Capstone 2: Milestone Report 2

Proposal

Problem Statement

While the linear regression model that was found for the first capstone project looks promising, the out-of-sample predictions should be more accurate. There are many tools and strategies available to obtain a regression model with better predictive power on unseen data. In order to deliver better predictions for the optimal hiring of new actors, a more generalizable model should be found.

Possible Clients

The types of clients that would be interested in the results of a profitable actor prediction model would be anyone involved in the hiring of actors. They would include, but not be limited to casting directors, directors, and producers in the movie industry. Agents may also want to glean some insight into the competition for acting roles, as well as assess the value of their own clients.

Dataset

The dataset used for this project will be the same one that was built for the first capstone project. Just to recall, it was originally extracted through API requests from the TMDb website. The process of extraction that was used can be seen in the various Data Acquisition notebooks and detailed explanatory PDF from the first project. The dataset includes characters, billing positions, release dates, runtimes, proprietary review scores, review counts, proprietary popularity scores, genres, casts, crews, budgets, information on whether movies belong to a collection, and revenues. For actor data, they have dates of birth, locations of birth, gender, and filmographies.

Solution

Considering the significant presence of outliers seen in the residual plots of the OLS model, a tree ensemble model will be optimized to find a superior replacement. In particular, a boosting model using XGBoost will be trained to produce better results. The hyperparameter selection will be performed using the informed search algorithm, Hyperopt. Also, data transformations will be performed, such as applying the natural log function and scaling, as well as analyzing the results of outlier extraction to reign in their effects more directly. The goal of this procedure will be to find a model that produces the most accurate predictions about the revenue potentials of unseen actors.

As the scale of target values is quite broad, a custom loss function will be used (Pseudo Huber) during the model optimization process. This loss function will strike the required balance of regularization between L1 (absolute loss) for large values and L2 (squared loss) for small values, depending on the magnitude of the prediction error. This will enable the model to make more accurate predictions on out-of-sample data over the full scale of movie revenues.

Demonstration

After consideration of the type of movie that needs to be cast, this model can be used to suggest which actors would be best to hire to provide that movie the highest chance of success. One way to use this model will be demonstrated by taking its predictions as suggestions for which actors should have been cast for movies filmed years ago, then comparing those predictions with the choices that were actually made in those years. As we can not know what might have happened with these alternate choices, the reader can have fun making objective assessments of the suggested recastings.

Deliverable

The final product will be available through a Github repository. It will contain a paper with all of the relevant analysis, detailed with proper visualizations and a compelling data story. To accompany this paper, a slide deck and the companion code that produced the results will be provided, as well.

Data Wrangling

The raw data used for the second capstone project was carried over from the first project. A detailed walk-through of the techniques used and choices made during data cleaning can be found in the Data Wrangling PDF from the first capstone project. The additional choices made, with respect to selecting the final form of the data, will be highlighted, thereby optimizing its use for making predictions using a boosted tree model.

The first difference between the datasets used in the two capstone projects pertains to the way that the individual movie data was aggregated over each actor. For the first capstone project, the monetary features, revenue and budget, had values that were the result of summations, as opposed to averages, over each actor. While having the data in this form aided in increasing the accuracy of the optimum linear regression model, it left a desire for more in terms of the usefulness of the predictions, as their interpretability was less informative. For example, the linear regression model made better predictions when using the total lifetime movie revenue of actors, instead of using the average movie revenue of each. Unfortunately, this resulted in a bias toward actors with longer careers. These actors were more likely to have larger movie revenue totals than did actors with fewer movies. To have a model that was more suited to making predictions on lesser known actors, the monetary data was modified, as described.

To ensure the usefulness of the final model to make predictions on unseen actors, it was important to simulate any actor's movie revenue history at some arbitrary point in that actor's career. This would allow the model to predict an actor's varying earning potential at different points in time. Training a model with this objective, would lead to better generalizability for making predictions about new actors, because actors could be observed as they once were early in a career. For example, a director may have needed more resources to guide the decision of whether or not to hire Harrison Ford for American Graffiti (1973) than the director who chose to hire him for Clear and Present Danger (1994). So, for the second capstone project, the average of the monetary movie features was taken, thereby selecting to have less bias toward actors with well established careers. Cell 128 of the Data Wrangler notebook shows the choices that were made for the aggregation by actor.

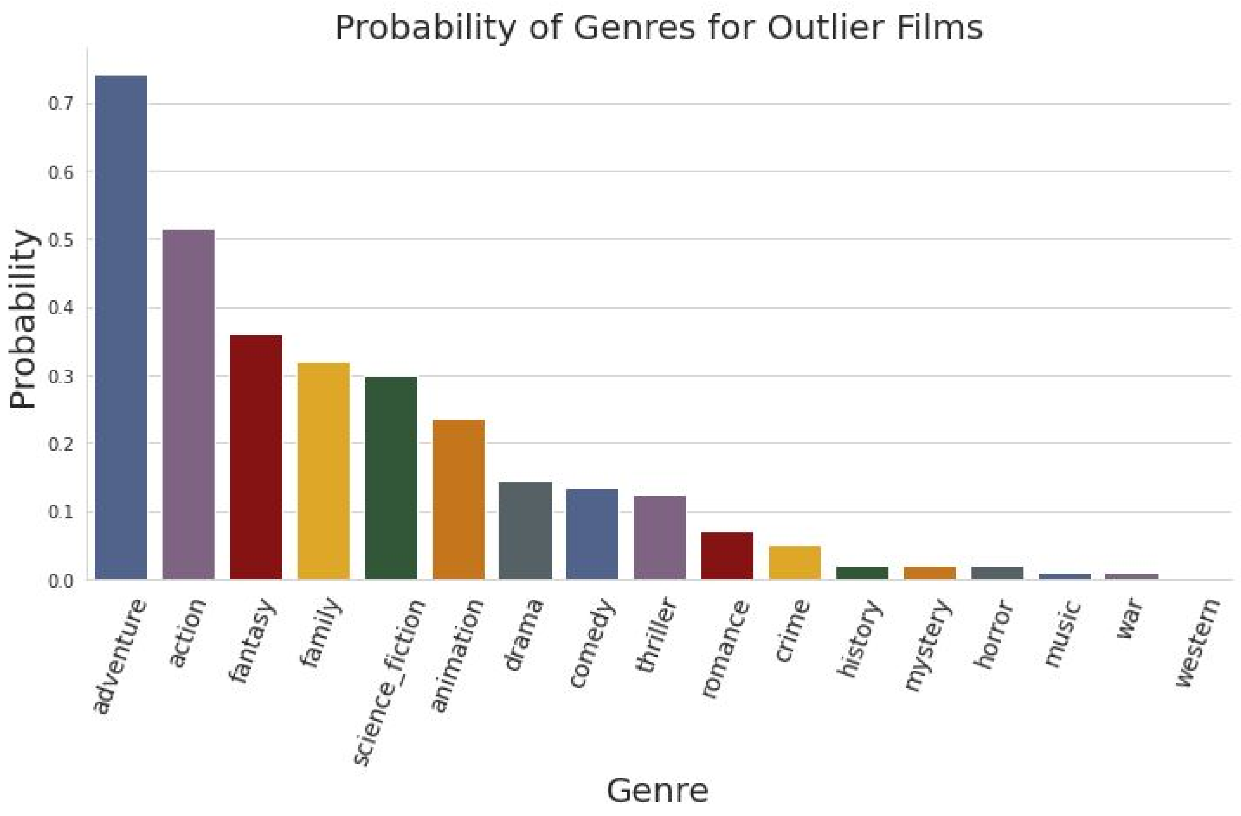
Optimize Target

The other way that the datasets were different relates to the way that the data was transformed after cleaning. In the OLS Regressor notebook of the first project, the linear regression model performance was optimized by applying a variety of data modification techniques on the features, such as various regularizations, feature selection, and outlier extraction. In the Optimize Target notebook of the second capstone project, the data modifications were restricted to the target variable, as opposed to the predictors. The selection of the optimal target values used for model training was obtained by comparing model performance, after applying the techniques of log transformation, outlier extraction, and scaling on either the revenues of individual movies (before aggregating over the actors) or the average movie revenues of each actor (after aggregation). Before transformations were made, a baseline mean absolute error (MAE) of $56,799,637 was obtained for the average 10 fold cross validation (CV) test errors, using an untuned XGBoost model.

To get the optimal model, the best strategy was to modify the movie revenues before aggregation over the actors. Removing the movies with revenues greater than three standard deviations from the mean of all movie revenues proved very useful. Performing this technique, directly on the movie revenue, reduced the error by just over 20%, with a CV test MAE of $45,296,500. Taking the natural log of these values improved that error by an additional 7% to a CV test MAE of $41,215,414 for an untuned model.

