Machine Learning Engineer Nanodegree

Capstone Project

Doug Brown
March 9, 2019

I. Definition

Project Overview

This project originates from a Kaggle challenge that is available at https://www.kaggle.com/c/tmdb-box-office-prediction

The entertainment industry is regularly creating new content. This is a heavy investment, with the businesses producing the content continually seeking ways to optimize their returns. Using predictive analytics is one way companies are using to help refine their decision making processes. The amount of data available to entertainment businesses for each property they create is vast, so if they can reach into that data and determine winning formulas they may be able to improve their revenue.

Netflix is a perfect example of this type of behavior. Based on their user's viewing habits they determined that their audience had high preferences for certain actors, writers, directors, and genres. As an experiment with their data they took this information and put a specific selection of individuals from these areas together to create a show. There was no concept for the show, just the desire to get a particular combination of people together with the expectation, based on the analytics, that the results would be a hit. From that experiment the series *House of Cards* was produced.

For this project there will be work done to see if similar insights can be made regarding movies produced and distributed through theaters. This is a global market that all movie makers are continually seeking to create greater reach in terms of audience and revenue, and the public domain data available for this kind of exercise is considerable.

Problem Statement

The quesion then is, given all the relevant, historical data for a selection of movies, can a model be created for predicting the performance of other films based on similar data. Performance in this case is measured by box office revenue, which is the true metric by which studios measure the success of their products.

From the Kaggle competition a dataset of movies is made available. There are ~3000 movies in the dataset that crosses years, genres, countries, languages, directors, actors, and practically every other descriptive data related to each film. Ideally, a model can be constructed that can consider these factors and use them to make a well-educated guess about the likely revenue for a film.

At a high level, this problem will be approached by

- Acquiring the data
- Doing any necessary pre-processing of the data

- Splitting the data into training and testing sets
- Developing the model based on predicting revenue
- Training the model using the training dataset
- Evaluating the model using the testing dataset
- Drawing conclusions regarding possible improvements to the model

Metrics

Judging the success of the model will be done using the Kaggle competition metric of Root Mean Log Squared Error. Since we are dealing strictly with financial outcomes (revenue) we can make good comparisons based on the disparity between our predictions and true results. Root mean log squared error will allow for taking a group of predictions and generally evaluating their accuracy again the actual results. Additionally, the Root Mean Square Log Error avoids penalizing large differences between results if the numbers being considered are also large. So small differences between small numbers and large differences between large numbers will be treated similarly.

No model will ever be perfectly able to predict the outcomes, which would require perfect knowledge of all variables both inside and outside the control of the movie production efforts. However, a model that can provide good, general guidance should be possible, and this metric will let us gauge this model.

II. Analysis

Data Exploration

The data for this project is very robust. It comes in the form of a csv file culled from The Movie Database and hs 3000 movies. The dataset provided contains the following attributes (see the appendix for a sample data record):

- belongs_to_collection: This field contains an array of data objects describing what series the film belongs to, including the unique series ids and names, if it is part of a series (i.e. *Star Wars*, *Jurassic Park*, etc.)
- budget: This field is an integer value representing the production budget for the film
- genres: This field contains an array of data objects describing the genres with which the film is categorized, including the unique genres ids and names
- homepage: This field contains a string representing the url of the marketing homepage for the film, if one existed
- imdb_id: The film's id within the Internet Movie Database index
- original_language: The string value of the ISO language value for the spoken language in the film's original distribution
- original_title: The string value of the title originally used for the film, which may be different than the commonly known title
- overview: The string representation of the film's summary
- popularity: A floating point value representing the popularity of the film
- poster_path: A string representing the url for an image of the movie's poster, if one is available
- production_companies: This field contains an array of objects describing the production companies involved in the film, including the unique id and name of the company
- production_countries: This field contains an array of objects describing the countries in which the film was produced, including the ISO id and name of the country
- release_date: A string representation of the date on which the film was originally released
- runtime: An intiger value for the runtime of the film in minutes
- spoken_languages: This is an array of objects representing the languages spoken in the film, with each item containing both the ISO id and name of the language
- status: A string representation of the film's status, either released or rumored
- tagline: A string representation of the marketing tagline for the film if there was one
- title: A string repreentation of the film's title at release
- keywords: This is an array of objects representing the keywords associated with the film, with each object containing both the unique id for the keyword and the string for the keyword

- cast: This is an array of objects representing all the actors associated with the film, with each object containing both the unique id and name of the actor along with the order of billing for each
- crew: This is an array of objects representing all the crew members associated with the film with each object containing both the unique id and name of the crew member along with their role in the film's production
- revenue: An integer value representing the global box-office revenue in US dollars

Each of the fields containing arrays of objects are presented as string values with the appearance of a JSON array. For example **[{'id': 313576, 'name': 'Hot Tub Time Machine Collection', 'poster_path': '/iEhb00TGPucF0b4joM1ieyY026U.jpg', 'backdrop_path': '/noeTVcgpBiD48fDjFVic1Vz7ope.jpg'}]**

Given that this format actually represents possibly multiple values of significant interest relating to the film it cannot be analyzed as-is. These field types will definitely need to be transformed. Fields such as *keywords*, *crew* and *actors* likely represent thousands of values each. The *actors* field does provide an attribute indicating the order of billing for cast members, so in this case how many actors are being considered could be reduced if we only consider the few top-billed members of each film. These values cannot be directly one-hot encoded, so additional work will be required and all null instances will need to be filled as 0.

All string fields require some type of transformation to be useful in analysis. Fields such as *tagline*, *overview*, and the transformed *keywords* could be one-hot encoded for consideration. Again, this will certainly lead to many thousands of data points.

The date field is a string and will require transformation into numeric values that are useful for analysis. In this case I will choose to represent them as *week* and *year*. The *week* will represent the number of the week (0-51) in which each film was released. Since films are commonly released on a Friday it is assumed it will be more useful to know which week the film was released rather than which specific month / day.

