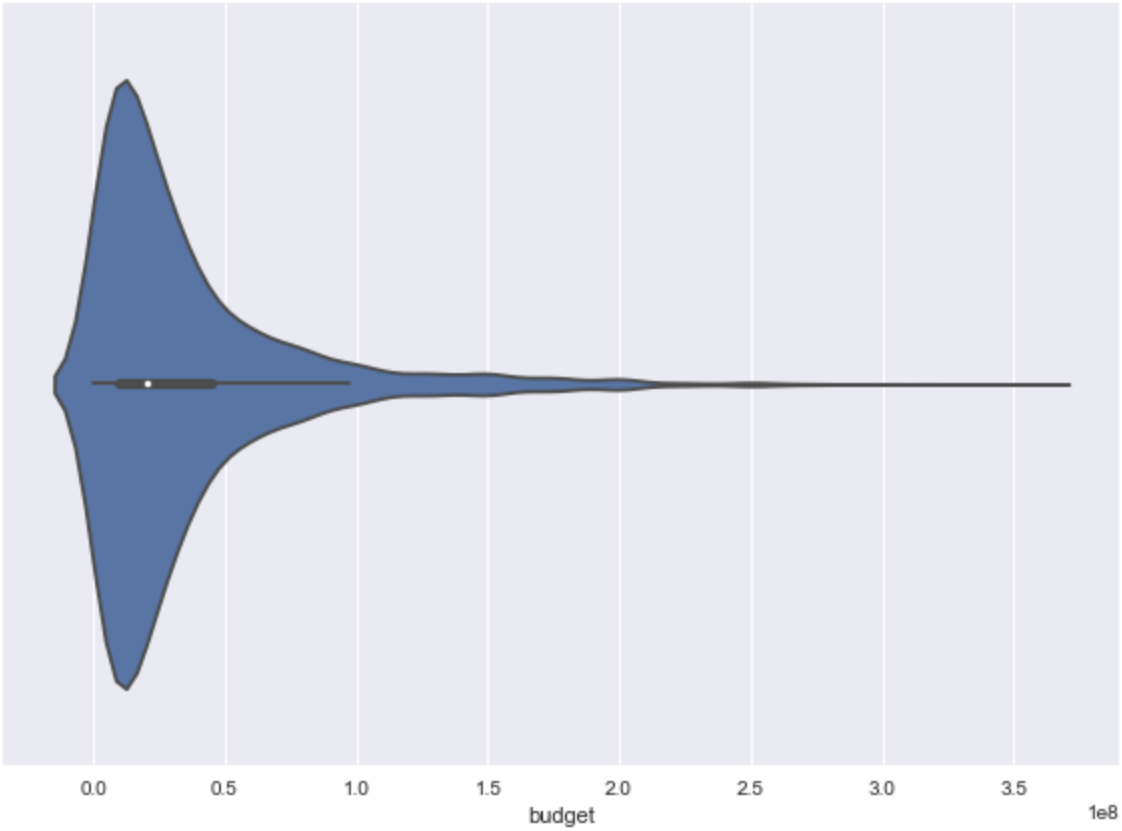
```
<matplotlib.axes._subplots.AxesSubplot at 0x1796ba28828>
```