After finding the best form for the target values, an analysis was performed of the outlier movies that had been removed and how those deletions changed the data that remained. There were 97 movies that were extracted from the original dataset, which consisted of 5626 films. Most of these movies were either parts of a blockbuster series or were animation films. These movies contained 541 actors. From a count of 11,693 actors in the original dataset, 83 of them were removed, as they only appeared in these outlier films. The remaining actors had their average revenues decreased to varying degrees. Mark Hamill lost nearly 98% of his average movie revenue, after the Star Wars saga was removed. Interestingly, Harrison Ford only lost 47% of his average movie revenue. Yet, he had Raiders of the Lost Ark removed, as well. This attested to the solid career he had, outside of his big blockbuster series appearances.

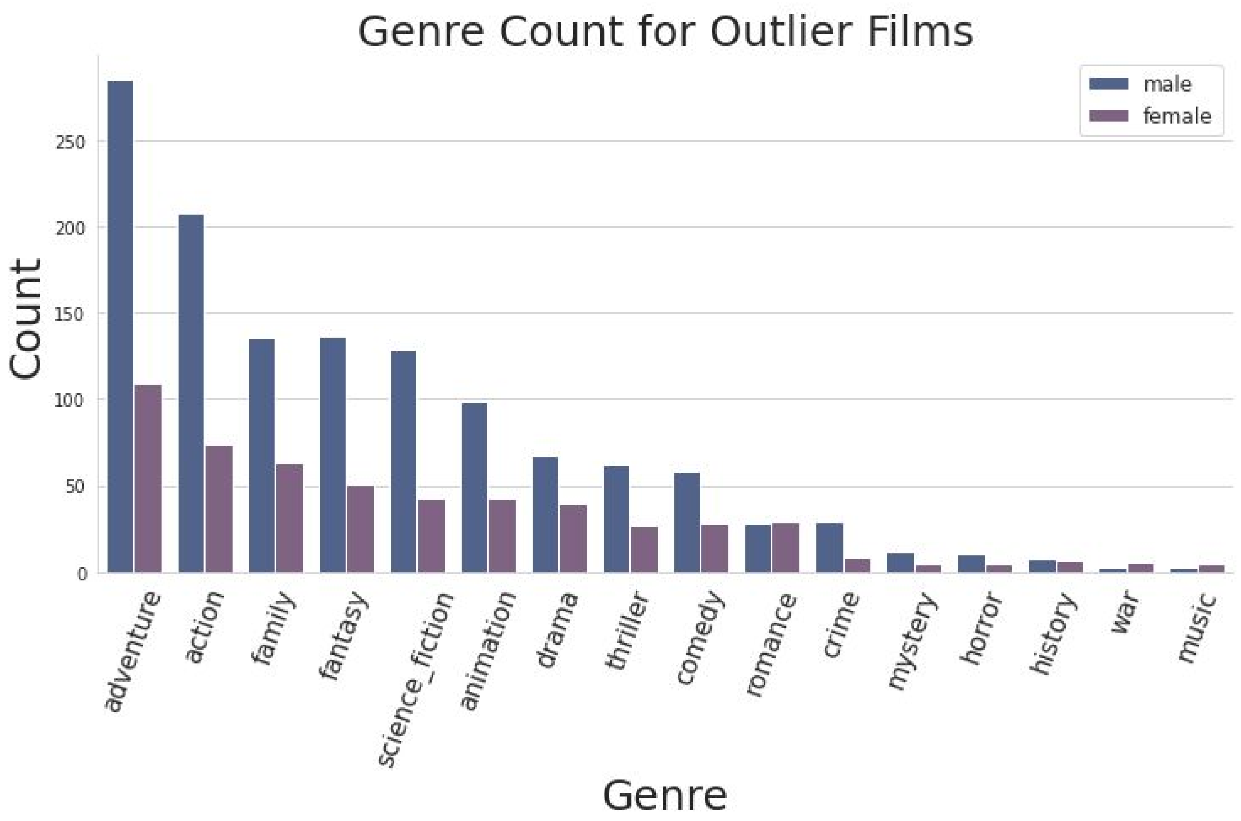
Next, the effects of removing the outlier movies shifted to how that transformation affected the distribution of the movie genres.



Adventure films comprised 74% of the outlier movies. Action was a genre description for 52% of them, as well. Fantasy, family, science fiction, and animation each made up at least 20% of the outlier movies. Many of these genres shared movies with each other. For instance, the adventure genre was included in the description of 45 out of 50 action movies, 28 out of 29 science fiction films, and 31 out of 35 fantasy releases. The family genre was named in all 23 animation films. These genre clusters blurred the lines, when defining what characteristics could be used to categorize them.

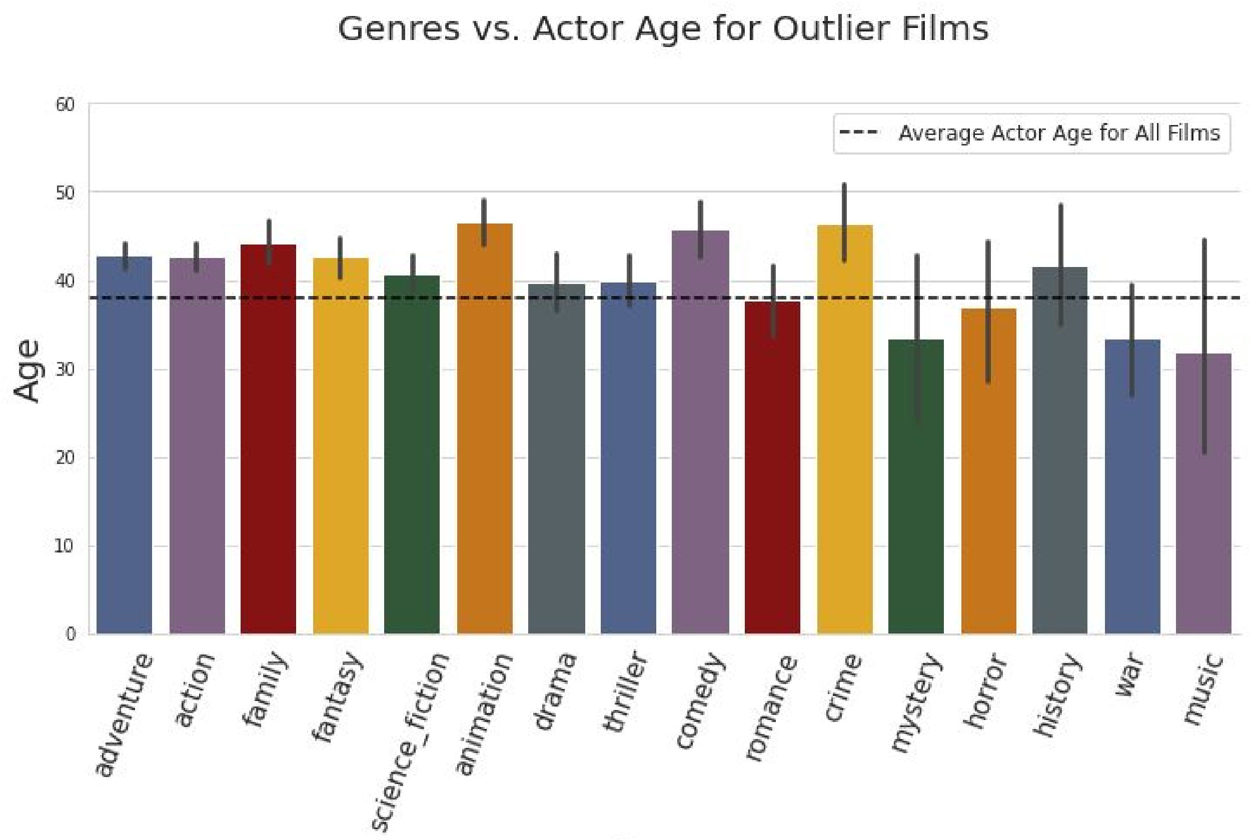
After the genres were inspected, the focus was turned to uncovering potential data biases, with respect to the socioeconomic variables. First, an analysis of the gender distribution was performed. Male actors made up almost 70% of the removed outlier data. They accounted for 61% of the full dataset. In these removed movies, the male actors had a total of 600 billion dollars in average revenue values, while the female actors had less than half that figure. Although, the percent revenue reduction, after outlier extraction, was 14% higher for the female actors. This was possible, because their count was less. In the end, the effect of many more male actors, losing a slightly smaller percent of average revenue, pulled the gender bias closer to equality.

A brief inspection of the gender distribution was performed, with respect to the genres.



The top 6 genres from the outlier movies were comprised of male actors by at least a 2:1 ratio. After these movies were taken out of the dataset, the respective gender biases in these movie categories were decreased.

The other bias that was considered was that of actor age. The actors in the outlier movies were divided into two groups, defined by the average age of all actors in the original dataset. The actors who were older than average made up 59% of those in the outlier movies. The average actor age increased from nearly 38 to just over 42 years.