Some fields, such as imdb_id, have no meaningful data for doing predictive analytics. The following fields will be dropped from consideration:

- imdb_id: This is an index key to another database and is not meaningful
- poster_path: This is a url for a movie poster which will not be visually analyzed here
- status: Since all films are assumed to have been released this is ignored (three films were mis-categorized as rumored)

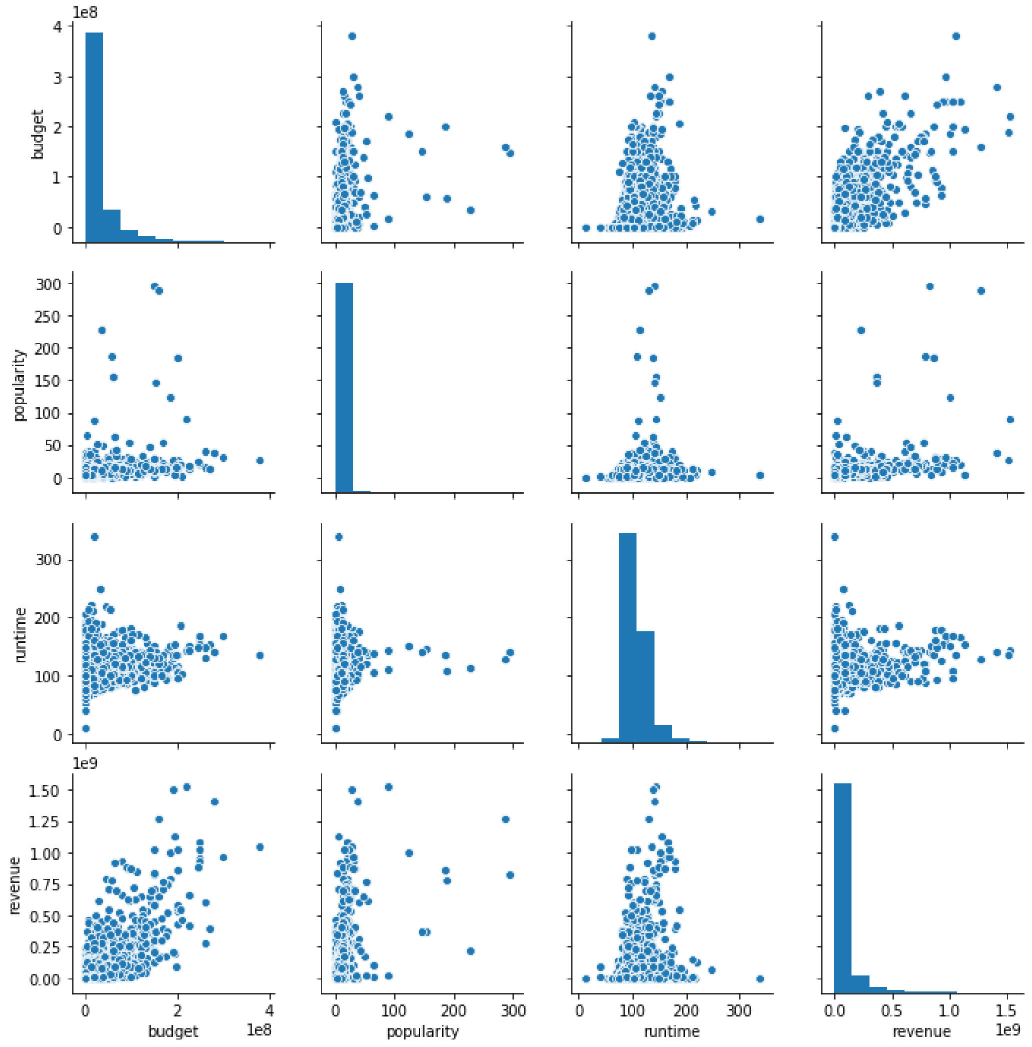
Some outliers, particulary for budget and revenue numbers, do exist within the data. These are represented as decimal values rather than integers, and are typically of values smaller than 100. Review of other data sources shows that in these cases the numbers represent millions of dollars, so in these cases the numbers are corrected.

There are also movies within the dataset that are missing revenue, budget, and runtime data. These values have been acquired from other public sources and will be manually corrected within the data at runtime.

Exploratory Visualization

Without pre-processing the data, there is very little data that can be visually analyzed. A quick pairplot on the following will let us see if there might be any strength between:

- budget
- popularity
- runtime
- revenue



It does appear that there may be a linear correlation between the budget and revenue. This intuitively seems reasonable, as a film with a higher budget might generally be thought of as having a higher quality. It also appears that there may be a correlation between the runtime and revenue. This takes more the shape of a bell curve however. Films around two hours appear to generally do best in terms of revenue, falling off on either side of that. Popularity and revenue seem to have very little interaction however.

Algorithms and Techniques

I will be using two different types of regression models for this analysis. The first will be a Linear Regression and will be used to obtain a benchmark. The second is a Decision Tree and will be used for more complex modeling. In both cases a shuffle split will be used for training purposes.

Linear Regression

This is a linear approach to modeling relationships between the dependent variable, in this case revenue, and one or more independent variables. These relationship are modeled using linear predictor function where the model parameters are estimated from the data provided. The model focuses on conditional probability distribution of responses given the data provided. This is useful for prediction with a smaller set of numeric data, which is what we have in the unprocessed dataset.

Decision Tree

Decision tree models build regressions in the form of a tree structure. The dataset is broken into incrementally smaller subsets, which informs the incremental development of the questions that make up the tree. The tree itself is then comprised of *decision nodes* and *leaf nodes*. Each decision node will have two or more branches, each of which will lead to either another decision node or a leaf node. The leaf nodes represent final determinations made by the model based on the data provided.

Shuffle Split

Since training requires that true outcomes be available in advance, it is important to split the data into training and testing sets and to separate the known outcomes from the predictive data. This does two things:

1. It prevents the model from memorizing all the data and not generalizing the model well. This is known as over-fitting.
2. The models should not have access to the known outcomes while training. It sounds obvious, but is important to keep in mind. The model will make predictions while it is learning and then compare against the known outcomes. This will further inform the models learning, depending on the model used.

Benchmark

There is no default benchmark value provided for this particular exercise. So creating a benchmark is required to allow comparison as a more complex model is built out.

Using the predetermined linear regression and a custom scorer for Root Mean Square Log Error a model was generated for the unprocessed data. Since this is the scoring mechanism used for the Kaggle competition it is used here as well. As noted earlier, Root Mean Square Log Error prevents large differences between large numbers from being penalized different from smaller differences between smaller numbers. This is important int the context of this data as some films have worldwide revenue of close to $1 billion US.

The best estimator was generated and run against the test data split from the training set. A look at the first 20 results shows that the model predicted several films to have revenues significantly in the negative amounts. The final score from the benchmark modeling was **1.6401099366269307**.