Box and Whisker Plot to Identify Skew and Outliers Within the Target Column

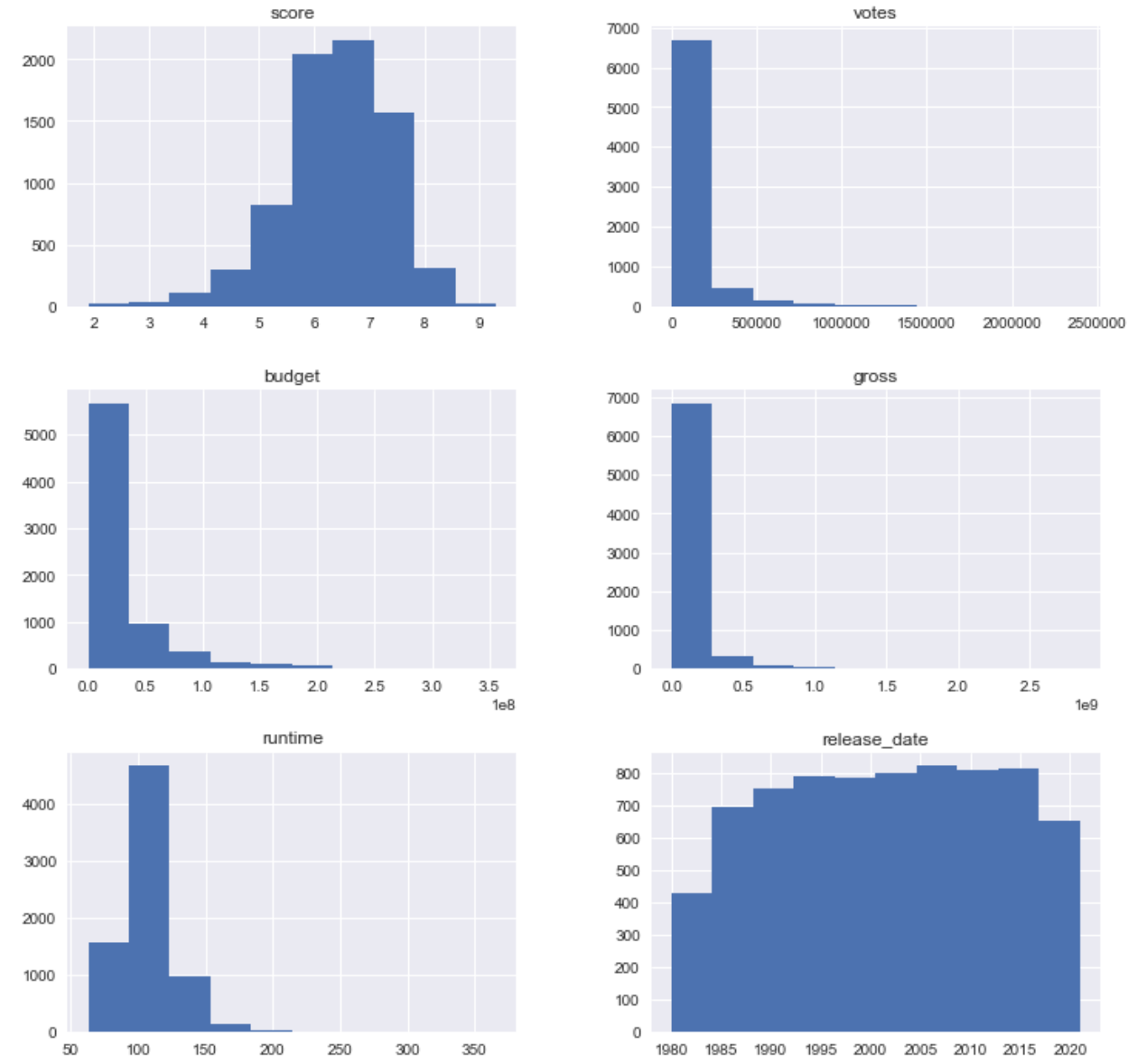
```
1  plt.figure(figsize=(10,7))
2  sns.violinplot(x='budget', data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1796ddf2b00>
```