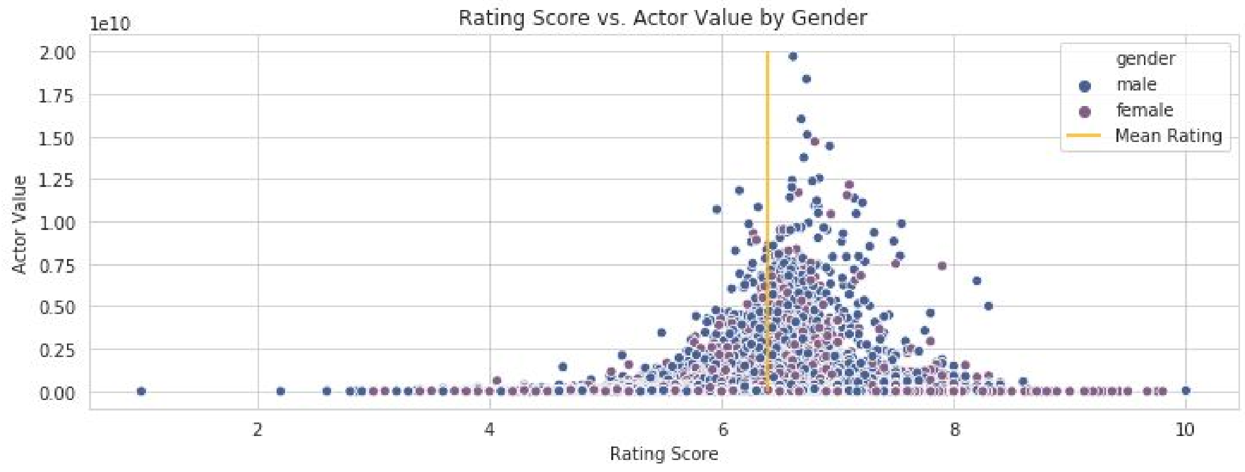


This time, the top 9 genres all had average actor ages that were above average, compared to the full dataset. Both the percent average revenue change, after outlier extraction, and the sum of the average revenues in the outlier dataset were similar for each age group. This time, the effects of the removal of outlier movies was less impactful on the bias.

Data Story

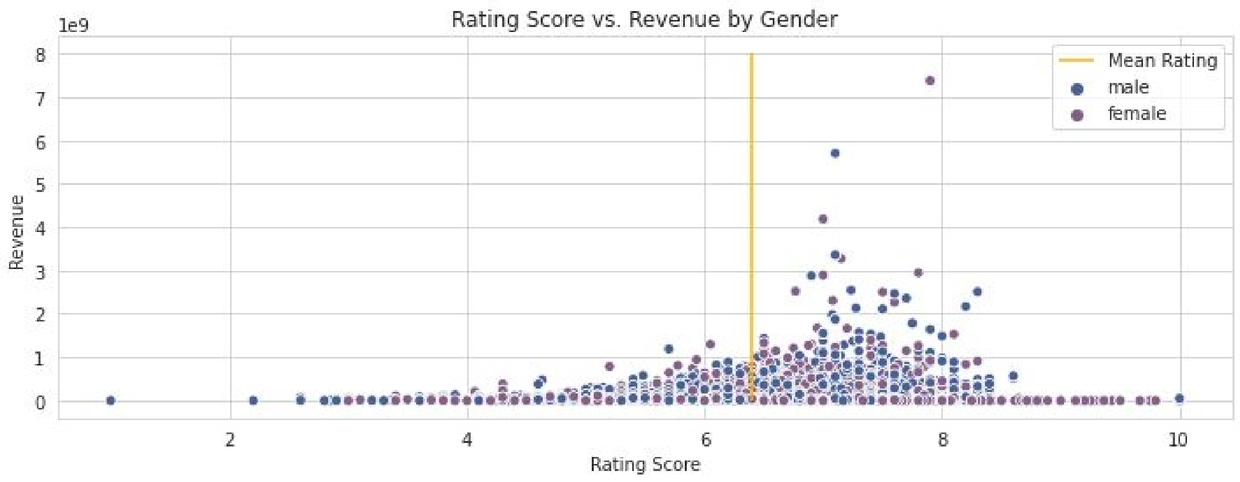
The visual analysis for the second capstone project began with revisiting the observations noted in the Data Story notebook of the first linear regression project. It was important to highlight any interpretability improvements that may have resulted from the way that the aggregation of the monetary values had been shifted, from summation to averaging, along with the removal of the outlier movies with the largest revenues. Two notebooks were created, which showed the results of applying these changes in succession. The first Data Story notebook was created with the dataset that had been aggregated, using the average monetary values for budget and revenue. The second Data Story (Optimized Target) notebook used that same aggregation method, but only after the movies with outlier revenues had been removed.

The first relation observed was between the actors' average movie ratings and the target variable, the revenue of those movies, when sorted by gender.

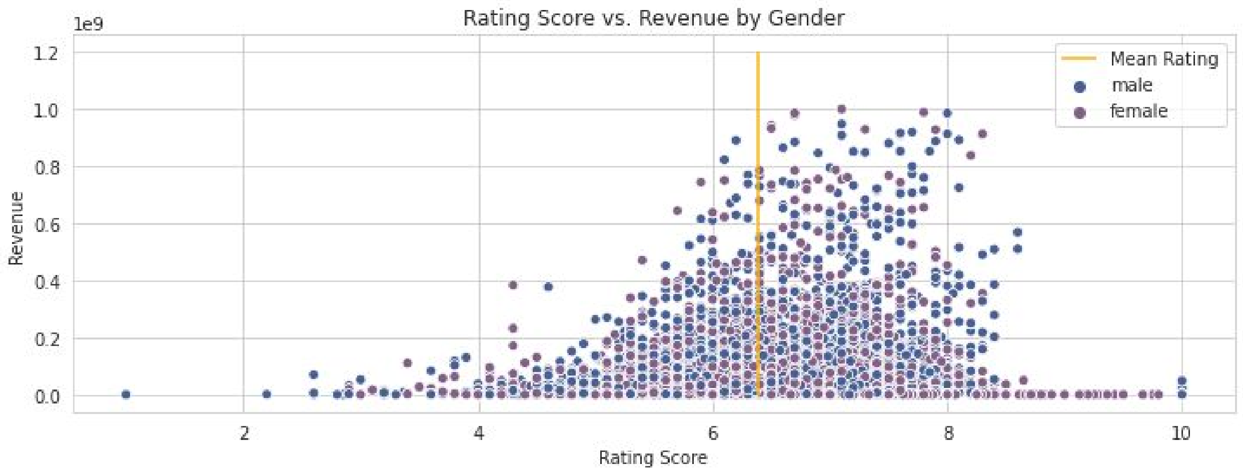


The plot for the aggregation of revenues using summation, named Actor Value in the first project, revealed gender bias in the data, as there were very few purple dots, signifying female actors, at target values above $8,000,000,000. Also, actors with higher target values counted in larger numbers at rating scores greater than the average, shown by the vertical yellow line.

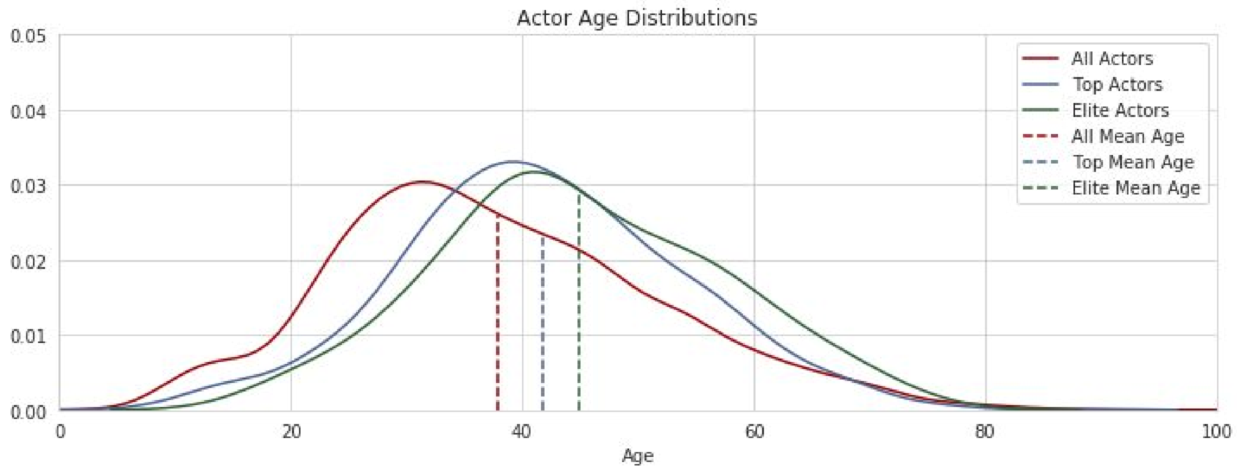
After the switch to the aggregation of revenues using averaging, called revenue in the second project, was made, the genders appeared more evenly distributed at the highest revenues.