III. Methodology

(approx. 3-5 pages)

Data Preprocessing

As noted earlier, there are several attributes that are arrays of objects stored as a JSON style string. All of these will need to be pre-processed. Effectively the ids for each will be one-hot encoded.

- genres: Will be encoded into new attributes labeled as *genres_X* where *X* is the id of the genre
- production_companies: Will be encoded into new attributes labeled as *pcomp_X* where *X* is the id of the company
- production_countries: Will be encoded into new attributes labeled as *pcoun_X* where *X* is the id of the country
- spoken_languages: Will be encoded into new attributes labeled as *spoken_X* where *X* is the id of the language
- Keywords: Will be encoded into new attributes labeled as *key_X* where *X* is the id of the keyword
- cast: Will be encoded into new attributes labeled as *cast_X* where *X* is the id of the actor / actress
- crew: Will be encoded into multiple new attributes:
    - Director will be encoded as *director_X* where *X* is the id of the director
    - Executive Producer will be encoded as *execprod_X* where *X* is the id of the executive producer

Other attributes will be encoded into booleans, based on whether there is any data on that attribute provided for a movie:

- belongs_to_collection
- homepage

The date attribute will be encoded into two new attributes, *year* and *week*.

As these transformations are completed the source attributes will be dropped from the dataset. Other attributes to be dropped from the outset are:

- imdb_id
- poster_path
- original_title

- overview
- status
- tagline
- title

Some of these attributes could be one-hot encoded but are anticipated to be noise given their correlation to marketing activities rather than the actual production of the films.

There are also films with missing data points. These are manually inserted into the data to make sure the predictive data is complete.

Once all of this preprocessing is complete, it results in a dataset that has a shape of (3000, 25270). This is a very wide dataset!

Implementation

Multiple function needed to be created to pre-process the data for analysis. The most complicated were related to the columns that used strings to represent arrays of object data. Originally the intent was to use a method described at https://datascience.stackexchange.com/questions/14847/multiple-categorical-values-for-a-single-feature-how-to-convert-them-to-binary-u. However, this was quickly found to not work with the particular representation of this data and had to be discarded.

A custom function, in the json_columnizer.py file and named *jcolumnize* was written to handle the encoding of most of these types of attributes. Parameters for this function were the dataframe, the column (attribute) to be encoded, which property of the object to use as the encoded value, the label prefix for encoding, and a castlimit value. This last parameter was used for the cast attribute and allowed for controlling how many cast members to encode from each film. The following steps were then required in the function:

- Select all rows where the identified column might have a null value and set them to the string "[]", representing an empty array
- Perform a literal evaluation on every row in that column to turn the string into an actual array of object data
- Create a copy of of the selected column and iterate over it to create a list of all the objects in all the arrays
- Dedup the items in the new list
- Create new columns in the dataframe for each item in the list, using the prefix and property parameter values, setting their values to 0 for all rows by default
- Iterate over all the rows in the dataframe and encode the values into the appropriate attributes

Encoding the crew and original language values were then separate functions that were a variation on this theme. Particularly for crew, the ability to specify which crew position was desired needed to be provided.

Encoding values that were to become booleans was far simpler. For all rows in the specified column where the value was not null the value was transformed to 1. Then all rows in the specified column where the value was null the value was set to 0.

Transforming the date required creation of a columnizeDates function with a strToDate helper function.

- Receive the dataframe and specified column as parameters
- Call the helper function to convert date strings in the specified column into proper date values
  - Since these were two-digit dates, assume any number < 30 should be converted to 20xx and number > 30 should convert to 19xx
- Create a year attribute and set all values to 0 by default
- Use the timtuple() of the date to set the year for each row
- Create a week attribute and set all values to 0 by default
- Use the isocalendar() of the date to set the week for each row

Some additional functions were also written to automate upating the missing attributes for some rows in the dataset.

Even with all of that done, there were still some value errors that occurred. So a final quality check function was put in place. This takes an array of column labels to make sure all the original source columns have been purged. Then all data is checked to make sure no un-transformed strings, lists, dicts, or tuples have been left behind. This step assured that the data used for modeling would be clean and of a format that could be processed by the regression algorithms.

Once the data was ready, training and testing sets were created using a train_test_split. The split size was 20% for testing, 80% for training. A Root Mean Square Log Error function was added (called *rmsle* in the code) for use as a scoring function. A function for fitting the model (*fit_model* in the code) was also created to do the following:

- Create cross-validation sets
- Initialize a Decision Tree regressor
- Set parameters (max_depth with a range of 1-20 for starters based on the number of columns in the dataset)
- Create the scorer with *make_scorer*
- Create a GridSearchCV to find the best model fit among the parameters
- Fit the grid and return the best estimator

At this point the model was generated, with a best max_depth of 19. From here predictions against the test set were retrieved and the *rmsle* was used to score the predictions against the real outcomes. The final score was **2.383488886396367**, considerably worse than the baseline achieved with linear regression.

Refinement

In this section, you will need to discuss the process of improvement you made upon the algorithms and techniques you used in your implementation. For example, adjusting parameters for certain models to acquire improved solutions would fall under the refinement category. Your initial and final solutions should be reported, as well as any significant intermediate results as necessary. Questions to ask yourself when writing this section:

- *Has an initial solution been found and clearly reported?*
- *Is the process of improvement clearly documented, such as what techniques were used?*
- *Are intermediate and final solutions clearly reported as the process is improved?*

The model was implemented with a GridSearchCV to allow for controlling multiple parameters and testing their various combinations. For an initial look at results I check only max_depth on the decision tree regressor, with all other parameters left as default. The max_depth parameter seemed the best place to start as it controls the number of decision points, or questions asked, by the model. The first max_depth tested was for a range of 1..4, which yielded a result of 3.023. Successive tests were made with other ranges of 1..n.

max_depth	score
4	3.0323020722497644
10	2.383488886396367
20	2.377367680621276
40	2.3225570929010173

Diving further into max_depth ended at an upper limit of 40 as that test showed the optimal max_depth was 29. My expectation had been that this number would wind up much higher given there being over 25,000 columns in the dataset. At this point I began adding other parametric combinations, all based on max_depth, min_samples_split, and max_features. Using min_samples_split seemed a good addition as it controlled the minimum number of samples required create a branch in the decision tree. And max_features also seemed good to use since it controlled how many feature would be considered in the formation of a branch. Again, given the number of columns involved, parameters that would break the dataset into more generalized (and smaller) chunks for decision branching seemed a good strategy.