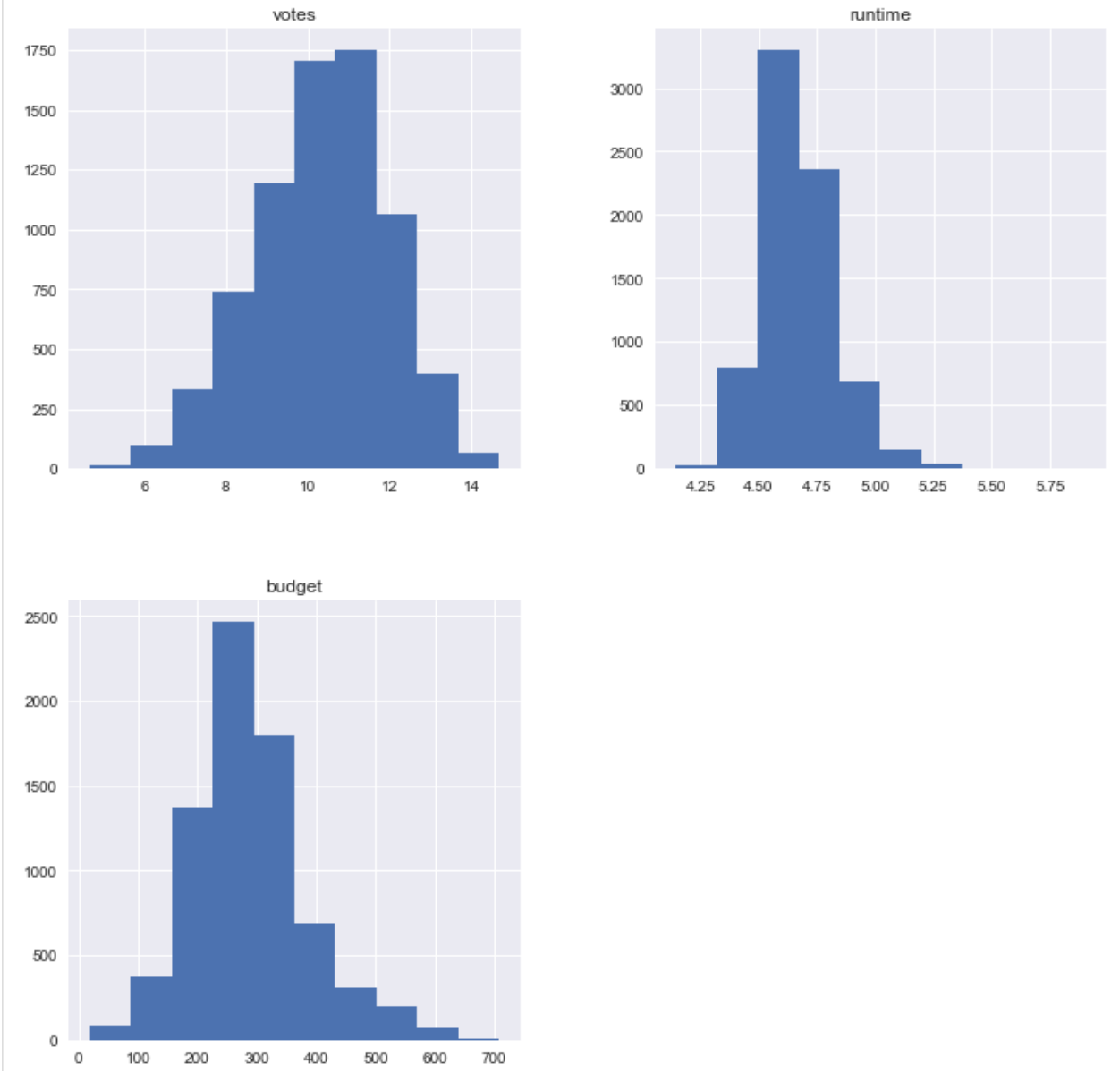
Violin Plot to Visualize Skew of the Budget Column

```
1  df_temp.hist(figsize=(12,12));
```