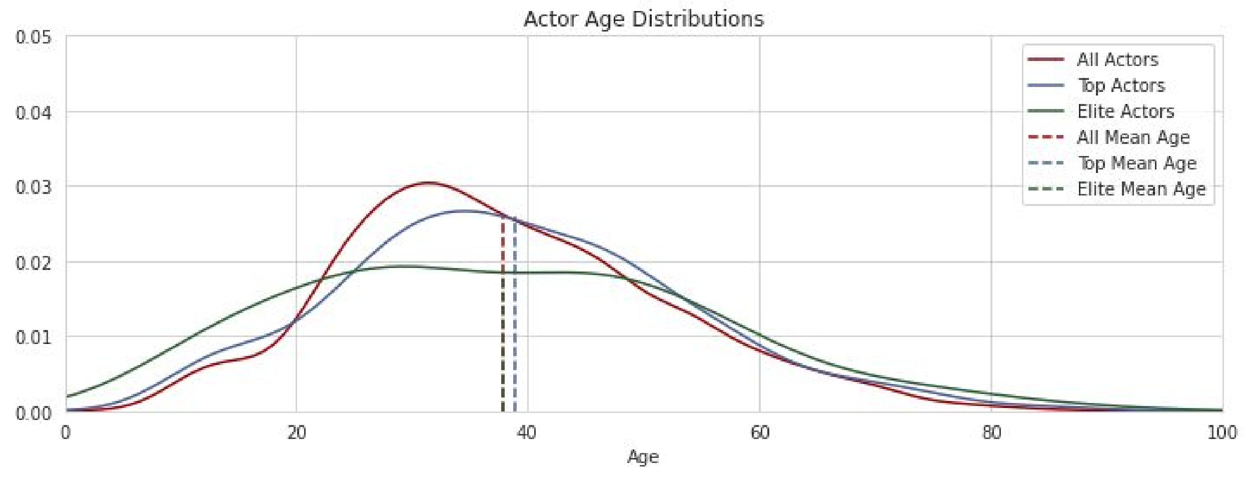
The relation between revenue and rating persisted. The Pearson correlation coefficient between the two was found to be 0.17. The key difference to notice was that the actors with the largest target value, now average movie revenue, were no longer obviously biased toward male actors. In fact, the Pearson coefficient between these variables was -0.02 for both datasets of the second project. This was the result of removing their advantage of having more movies to contribute to their revenue sum, since the careers of male actors were longer on average. Predictions made for the highest potential earners for the average movie would have been more biased toward them, when using the dataset from the first project. Removing the outlier movies did not improve this result. The genders are well distributed, and the top revenue data points are more tightly clustered.



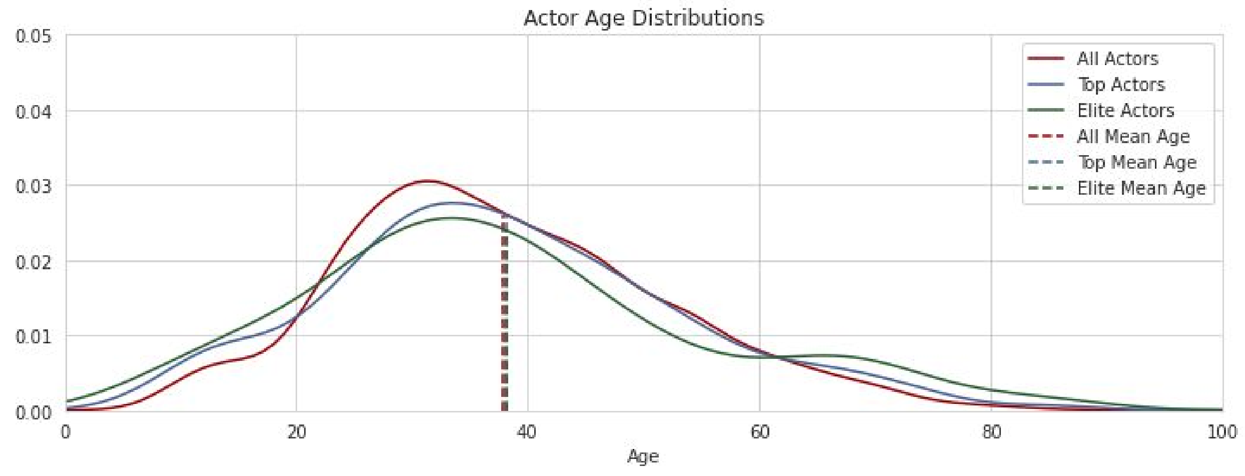
The analysis continued after the actors were grouped by various tiers, based on the average revenues of their films. The first tier included all actors. The second group, named the top tier, contained only those actors with average movie revenues that were in the top 10%, overall. The final tier consisted of actors whose average movie revenues made up the top 1% of all actors. These were the elite tier of actors. Note that the actors in the elite tier were counted in the top tier, as well. At this point in the notebook, the various predictor value distributions were analyzed with respect to the target variable. The changes in the actor age distribution were noteworthy.



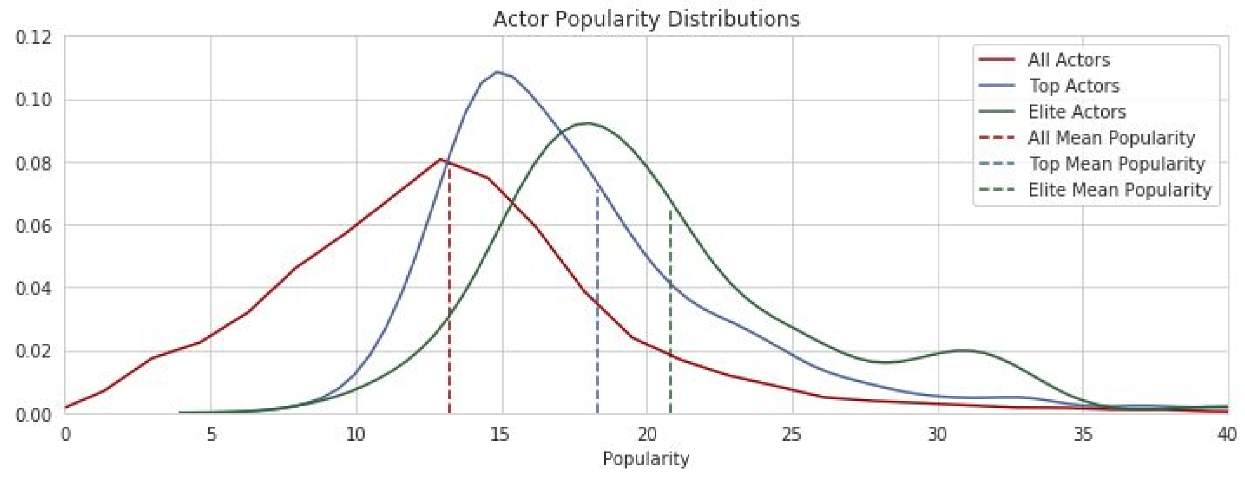
The average actor age increased with the progression to the upper actor tiers, when observing the data from the first project. These values started to converge, after the aggregation method was changed from summation to averaging.



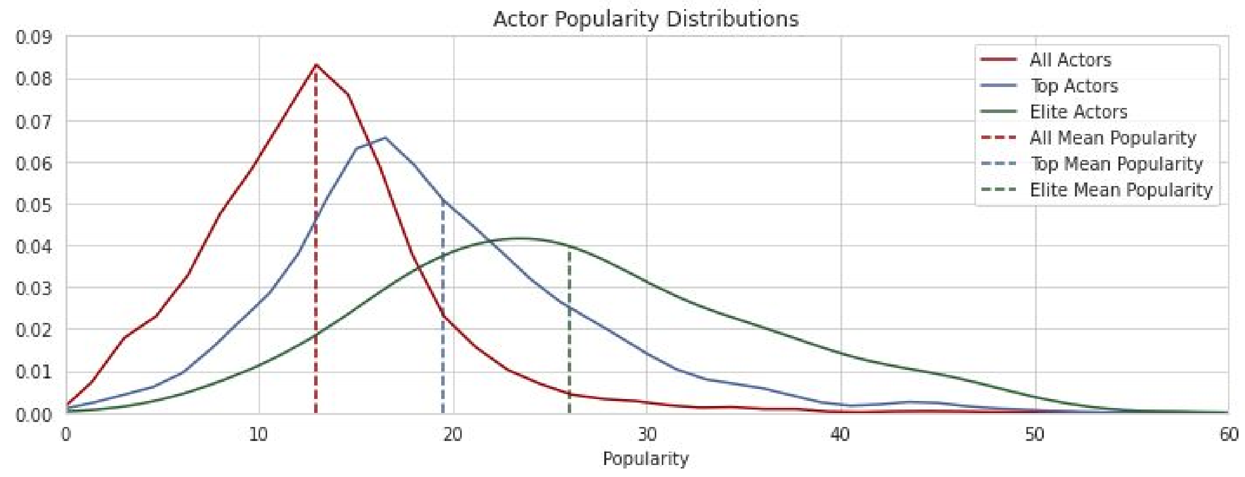
The effect of removing the films with the highest revenues completed this convergence.