Test 1 The following parameters:

- max_depth: [30, 40, 50, 60, 80, 100]
- min_samples_split: [2, 4, 8, 16, 32, 64, 128]

- max_features: [10, 20, 40, 80, 100, 200, 400, 800, 1000]

Yielded results of:

- max_depth = 100
- min_samples_split = 32
- max_features = 1000
- score = **2.3262091514997207**

Test 2 The following parameters:

- max_depth: [60, 80, 100, 200, 400, 800, 1000]
- min_samples_split: [2, 4, 8, 16, 32, 64, 128]
- max_features: [100, 200, 400, 800, 1000, 2000, 4000, 8000]

Yielded results of:

- max_depth = 800
- min_samples_split = 64
- max_features = 4000
- score = **2.162784944429385**

Test 3 The following parameters:

- max_depth: [800, 850, 900, 950, 1000]
- min_samples_split: [64, 80, 96, 112, 128]
- max_features: [4000, 5000, 6000, 7000, 8000]

Yielded results of:

- max_depth = 850
- min_samples_split = 64
- max_features = 7000
- score = **2.153853569439121**

Test 4 The following parameters:

- max_depth: [850, 860, 870, 880, 890, 900]
- min_samples_split: [32, 64, 70, 75, 80]
- max_features: [7000, 7250, 7500, 7750, 8000]

Yielded results of:

- max_depth = 880
- min_samples_split = 64
- max_features = 7750
- score = **2.153772540838531**

Test 5 The following parameters:

- max_depth: [880, 885, 890, 895, 900, 905, 910]
- min_samples_split: [32, 64, 70, 102, 128]
- max_features: [7500, 7750, 8000, 8250, 8500, 8750, 9000]

Yielded results of:

- max_depth = 900
- min_samples_split = 70

- max_features = 8750
- score = **2.1607568160810113**

The min_samples_split had quickly settled in at 64, while max_depth and max_features continued to drive incremental improvements as they were allowed to grow until Test 5.

IV. Results

Model Evaluation and Validation

Refinements stopped at Test 5. While additional refinement may be possible, that the score increased slightly at this point is an indication that the sensitivity had reached a point that changes in the training data were beginning to reflect in the score. This was illustrated by the score ticking up slightly. The parameters seem appropriate for the number of columns in the dataset.

Each refinement run of the model used a randomized testing set, so another full run of Test 5 was performed and resulted in *max_features* being **7500** with a *score* of **2.206622088390178**, further validating the perturbations caused by the training data.

Generally speaking, it does not seem that the results from this model can be trusted. While the parameters and model seem reasonable on the surface, the results are not what should be expected. A score of 2.16 shows a fairly large degree of variability. The best scores among the Kaggle competitors are producing much better results.

Justification

Recalling the benchmark test used at the beginning, a Linear Regression of the dataset produced a final score of **1.6401099366269307**. Using a Decision Tree with a number of rounds of tuning, the final model produced a score of **2.1607568160810113**. The results of the Decision Tree are clearly weaker than the Linear Regression.

To further investigate, a Decision Tree regression was run against the same basic data used for the Linear Regression model. The following grid search parameters were used:

- max_depth: [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
- min_samples_split: [2, 3, 4]
- max_features: [1, 2, 3, 4]

The results were:

- max_depth: 10
- min_samples_split: 3
- max_features: 2

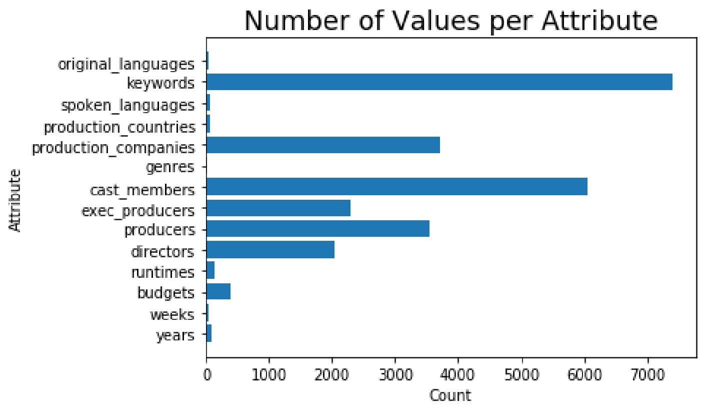
The final score of this test was **2.4901160573631285**

Clearly the Decision Tree regressor produces inferior results to a basic linear regression, whether using the same data or an expanded set of data. With the same data this makes a certain amount of sense, given that only budget, popularity, and runtime are being considered. As decision points these are fairly weak whereas the Linear Regression readily produces a well-defined prediction line between budget and revenue, as illustrated earlier.

While a wider dataset produces more accurate results for the Decision Tree due to richer decision points (cast, director, genre, etc) it is still not delivering an acceptable level of accuracy for predictions.

V. Conclusion

Free-Form Visualization



This visualization illustrates the number of unique columns / values associated with primary attributes of the movie data. For cast members it must be kept in mind that this modeling only considered the top four billed cast members of each film. So the true number for this attribute is certainly much higher. Other attributes, such as description, were not considered in the modeling at all and would have created many additional values.

Each of these values represents a possible desicion point for the Decision Tree. All this data could either be rich in terms of decision value, or could actually be a source of noise and adding confusion to the model. Each seems like they should be important, but considering that several of these items have more values than the actual number of movies in the dataset it could be problematic.

Reflection

This project involved:

- Gathering the necessary data for analysis
- Reviewing the data for cleanliness
- Planning how to clean bad / incorrect data
- Planning how to pre-process data for best representation in modeling
- Writing all the functions to manage pre-processing in a reusable fashion
- Executing the pre-processing
- Identifying a baseline model to use as a target for improvement
- Developing a target model for learning
- Refining the parameters of the model, and doing so in a way that would be optimized for performance
- Reviewing model results and refining further

Writing all the function for pre-processing was by far the most difficult aspect of the project, at least at first. There were some complex transmutations of the data that needed to take place on several of the columns before modeling could be done. For someone more adept with Python this may have been a much more trivial exercise, but that was not the case for me.

The modeling itself required some patience and experimentation. There was a point where I realized that the parameters I was testing in a Grid Search would actually take weeks to run, and that I would need to make more targeted adjustments to refine parameters. This was much more informative about the direction those adjustments should take and accelerated the process significantly, with the tradeoff being that it required a more attentive eye.