Histograms of Imputed Untransformed Dataset

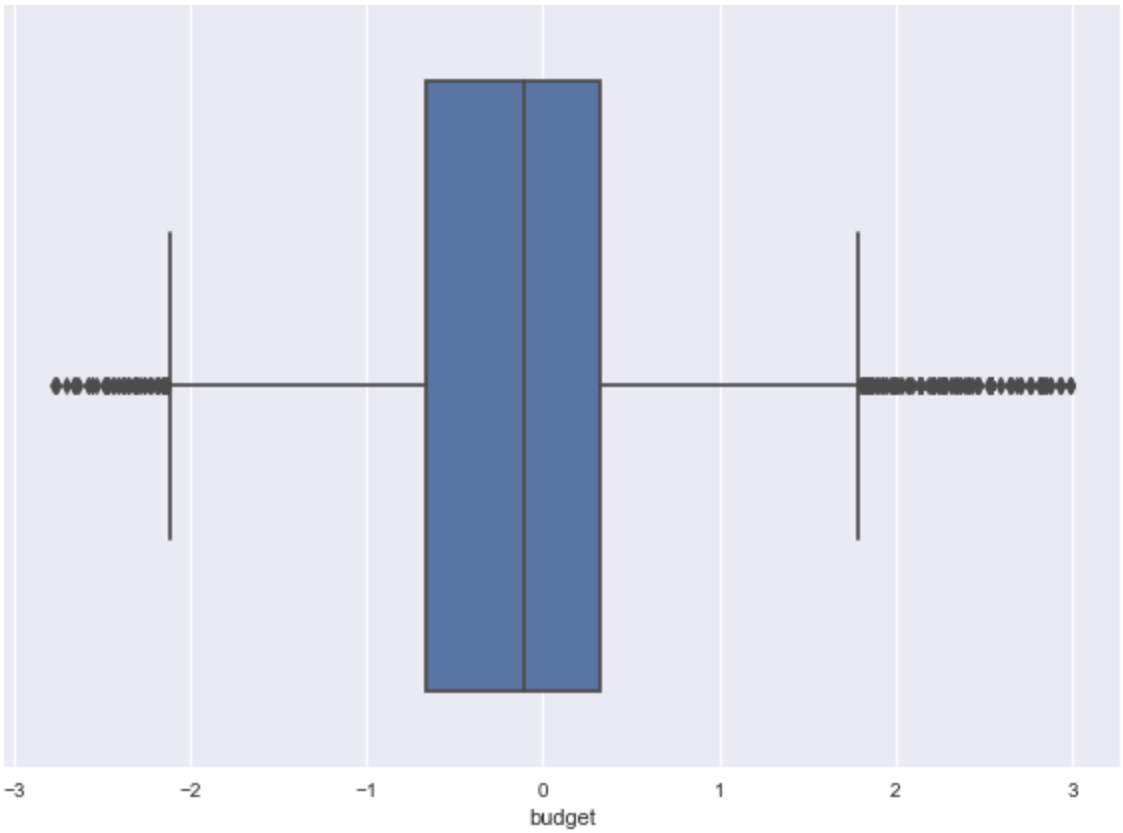
```
df_temp_transform = pd.DataFrame([])

df_temp_transform['votes'] = np.log(df_temp['votes'])
df_temp_transform['runtime'] = np.log(df_temp['runtime'])
df_temp_transform['budget'] = df_temp['budget'] ** (1/3)

df_temp_transform.hist(figsize=(12,12));
```



Histograms of Columns Post-Transformations

```
1  plt.figure(figsize=(10,7))
2  sns.boxplot(x='budget', data=df_scaled)
```