The movie popularity rating was a TMDb proprietary metric, based on recent traffic pertaining to a movie on their website.

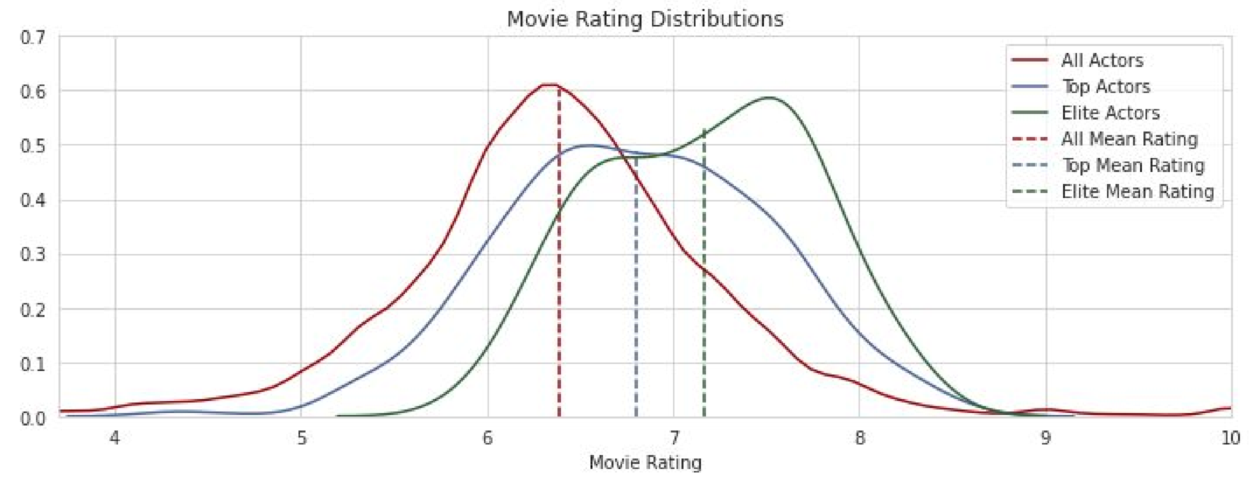


This value increased as the tiers progressed toward the elite actors.

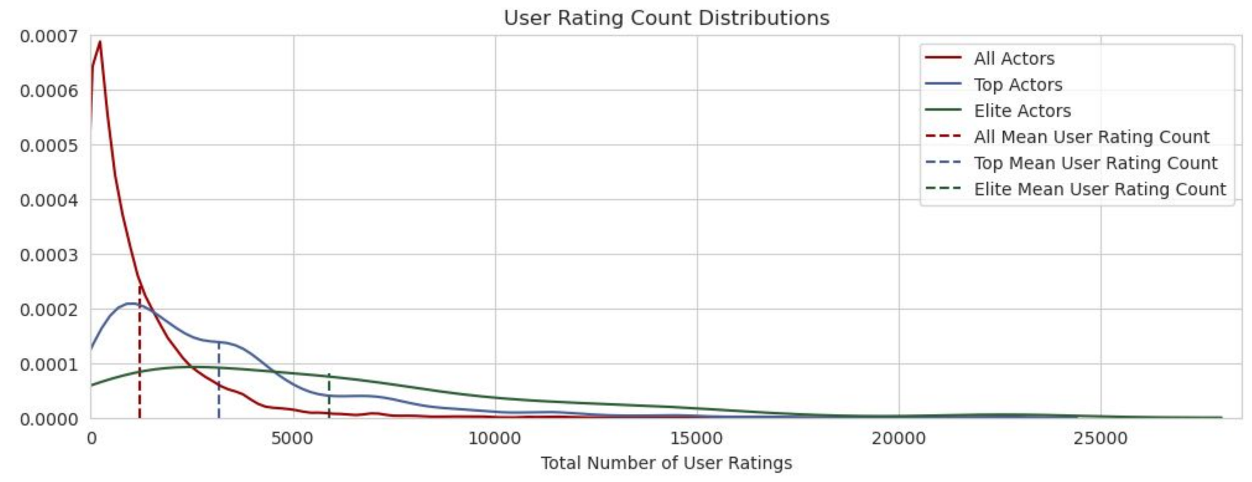


By the time the final dataset was reached, the popularity values for the top and elite actors had shifted higher. This was the result of removing a lot of the older blockbuster series and Disney animated films, which would not have as much current interest, as the more recent releases would have.

Both the average movie rating, and the total number of users who rated a movie, grew when observing the progression through the tiers of actors, which were derived from increasing thresholds of the actors’ movie revenues. This trend was present in every form of the data.



The user ratings for these films directly correspond to which movies had the highest ticket sales at the box office. The second mode, at the highest ratings in the elite actor distribution, indicated that some of those actors elicited strong tribal support from their admirers. This is the idea of star power, or the ability of actors with high name recognition to be able to drive ticket sales.



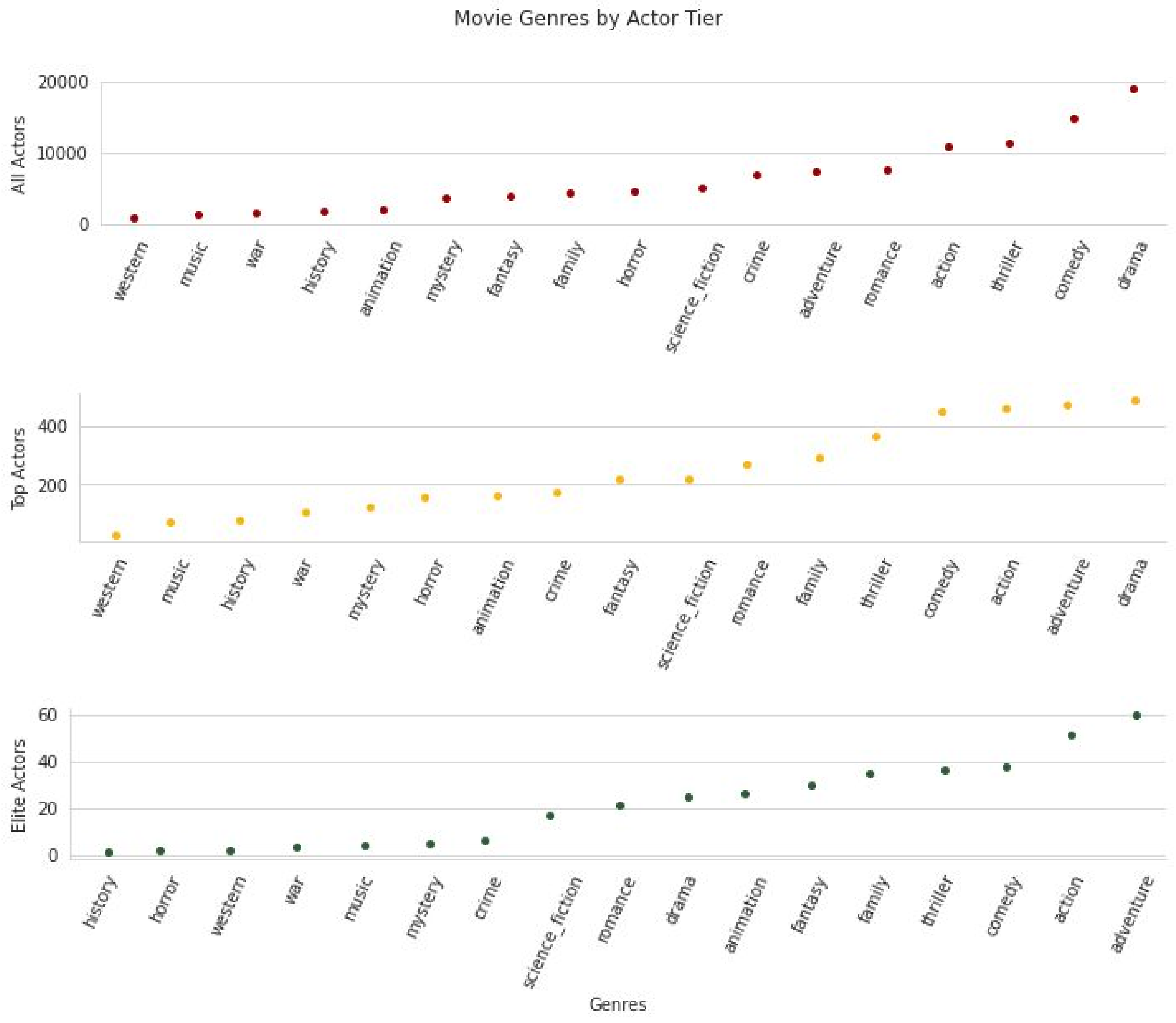
The elite actors dominated the rating traffic on the TMDb website, as well. Their average number was almost 6,000 individual ratings, which was more than 5 times the number of ratings belonging to the average actor.