The hardest part in the end though was acknowleding that the Decision Tree model approach I had set out to use as the solution was poor. While there are adjustments that could be made, the Decision Tree likely just isn't the right approach to take.

Improvement

One possible solution to gain better results from this model centers on the insights derived from the graph shown in the free-form visualization above. While the decision points provided by adding data like directors, actors, keywords and the like is rich, it is also extremely noisy. There are many directors, actors, production companies and so forth that are only ever associated with a single film. Or perhaps only a single film in a given genre. Or any one of a number of other combinations.

Cleaning up some of this noise could be achieved by reducing the number of items being considered. Only directors that have been part of more than X number of movies for instance. The same with actors, production companies, etc. Then putting all others under an "other" category. Those individuals that are part of more than one film are generally so because they are successful, and so would provide better decision points for predictions. Those who are less so being grouped as "other" might represent lower success rates.

Another improvement that could be considered is adjusting budgets and revenues for inflation based on release dates. Given that we are making predictions of films against each other, even where the same cast, director, genre and series may be involved (i.e. Star Wars) the difference in years produced make a significant difference in terms of revenue based on difference in ticket prices. A movie ticket costing $3-$4 in the mid 1970s has an average of $10 today. Flattening this data, then modeling predictions and calculating inflation-adjusted results by year would certainly reduce an aspect of noise.

Appendix

Sample Data

The following is the data record for one film in the dataset. If you like *Hot Tub Time Machine* then you'll like this sample data. :)

1 [{'id': 313576, 'name': 'Hot Tub Time Machine Collection', 'poster_path': '/iEhb00TGPucF0b4joM1ieyY026U.jpg', 'backdrop_path': '/noeTVcgpBiD48fDjFVic1Vz7ope.jpg'}] 14000000 [{'id': 35, 'name': 'Comedy'}] tt2637294 en Hot Tub Time Machine 2 When Lou, who has become the "father of the Internet," is shot by an unknown assailant, Jacob and Nick fire up the time machine again to save their friend. 6.575393 /tQtWuwvMf0hCc2QR2tkolwl7c3c.jpg [{'name': 'Paramount Pictures', 'id': 4}, {'name': 'United Artists', 'id': 60}, {'name': 'Metro-Goldwyn-Mayer (MGM)', 'id': 8411}] [{'iso_3166_1': 'US', 'name': 'United States of America'}] 2/20/15 93 [{'iso_639_1': 'en', 'name': 'English'}] Released The Laws of Space and Time are About to be Violated. Hot Tub Time Machine 2 [{'id': 4379, 'name': 'time travel'}, {'id': 9663, 'name': 'sequel'}, {'id': 11830, 'name': 'hot tub'}, {'id': 179431, 'name': 'duringcreditsstinger'}] [{'cast_id': 4, 'character': 'Lou', 'credit_id': '52fe4ee7c3a36847f82afae7', 'gender': 2, 'id': 52997, 'name': 'Rob Corddry', 'order': 0, 'profile_path': '/k2zJL0V1nEZuFT08xUdOd3ucfXz.jpg'}, {'cast_id': 5, 'character': 'Nick', 'credit_id': '52fe4ee7c3a36847f82afaeb', 'gender': 2, 'id': 64342, 'name': 'Craig Robinson', 'order': 1, 'profile_path': '/tVaRMkJXOEVhYxtnnFuhqW0Rjzz.jpg'}, {'cast_id': 6, 'character': 'Jacob', 'credit_id': '52fe4ee7c3a36847f82afaef', 'gender': 2, 'id': 54729, 'name': 'Clark Duke', 'order': 2, 'profile_path': '/oNzK0umwm5Wn0wyEbOy6TVJCSBn.jpg'}, {'cast_id': 7, 'character': 'Adam Jr.', 'credit_id': '52fe4ee7c3a36847f82afaf3', 'gender': 2, 'id': 36801, 'name': 'Adam Scott', 'order': 3, 'profile_path': '/5gb65xz8bzd42yjMAl4zwo4cvKw.jpg'}, {'cast_id': 8, 'character': 'Hot Tub Repairman', 'credit_id': '52fe4ee7c3a36847f82afaf7', 'gender': 2, 'id': 54812, 'name': 'Chevy Chase', 'order': 4, 'profile_path': '/svjpyYtPwtjvRxX9IZnOmOkhDOt.jpg'}, {'cast_id': 9, 'character': 'Jill', 'credit_id': '52fe4ee7c3a36847f82afafb', 'gender': 1, 'id': 94098, 'name': 'Gillian Jacobs', 'order': 5, 'profile_path': '/rBnhe5vhNPnhRUdtYahBWx90fJM.jpg'}, {'cast_id': 10, 'character': 'Sophie', 'credit_id': '52fe4ee7c3a36847f82afaff', 'gender': 1, 'id': 1159009, 'name': 'Bianca Haase', 'order': 6, 