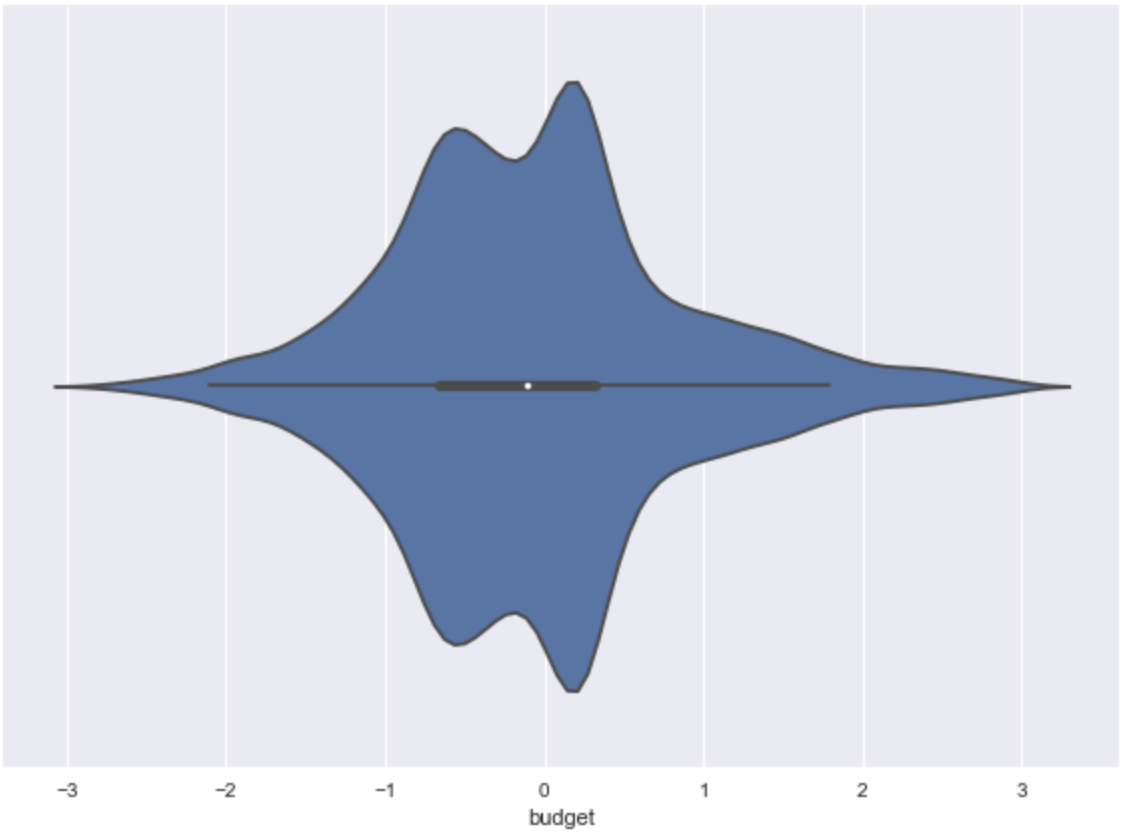
```
<matplotlib.axes._subplots.AxesSubplot at 0x1796b97c748>
```



Box and Whisker Plot of Budget Column Post-Scaling

```
1  plt.figure(figsize=(10,7))
2  sns.violinplot(x='budget', data=df_scaled)
```

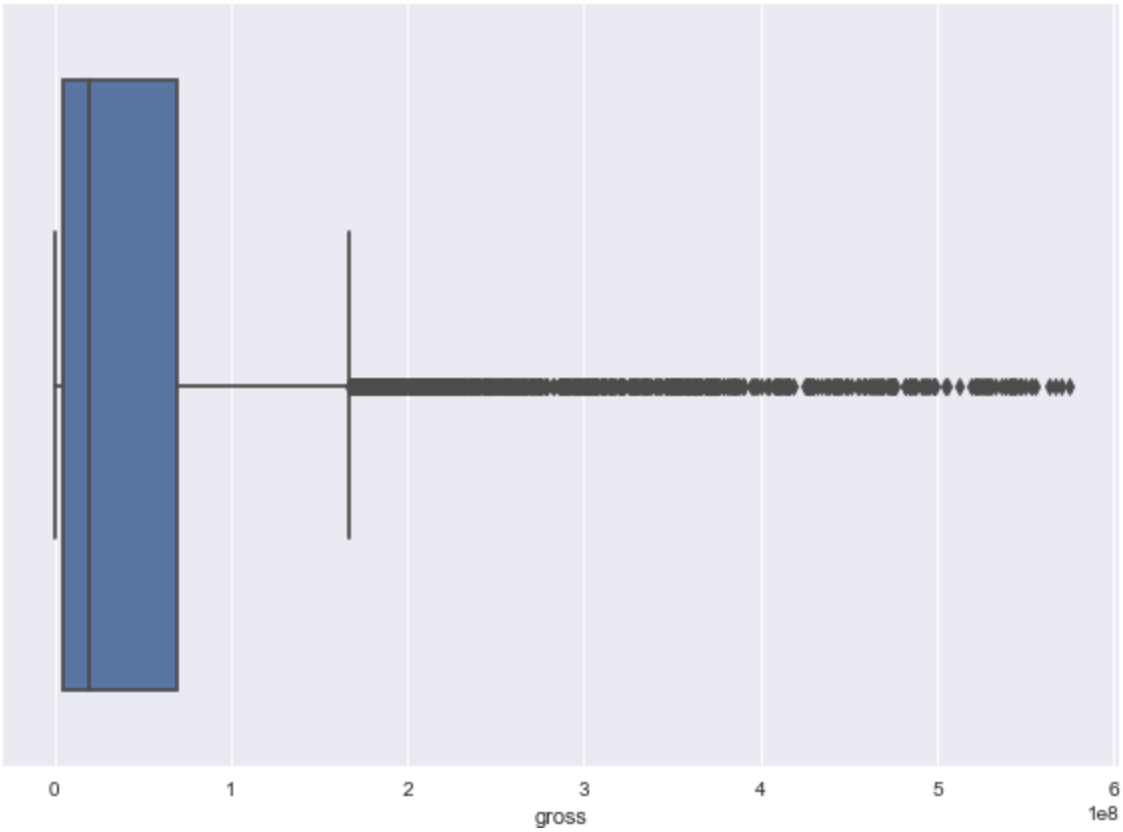
```
<matplotlib.axes._subplots.AxesSubplot at 0x1796b616940>
```