Interestingly, before the removal of movie outliers, the Pearson correlation coefficients of several predictors flipped signs, when progressing through the actor tiers. The predictor, budget, went from 0.34 to - 0.32. Popularity changed from 0.33 to -0.2, and the number of movie ratings went from 0.44 to -0.26. Once the data was transformed to its final form, this behavior disappeared. It was possible that the observations for actors who were mostly, or exclusively, in movies with large revenues did not produce the same signal, as the observations of other actors. Dropping these outlier movies may have brought the entire dataset into better cohesion.

This would have the effect of giving the models a simpler training set to digest, producing more accurate predictions.

In reference to the full dataset, one of the predictor correlations with the target variable was significantly strengthened, after the removal of the outlier movies. The Pearson coefficient between budget and revenue went from 0.34 to 0.5. The average revenue dropped by a factor of 60, while the average budget only fell by a factor of 10. This suggested that the outlier movies had revenues that were inflated for their budgets, compared to the rest of the movies. Given that budget is the feature most closely correlated with the target, removing the outlier movies should produce a simpler dataset for model training.

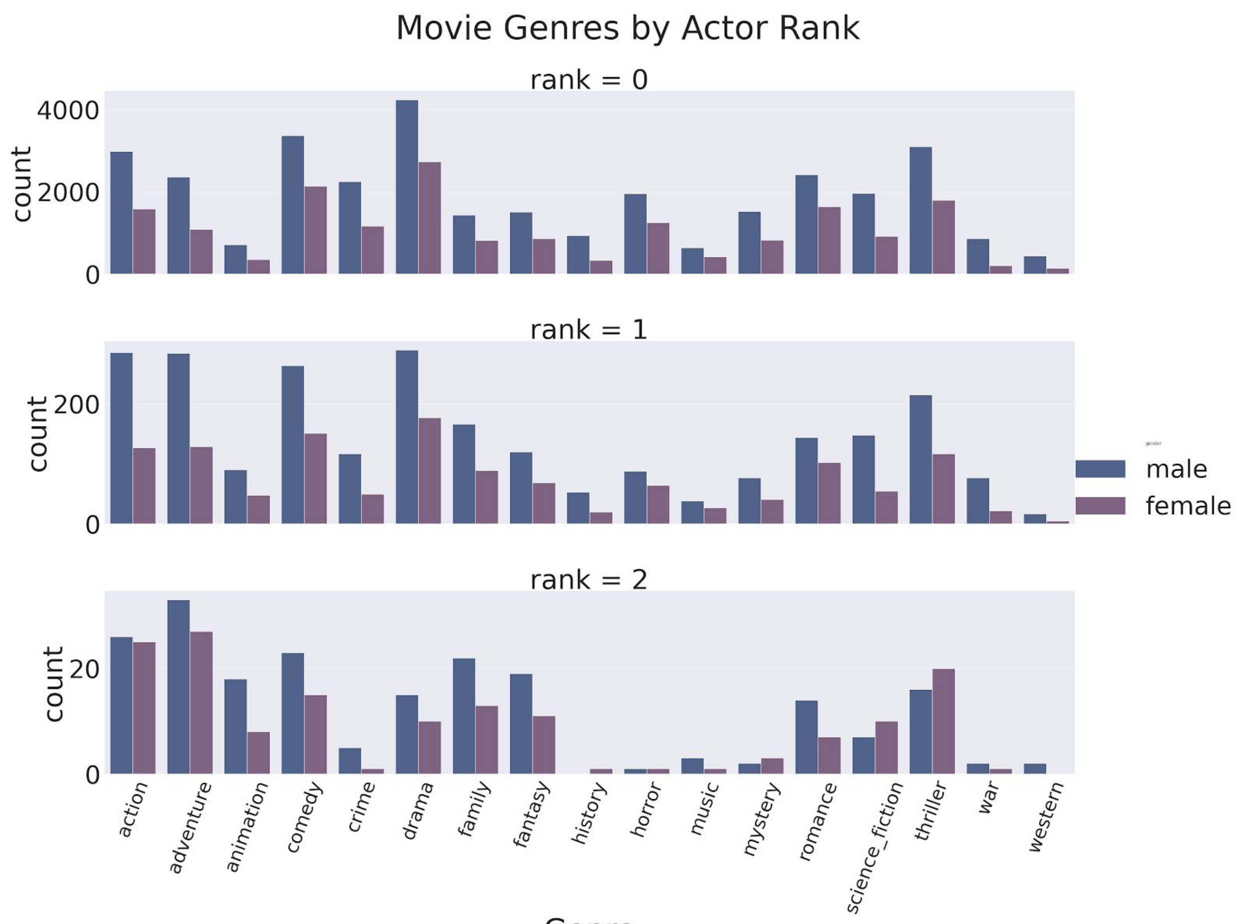
The final analysis that was revisited from the linear regression project concerned the distribution of the movie genres among the 3 tiers of actors.



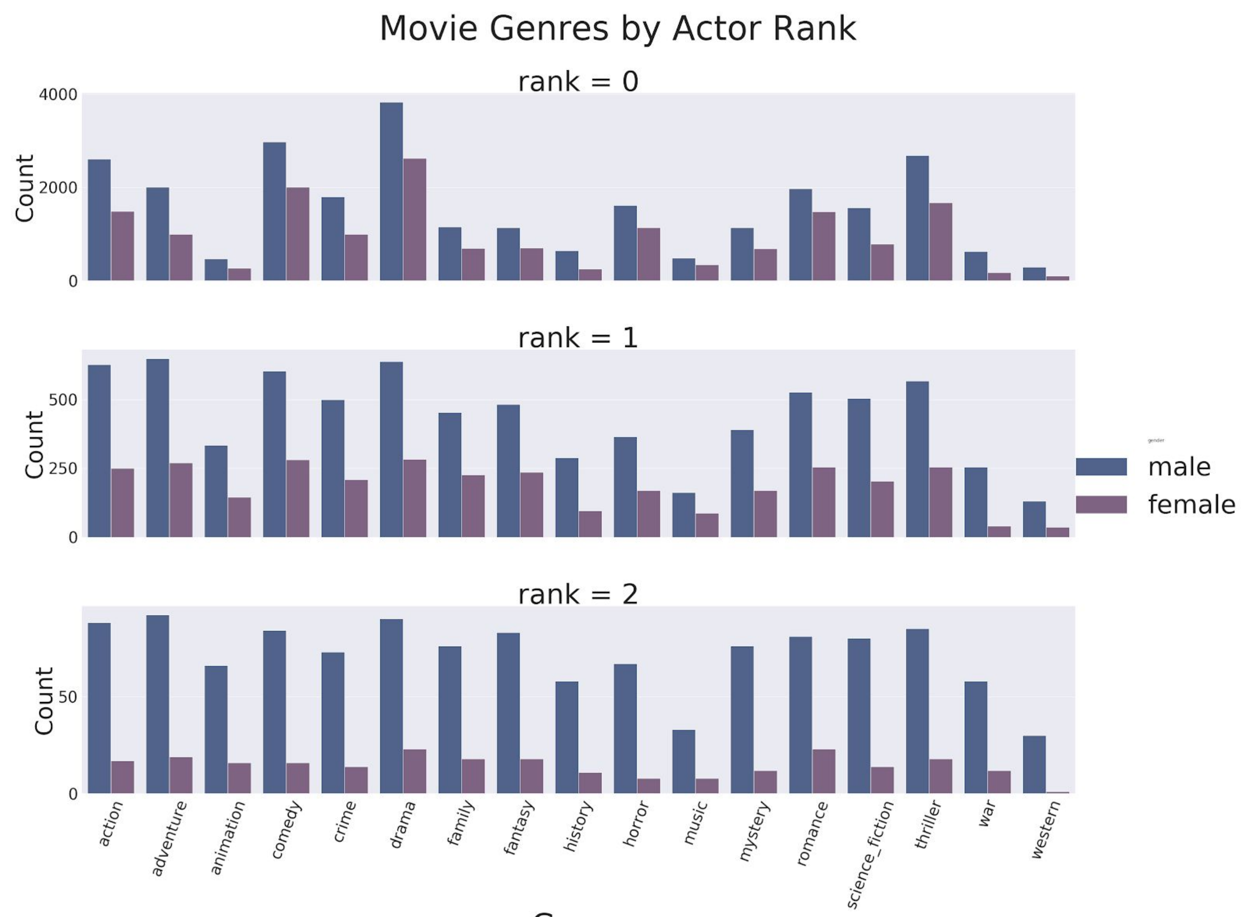
The drama genre dropped significantly in importance when observing the elite actors, which did not happen when using the dataset in the previous project. Although, comedies did have a higher count within that tier, this time around. Adventure movies began to rally for actors in the top tier, then went on to take the lead among the elite tier of actors. The genres with the lowest counts remained near the bottom, regardless of the form of the dataset.