'profile_path': '/4x3nbtD8q8phAJPmoGWXPvz0iM.jpg'}, {'cast_id': 11, 'character': 'Kelly', 'credit_id': '5524ec51c3a3687df3000dbb', 'gender': 1, 'id': 86624, 'name': 'Collette Wolfe', 'order': 7, 'profile_path': '/aSD4h5379b2eEw3bLou9ByLimmq.jpg'}, {'cast_id': 13, 'character': 'Brad', 'credit_id': '5524ec8ec3a3687ded000d72', 'gender': 2, 'id': 466505, 'name': 'Kumail Nanjiani', 'order': 9, 'profile_path': '/x4nAztHY72SVciRfxEsbhIVTsIu.jpg'}, {'cast_id': 14, 'character': 'Courtney', 'credit_id': '5524ec9bc3a3687df8000d13', 'gender': 1, 'id': 70776, 'name': 'Kellee Stewart', 'order': 10, 'profile_path': '/w3xmsEPmJc1Cf0dQ4aIn8YmlHbk.jpg'}, {'cast_id': 15, 'character': 'Terry', 'credit_id': '5524eca892514171cb008237', 'gender': 2, 'id': 347335, 'name': 'Josh Heald', 'order': 11, 'profile_path': '/pwXJIenrDMrG7t3zNfLvr8w1RGU.jpg'}, {'cast_id': 16, 'character': 'Susan', 'credit_id': '5524ecb7925141720c001116', 'gender': 0, 'id': 1451392, 'name': 'Gretchen Koerner', 'order': 12, 'profile_path': '/muULPexCTJGyJba4yKzxronpD50.jpg'}, {'cast_id': 17, 'character': 'Herself', 'credit_id': '5524ecc3c3a3687ded000d74', 'gender': 1, 'id': 98879, 'name': 'Lisa Loeb', 'order': 13, 'profile_path': '/bGqg58ca0bZR38z9HliUMmeNGE.jpg'}, {'cast_id': 18, 'character': 'Herself', 'credit_id': '5524ecd3c3a3687e11000ed3', 'gender': 1, 'id': 1394648, 'name': 'Jessica Williams', 'order': 14, 'profile_path': '/A4syKjkcYB92wLEhH0c0hC3BCpz.jpg'}, {'cast_id': 19, 'character': 'Himself', 'credit_id': '5524ece6925141718d001009', 'gender': 0, 'id': 1451393, 'name': 'Bruce Buffer', 'order': 15, 'profile_path': None}, {'cast_id': 20, 'character': 'Shot Girl', 'credit_id': '5524ecf5c3a3687e08000dc2', 'gender': 0, 'id': 1451394, 'name': 'Mariana Paola Vicente', 'order': 16, 'profile_path': '/ckPllza8624UHWGHCbLShkLxCD1.jpg'}, {'cast_id': 33, 'character': 'Choozy Doozy Host', 'credit_id': '555844da9251412afe0013a9', 'gender': 2, 'id': 2224, 'name': 'Christian Slater', 'order': 17, 'profile_path': '/3ElLWjnvchMS6Q4cIQOK8QNAoMG.jpg'}, {'cast_id': 35, 'character': 'Gary Winkle', 'credit_id': '55872027c3a3683853005074', 'gender': 0, 'id': 185805, 'name': 'Jason Jones', 'order': 18, 'profile_path': '/aIoCw6vo8AGMdsQRAI5g2t0yJT3.jpg'}, {'cast_id': 36, 'character': 'Bridesmaid', 'credit_id': '55efe971c3a368090c00cd1b', 'gender': 0, 'id': 1507448, 'name': 'Olivia Jordan', 'order': 19, 'profile_path': '/szMukAEiIDeasel0lvyaeyKuych.jpg'}, {'cast_id': 37, 'character': 'Christine', 'credit_id': '55efe980c3a36871bf008176', 'gender': 1, 'id': 1334091, 'name': 'Christine Bently', 'order': 20, 'profile_path': '/oUZltnGa55OXE52hfyPTfCshuNy.jpg'}, {'cast_id': 38, 'character': 'Excited Girl', 'credit_id': '55efe98e9251413e3201d316', 'gender': 0, 'id': 557803, 'name': 'Stacey Asaro', 'order': 21, 'profile_path': '/qTPdlr1dXf3kNdyHuDsgtGC0HCC.jpg'}, {'cast_id': 64, 'character': 'Adam (uncredited)', 'credit_id': '58f2135ac3a3682e95008b91', 'gender': 2, 'id': 3036, 'name': 'John Cusack', 'order': 22, 'profile_path': '/uKydQYuZ9TnCzvbQLtj6j98vWAT.jpg'}, {'cast_id': 65, 'character': 'J-Bird', 'credit_id': '59ac0240c3a3682cc802c399', 'gender': 2, 'id': 59256, 'name': 'Adam Herschman', 'order': 23, 'profile_path': '/wZMwiuX1DslF6hDS50z9OTN6z1X.jpg'}, {'cast_id': 66, 'character': 'Bridesmaid', 'credit_id': '59ac02cd925141079d02b1b4', 'gender': 1, 'id': 129714, 'name': 'Kisha Sierra', 'order': 24, 'profile_path': None}] [{'credit_id': '59ac067c92514107af02c8c8', 'department': 'Directing', 'gender': 0, 'id': 1449071, 'job': 'First Assistant Director', 'name': 'Kelly Cantley', 'profile_path': None}, {'credit_id': '52fe4ee7c3a36847f82afad7', 'department': 'Directing', 'gender': 2, 'id': 3227, 'job': 'Director', 'name': 'Steve Pink', 'profile_path': '/myHOgo8mQSCiCAZNGMRdHVr03jr.jpg'}, {'credit_id': '5524ed25c3a3687ded000d88', 'department': 'Writing', 'gender': 2, 'id': 347335, 'job': 'Writer', 'name': 'Josh Heald', 'profile_path': '/pwXJIenrDMrG7t3zNfLvr8w1RGU.jpg'}, {'credit_id': '5524ed2d925141720c001128', 'department': 'Writing', 'gender': 2, 'id': 347335, 'job': 'Characters', 'name': 'Josh Heald', 'profile_path': '/pwXJIenrDMrG7t3zNfLvr8w1RGU.jpg'}, {'credit_id': '5524ed3d92514166c1004a5d', 'department': 'Production', 'gender': 2, 'id': 57822, 'job': 'Producer', 'name': 'Andrew Panay', 'profile_path': None}, {'credit_id':