Violin Plot of Budget Column Post-Scaling

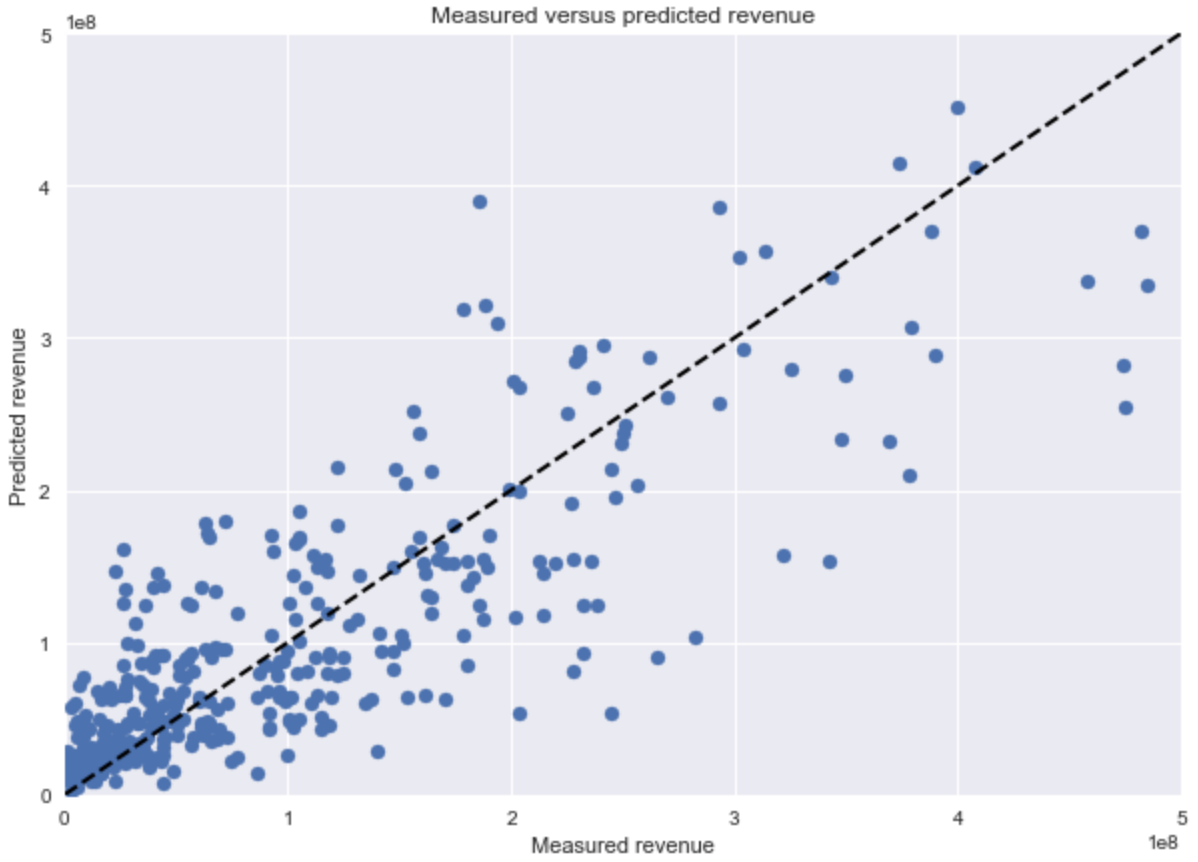
```
1  plt.figure(figsize=(10,7))
2  sns.boxplot(x='gross', data=df_scaled)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1796ee0b780>
```