In order to bring a more practical feel to the project, a deeper dig into the actual movies and actors behind the data was performed. The breakdown of the elite actors was observed, as they appeared in the various genres plotted above. In the first Data Story notebook of the second project, these observations were made, before outlier movies were eliminated from the dataset. In fact, the observation of the particular movies and actors behind the numbers, inspired that change, in order to eliminate the biases that were revealed. For instance, most of the appearances of elite actors who were in multiple adventure movies were acting in major film series, such as Star Wars, Harry Potter, and Lord of the Rings, among others. After the movies with the largest revenues were eliminated, what was left were mostly James Bond movies for actors with multiple adventure film appearances. Though technically still part of a film series, considering the regular turnover of actors in Bond flicks throughout the years, it is fair to say that those films were a good tradeoff for Harry Potter movies, which had a more regular cast throughout its series. The same could be said for family movies. What was a list of mostly Disney movies from the 50s and 60s became one with a broader selection, including a more recent Disney release, a Pixar film, and many other movies that were not in the animation genre, as well. The diversity of the movies and actors became richer, after the outlier films were removed.

As in the first project, the actors were ranked in their separate tiers. This time, the elite actors were excluded from the top tier. This gave rank 2 for elite actors, rank 1 for only the actors with average revenues between the top 10 and 1%, and rank 0 for those actors with revenues below the top 10%. Then, the counts of the genre appearances by each actor rank were plotted, while each genre was separated by gender. Note that the elite actors were not included in rank 1.



While this plot didn't show much that was different from the previous one, with respect to the genre distribution between the different ranks, it did show the male actors outnumbering the female actors in almost every genre at every tier. This was not so unusual, as female actors comprised just under 40% of all the actors in the dataset. The exceptions were found only in the rank 2 distribution. Thriller and science fiction were the genres most strongly represented by female actors in that rank. This observation was contrasted with the plot from the first project.



The plot from the first project showed a very strong bias toward elite actors being male. In contrast, the distribution between the genders became more representative of the dataset, after the target values were optimized. In fact, the female actors became over-represented among the rank 2 actors, after the outlier films were removed and monetary values were averaged. In the dataset from the first project, the female actors comprised just over 20% of the elite actors, where they represented over 43% of the elite actors in the optimized dataset. Recall, just under 40% of the actors were female in either dataset. The final choice of data representation increased model performance, while it reduced the gender bias of the data, something that should be ensured gets checked, when processing socioeconomic data.

Statistical Analysis

A different form of this dataset was analyzed in the first project, using resampling techniques, such as bootstrapping and permutations. A parallel analysis was performed on each of the two modified datasets in the second project. Hypothesis tests were performed to determine if the mean ages and age distributions were each identical between the genders. It was found that they were not. The Pearson correlation coefficient, between gender and age, was -0.22, as was shown in the Statistical Analysis notebooks from the second project. The same analysis was done to the revenue distributions with respect to gender. It was shown that these distributions were distinct, as well. The Pearson coefficient was much weaker, at -0.02. The results of all three analyses were nearly identical, regardless of the form the target variable took. As such, the resampling processes used in this analysis was thoroughly explained in the detailed PDF from the first project.

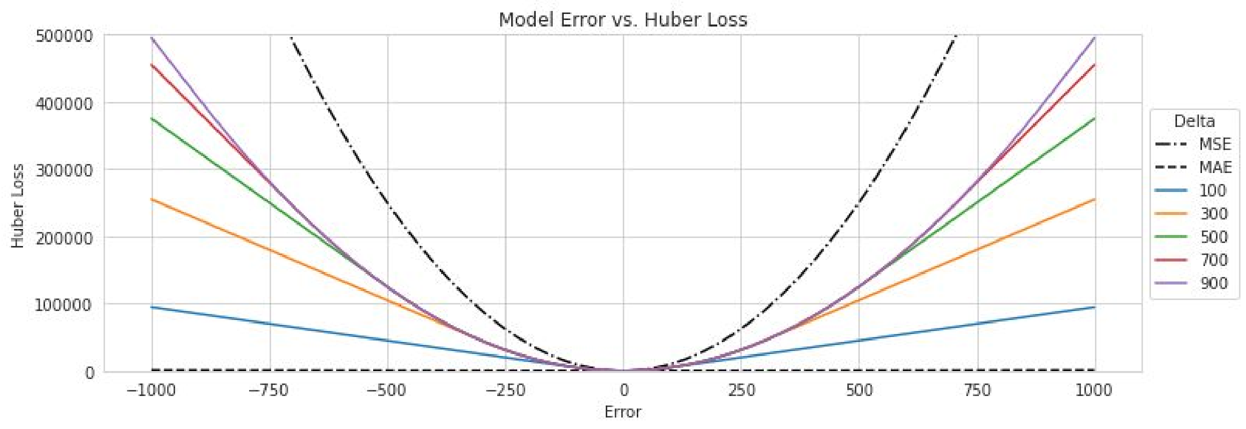
In-Depth Analysis

The model optimization process, found in the XGB Regressor (Optimized Target) notebook, began with choosing the most appropriate functions to use to minimize the prediction errors, specific to the dataset. The chosen tree boosting library for this project, XGBoost, provided the ability to use a custom objective function for metric performance monitoring, during model training. The XGBoost training module required the objective function to return 2 values, the gradient and the hessian of the evaluation metric, also called the loss function. When given the model predictions and the true data points, this second function was used to calculate the prediction errors. These errors were used as evaluation metrics to assess when training should be halted, called early stopping, due to their lowest values remaining unchanged, after a specified number of additional boosting rounds. For this dataset, target values ranged in scale from under 10 dollars to nearly 10 billion dollars. Considering that many prediction errors were going to be very large, the mean absolute error (MAE) was chosen, as the evaluation metric, as opposed to the mean squared error (MSE). The MSE would have penalized models more than the MAE did, for those predictions that were large in scale. Because the target had been transformed using the natural log function, the loss function was written from scratch to get the prediction errors in dollar amounts. To accomplish this, it was necessary to transform both the predictions and the test labels, using the exponential function, before the loss function was applied to them.

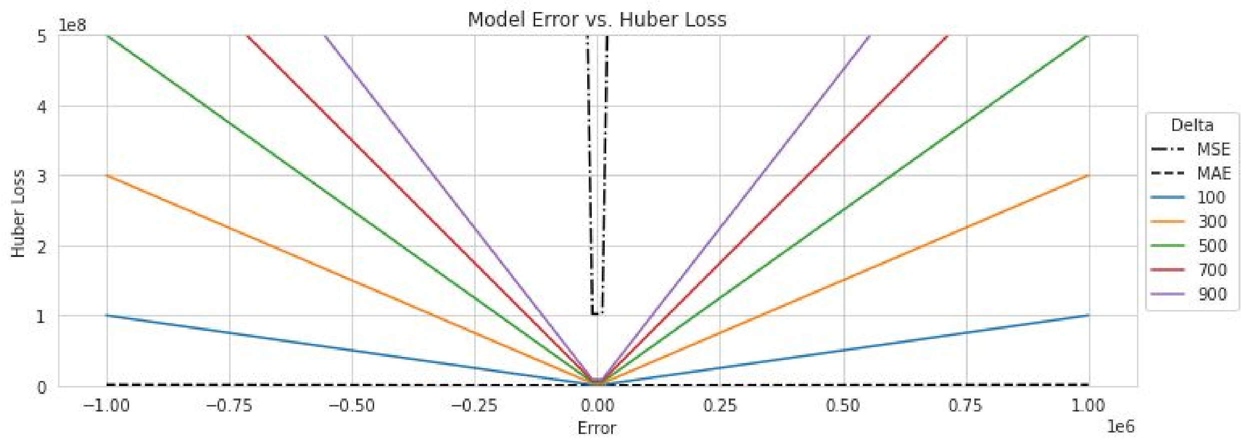
After the evaluation metric was chosen, the objective function needed to be selected. The Huber objective function had the ability to adapt to the wide range of prediction errors that were expected.

$$L_\delta(error) = \begin{cases} 1/2 * (error)^2, & \text{for } |\text{error}| \le \delta \\ \delta(|\text{error} - 1/2 * \delta|), & \text{otherwise} \end{cases}$$

First, the value for the hyperparameter of this function, delta, needed to be set. The delta value would set an absolute error threshold, below which the behavior of the Huber function would act as a squared error (MSE).



For errors that were above the value of delta, the Huber function would behave like an absolute error (MAE).



Using this function would enable the models to make more accurate predictions, given the wide range of the target values.

Since the Huber function was not everywhere twice differentiable, and the hessian values required by the XGBoost training module were the results of taking the second derivative of the optimization function, the Pseudo-Huber function needed to be used.