'5524ed4bc3a3687df3000dd2', 'department': 'Production', 'gender': 0, 'id': 1451395, 'job': 'Associate Producer', 'name': 'Adam Blum', 'profile_path': None}, {'credit_id': '5524ed5a925141720c00112c', 'department': 'Production', 'gender': 2, 'id': 52997, 'job': 'Executive Producer', 'name': 'Rob Corddry', 'profile_path': '/k2zJL0V1nEZuFT08xUdOd3ucfXz.jpg'}, {'credit_id': '5524ed85c3a3687e0e000f56', 'department': 'Production', 'gender': 0, 'id': 62807, 'job': 'Executive Producer', 'name': 'Ben Ormand', 'profile_path': None}, {'credit_id': '5524ed9fc3a3687e0e000f59', 'department': 'Sound', 'gender': 2, 'id': 23486, 'job': 'Original Music Composer', 'name': 'Christophe Beck', 'profile_path': '/2fnJUmCk6IEpVIptpYaUk31epHx.jpg'}, {'credit_id': '5524eda6c3a3687e03000d28', 'department': 'Camera', 'gender': 2, 'id': 6117, 'job': 'Director of Photography', 'name': 'Declan Quinn', 'profile_path': None}, {'credit_id': '5524edb4925141720c00113d', 'department': 'Editing', 'gender': 0, 'id': 1451396, 'job': 'Editor', 'name': 'Jamie Gross', 'profile_path': None}, {'credit_id': '5524edc1925141727600102e', 'department': 'Production', 'gender': 0, 'id': 22219, 'job': 'Casting', 'name': 'Susie Farris', 'profile_path': None}, {'credit_id': '5524edd192514171cb008257', 'department': 'Art', 'gender': 0, 'id': 1002643, 'job': 'Production Design', 'name': 'Ryan Berg', 'profile_path': None}, {'credit_id': '555ad9be9251411e5b00d485', 'department': 'Production', 'gender': 2, 'id': 57431, 'job': 'Executive Producer', 'name': 'Matt Moore', 'profile_path': None}, {'credit_id': '5677e93bc3a36816890087dc', 'department': 'Directing', 'gender': 0, 'id': 1551818, 'job': 'Script Supervisor', 'name': 'Nicole Garcea', 'profile_path': None}, {'credit_id': '5677e96a92514179e10093d0', 'department': 'Production', 'gender': 0, 'id': 1551819, 'job': 'Production Coordinator', 'name': 'Jason Salzman', 'profile_path': None}, {'credit_id': '5677e98492514179d2008cd9', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1422996, 'job': 'Costume Design', 'name': 'Carol Cutshall', 'profile_path': None}, {'credit_id': '5677e9d5c3a368168e009414', 'department': 'Art', 'gender': 2, 'id': 500199, 'job': 'Set Decoration', 'name': 'Tim Cohn', 'profile_path': None}, {'credit_id': '5677f89d9251417845001a61', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1527917, 'job': 'Hair Department Head', 'name': 'Voni Hinkle', 'profile_path': None}, {'credit_id': '5677f8b392514179dd0089fb', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1431554, 'job': 'Makeup Department Head', 'name': 'Remi Savva', 'profile_path': None}, {'credit_id': '5677f8d1c3a3681689008a4b', 'department': 'Art', 'gender': 0, 'id': 66495, 'job': 'Art Direction', 'name': 'Jason Baldwin Stewart', 'profile_path': None}, {'credit_id': '5677f8eec3a3681685008dd5', 'department': 'Production', 'gender': 0, 'id': 1412466, 'job': 'Production Supervisor', 'name': 'Korey Budd', 'profile_path': None}, {'credit_id': '5677f90a9251417845001a7d', 'department': 'Sound', 'gender': 0, 'id': 1401562, 'job': 'Sound Re-Recording Mixer', 'name': 'Gary C. Bourgeois', 'profile_path': None}, {'credit_id': '5677f91e9251417845001a84', 'department': 'Sound', 'gender': 0, 'id': 1396794, 'job': 'Sound Re-Recording Mixer', 'name': 'Gabriel J. Serrano', 'profile_path': None}, {'credit_id': '5677f938c3a3681680008dd4', 'department': 'Editing', 'gender': 0, 'id': 13168, 'job': 'Dialogue Editor', 'name': 'Victoria Rose Sampson', 'profile_path': None}, {'credit_id': '5677f94e92514179dd008a1f', 'department': 'Sound', 'gender': 0, 'id': 1551839, 'job': 'Production Sound Mixer', 'name': 'Michael B. Koff', 'profile_path': None}, {'credit_id': '5677f968c3a368168e009698', 'department': 'Sound', 'gender': 0, 'id': 113052, 'job': 'Sound Effects Editor', 'name': 'Randall Guth', 'profile_path': None}, {'credit_id': '5677f98dc3a3681685008e02', 'department': 'Crew', 'gender': 2, 'id': 1442535, 'job': 'Stunt Coordinator', 'name': 'Chuck Picerni Jr.', 'profile_path': '/yE5QtXUzcrnCzMRctZL8F5g842B.jpg'}, {'credit_id': '5677f9a692514179dd008a49', 'department': 'Camera', 'gender': 0, 'id': 1437305, 'job': 'Camera Operator', 'name': 'Michael Applebaum', 'profile_path': None}, {'credit_id': '5677f9bd9251417845001aae', 'department': 'Camera', 'gender': 0, 'id': 1401765, 'job': 'Still Photographer', 'name': 'Steve Dietl', 'profile_path': None}, {'credit_id': '5677f9e592514179e7008bf7', 'department': 'Lighting', 'gender': 0, 'id': 1402721, 'job': 'Rigging Gaffer', 'name': 'Tarik Naim Alherimi', 'profile_path': None}, {'credit_id': '5677f9f4c3a368167c0090ed', 'department': 'Lighting', 'gender': 0, 'id': 1402719, 'job': 'Gaffer', 'name': 'Paul Olinde', 'profile_path': None}, {'credit_id': '5677fa21c3a368168e0096ca', 'department': 'Sound', 'gender': 0, 'id': 1551840, 'job': 'Music Supervisor', 'name': 'Steve Griffen', 'profile_path': None}, {'credit_id': '5677fa31c3a3681680008e04', 'department': 'Sound', 'gender': 0, 'id': 1551841, 'job': 'Music Editor', 'name': 'Matt Fausak', 'profile_path': None}, {'credit_id': '5677fa4392514179dd008a76', 'department': 'Sound', 'gender': 0, 'id': 1551840, 'job': 'Music Editor', 'name': 'Steve Griffen', 'profile_path': None}, {'credit_id': '5677fa609251417845001acf', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1403416, 'job': 'Costume Supervisor', 'name': 'Shonta T. McCray', 'profile_path': None}, {'credit_id': '5677fa8492514179d2008fb3', 'department': 'Camera', 'gender': 0, 'id': 1425831, 'job': 'Steadicam Operator', 'name': 'Mark Karavite', 'profile_path': None}, {'credit_id': '5677fab2c3a3681689008ac3', 'department': 'Camera', 'gender': 0, 'id': 1551842, 'job': 'First Assistant Camera', 'name': 'Joe Waistell', 'profile_path': None}, {'credit_id': '5677faecc3a368168e0096fe', 'department': 'Sound', 'gender': 0, 'id': 58362, 'job': 'Supervising Sound Editor', 'name': 'Michael Hilkene', 'profile_path': None}, {'credit_id': '59ac0368c3a3682c0a02c484', 'department': 'Crew', 'gender': 0, 'id': 1881584, 'job': 'Additional Writing', 'name': 'John Karnay', 'profile_path': None}, {'credit_id': '59ac0411c3a3682bf0028966', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1431552, 'job': 'Hairstylist', 'name': 'Daina Daigle', 'profile_path': None}, {'credit_id': '59ac0504925141072302b8fb', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1712001, 'job': 'Makeup Artist', 'name': 'Allison Gordin', 'profile_path': None}, {'credit_id': '59ac0570c3a3682bf0028aac', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 578725, 'job': 'Makeup Artist', 'name': 'Darryl Lucas', 'profile_path': None}, {'credit_id': '59ac05a4925141077e02c97e', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1463274, 'job': 'Makeup Artist', 'name': 'Annabelle MacNeal', 'profile_path': None}, {'credit_id':