Box and Whisker Plot of Target Column Post-Scaling of Other Columns

```
fig, ax = plt.subplots(figsize=(10,7))
ax.scatter(y_test, y_pred)
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=2)
ax.set_xlabel('Measured revenue')
ax.set_ylabel('Predicted revenue')
plt.title('Measured versus predicted revenue')
plt.ylim((0, 500000000))   # set the ylim to bottom, top
plt.xlim(0, 500000000)     # set the ylim to bottom, top
plt.show()
```



Scatter Plot of the Final Tuned Model's Predicted Gross to the Actual Gross

Accuracy Assessment

The highest standard R-squared value attained by any model within the data product is 73.1%, meaning that 73.1% of the variance in the data is explained by the features within the dataset; the remaining variance is explained by features outside of the product's purview. This may include things like additional cast and crew members, marketing budget and methods, world events on release day, the effects of COVID, or any number of other factors.

Testing, Revisions, and Optimization

Throughout the project, unit testing was performed on each individual block of code to ensure a baseline level of functionality. Blocks of code included something as simple as importing relevant packages, all the way to constructing, training, and scoring a machine learning model. Further unit testing was performed once later code blocks were written to ensure compatibility and proper setup for the later code blocks. In essence, the code was written and rewritten to ensure that the entire project could be run from start to finish in one kernel session, provided all the package requirements were met.

Once all code blocks were written, and the project arrived at the state of being run from top to bottom, regression testing was performed in order to ensure the elimination of semantic and logical errors. In other words, unit testing eliminated system errors, and regression testing was employed to make sure the code's output fell well within the expected norms.

After all regression testing was completed, the entire system was tested from top to bottom, with values changed in key places, to ensure that the code was able to handle unexpected outputs, especially from its user-facing side.

Finally, all deliverables were run through acceptance testing to ensure that the project satisfied all agreed upon requirements.

Source Code

The entirety of the project's source code is located within this repository, under the file name "Predicting Movie Revenue.ipynb".

Quick Start Guide

1. Download and install Python version 3.7 (Link, scroll to bottom to find the appropriate release for your machine)
2. Download Tableau Public (Link)
3. Within a Terminal (Mac) or Command Prompt (Windows), navigate to the location of this repository
4. Run the following command
    a. Mac: `pip install -r requirements.txt`
    b. Windows: `python -m pip install -r requirements.txt`
5. Once all packages have been installed, run the following command
    a. Mac: `jupyter notebook`
    b. Windows: `python -m jupyter notebook`
6. This repository should open in a web browser. Click on "Predicting Movie Revenue.ipynb"
7. At the top toolbar, click "Cell", followed by "Run All"

    a. Note that one cell within the notebook asks for permission to run before running
8. Once all cells have run, interact with the sliders and text bar widgets at the bottom of the notebook to interact directly with the model
9. To visualize the included dashboard, open the file titled “Capstone_Dashboard.twbx” within Tableau Public