'59ac05c6925141076502d106', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1881586, 'job': 'Makeup Artist', 'name': 'Marina Savva', 'profile_path': None}, {'credit_id': '59ac0615c3a3682c480296aa', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1406267, 'job': 'Hairstylist', 'name': 'Carl G. Variste', 'profile_path': None}, {'credit_id': '59ac06ba925141076502d1fa', 'department': 'Directing', 'gender': 0, 'id': 1798593, 'job': 'First Assistant Director', 'name': 'Josh King', 'profile_path': None}, {'credit_id': '59ac06f1c3a3682c2202aca0', 'department': 'Art', 'gender': 0, 'id': 1415083, 'job': 'Greensman', 'name': 'Scott C. Bivona', 'profile_path': None}, {'credit_id': '59ac072c925141076502d260', 'department': 'Art', 'gender': 0, 'id': 1881587, 'job': 'Title Designer', 'name': 'Eunha Choi', 'profile_path': None}, {'credit_id': '59ac077c925141077e02cb62', 'department': 'Art', 'gender': 0, 'id': 1585302, 'job': 'Construction Coordinator', 'name': 'Daniel Coe', 'profile_path': None}, {'credit_id': '59ac07e0925141078a02d842', 'department': 'Art', 'gender': 0, 'id': 1495523, 'job': 'Set Designer', 'name': 'Spencer Davison', 'profile_path': None}, {'credit_id': '59ac0862925141072f02cf6f', 'department': 'Art', 'gender': 0, 'id': 1881589, 'job': 'Painter', 'name': 'Sonia L. Garcia', 'profile_path': None}, {'credit_id': '59ac08e0c3a3682bf0028e51', 'department': 'Art', 'gender': 0, 'id': 1424896, 'job': 'Art Department Coordinator', 'name': 'Caleb Guillotte', 'profile_path': None}, {'credit_id': '59ac0920c3a3682c2202af36', 'department': 'Art', 'gender': 0, 'id': 1393375, 'job': 'Leadman', 'name': "Pat A. O'Connor", 'profile_path': None}, {'credit_id': '59ac095592514107af02cc39', 'department': 'Art', 'gender': 0, 'id': 1881592, 'job': 'Set Designer', 'name': 'Brendan Turrill', 'profile_path': None}, {'credit_id': '59ac0989925141072302bdfa', 'department': 'Art', 'gender': 2, 'id': 76497, 'job': 'Property Master', 'name': 'Brook Yeaton', 'profile_path': None}, {'credit_id': '59ac0a2cc3a3682c9c02add1', 'department': 'Sound', 'gender': 0, 'id': 1881596, 'job': 'Boom Operator', 'name': 'Matthew Armstrong', 'profile_path': None}, {'credit_id': '59ac0aa8925141072f02d282', 'department': 'Visual Effects', 'gender': 2, 'id': 1558086, 'job': 'Special Effects Supervisor', 'name': 'Matt Kutcher', 'profile_path': None}, {'credit_id': '59ac0b2ac3a3682c2202b192', 'department': 'Crew', 'gender': 2, 'id': 1558087, 'job': 'Special Effects Coordinator', 'name': 'Eric Roberts', 'profile_path': None}, {'credit_id': '59ac0b7ac3a3682c2202b1fb', 'department': 'Visual Effects', 'gender': 0, 'id': 1392098, 'job': 'Visual Effects Supervisor', 'name': 'Rocco Passionino', 'profile_path': None}, {'credit_id': '59ac0bbe925141077e02d0c4', 'department': 'Visual Effects', 'gender': 0, 'id': 1558716, 'job': 'Visual Effects Coordinator', 'name': 'Joseph Payo', 'profile_path': None}, {'credit_id': '59ac0bf2c3a3682cc802cefa', 'department': 'Visual Effects', 'gender': 0, 'id': 1408784, 'job': 'Visual Effects Producer', 'name': 'Chris Roff', 'profile_path': None}, {'credit_id': '59ac0c51c3a3682c48029d99', 'department': 'Lighting', 'gender': 0, 'id': 1881600, 'job': 'Best Boy Electric', 'name': 'Ulyan Atamanyuk', 'profile_path': None}, {'credit_id': '59ac0cbac3a3682c0a02cff6', 'department': 'Camera', 'gender': 0, 'id': 1881602, 'job': 'Key Grip', 'name': 'Chris Ekstrom', 'profile_path': None}, {'credit_id': '59ac0d54925141072f02d5e6', 'department': 'Lighting', 'gender': 0, 'id': 1484984, 'job': 'Best Boy Electric', 'name': 'Brad Garris', 'profile_path': None}, {'credit_id': '59ac0db0925141078a02df86', 'department': 'Camera', 'gender': 0, 'id': 1881603, 'job': 'Dolly Grip', 'name': 'Kendell Joseph', 'profile_path': None}, {'credit_id': '59ac0e5a925141077e02d39f', 'department': 'Camera', 'gender': 0, 'id': 1549179, 'job': 'Dolly Grip', 'name': 'Spencer Wilcox', 'profile_path': None}, {'credit_id': '59ac0e9f925141079d02bee6', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1552626, 'job': 'Key Costumer', 'name': 'Sarah P. Koeppe', 'profile_path': None}, {'credit_id': '59ac0ec1c3a3682bf0029524', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1881605, 'job': 'Seamstress', 'name': 'Catherine Rodi', 'profile_path': None}, {'credit_id': '59ac0eef925141070702c7ff', 'department': 'Costume & Make-Up', 'gender': 0, 'id': 1463801, 'job': 'Seamstress', 'name': 'Giselle Spence', 'profile_path': None}, {'credit_id': '59ac0f5dc3a3682c4802a0f5', 'department': 'Production', 'gender': 0, 'id': 1400837, 'job': 'Location Manager', 'name': 'John A. Johnston', 'profile_path': None}, {'credit_id': '59ac0ff2c3a3682c4802a196', 'department': 'Crew', 'gender': 0, 'id': 1844322, 'job': 'Production Controller', 'name': 'Gail Marks', 'profile_path': None}] 12314651