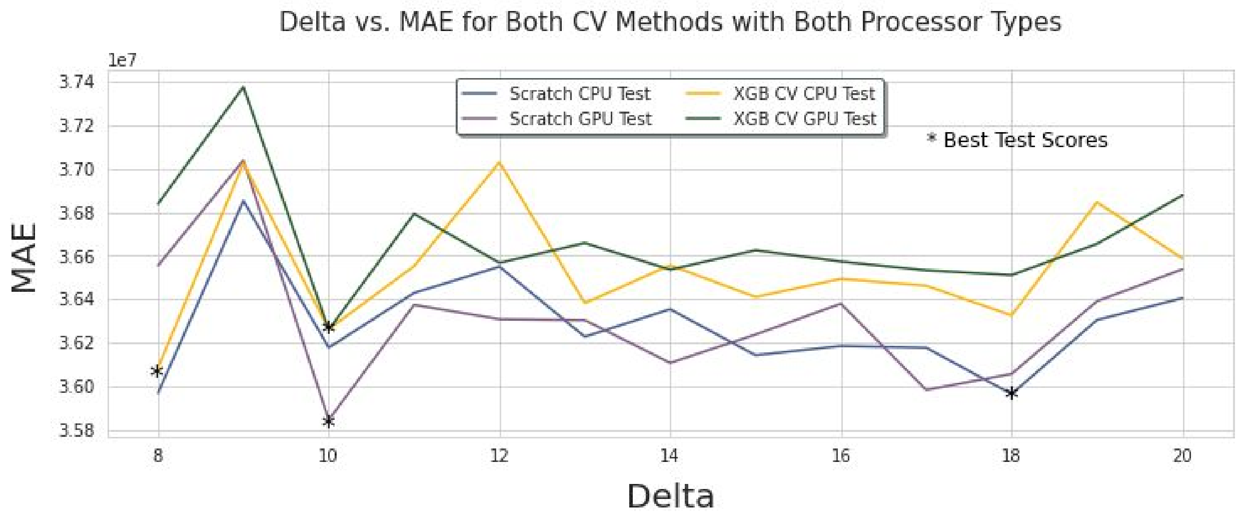
$$L_\delta(error) = \delta^2 * [(1 + (\text{error}/\delta)^2)^{1/2} - 1]$$

This function replicated the behavior of the Huber function used by the XGBoost training module to perform gradient descent, but was everywhere twice differentiable.

A 10 fold cross validation (CV) function was used to select for the best model, to give it better generalizability on new data. The built-in CV function from XGBoost performed differently than a custom CV function that was written from scratch.

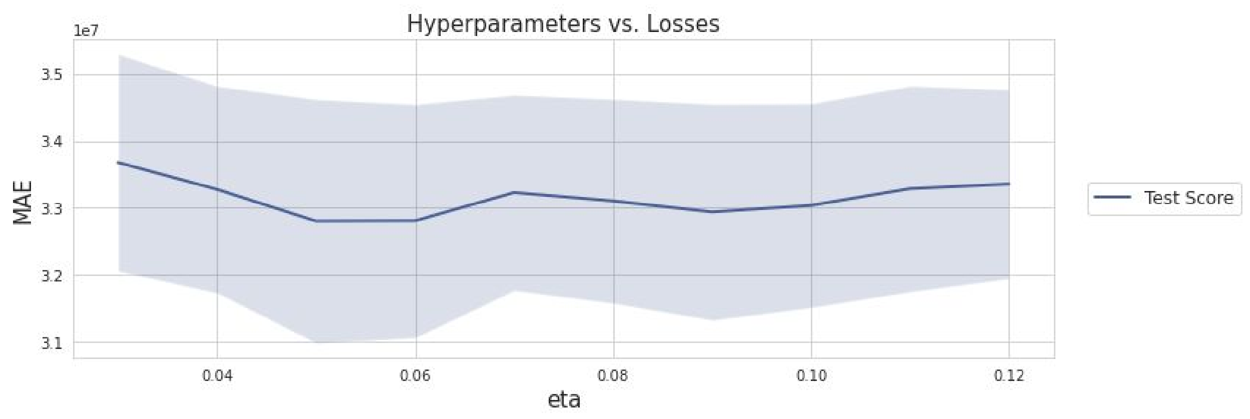
Because accuracy can vary, depending on the hardware doing the calculating, training was performed by using either CPU or GPU operations. The performance differences between these two options was evaluated during the choosing of the optimal delta value to use in the Pseudo-Huber function. Time complexity costs were considered, also, as run-times for the hyperparameter search for the boosted models could get quite long.

Training sessions were run over a range of delta values to find the one which optimized the evaluation errors best. Delta values between 8 and 20 were used , in conjunction with each hardware option and the two CV functions.



The best model found through this process was the result of using GPU calculations with the custom CV function and a delta value of 10. The average 10 fold CV test error, using default boosting hyperparameters, was lowered to $35,843,487, after these optimizations were used. The error had been reduced by an additional 10% from the baseline value. By using the GPU option, the training times were slightly reduced, as well.

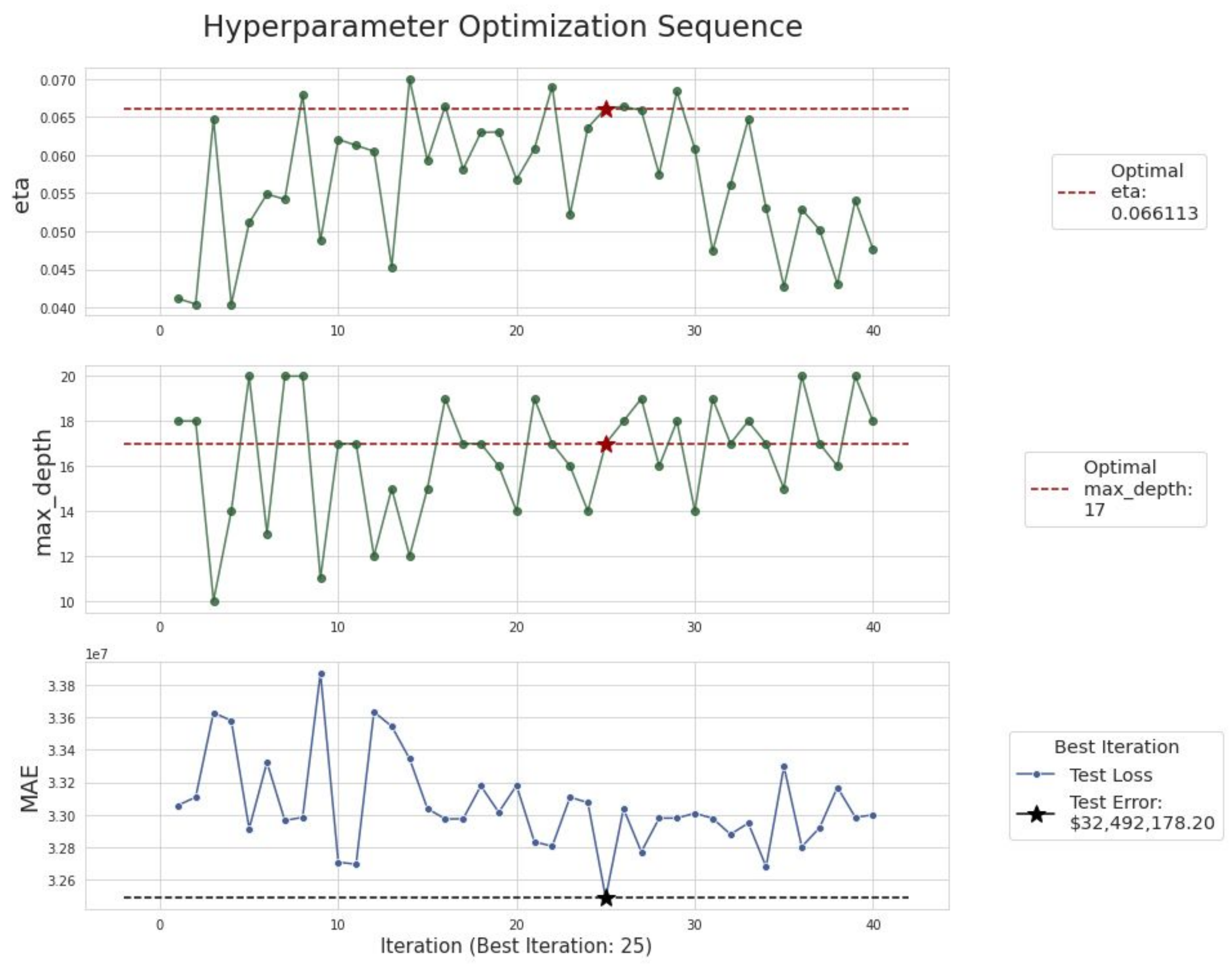
The final optimization phase involved a search for the best boosting hyperparameters. An informed (Bayesian) search technique was performed with the Hyperopt algorithm. The informed search method offered time cost benefits over a brute force grid search, which would have tried all combinations of hyperparameters throughout the search space. Instead, the choices of hyperparameters were based on the results of previously chosen models. Each round of tuning used a range of values, selecting one hyperparameter at a time. Each search round ended after a preselected number of tries did not produce a better model. This was another instance of early stopping. After one hyperparameter was selected, any others that were previously tuned were double checked for retuning. The average 10 fold CV test error was the metric used to evaluate which model had the best set of hyperparameters. A very limited grid search was performed to determine the best initial tuning range to set for each hyperparameter.



After the results were plotted, local minima were revealed. These were the values that made up the initial search spaces.

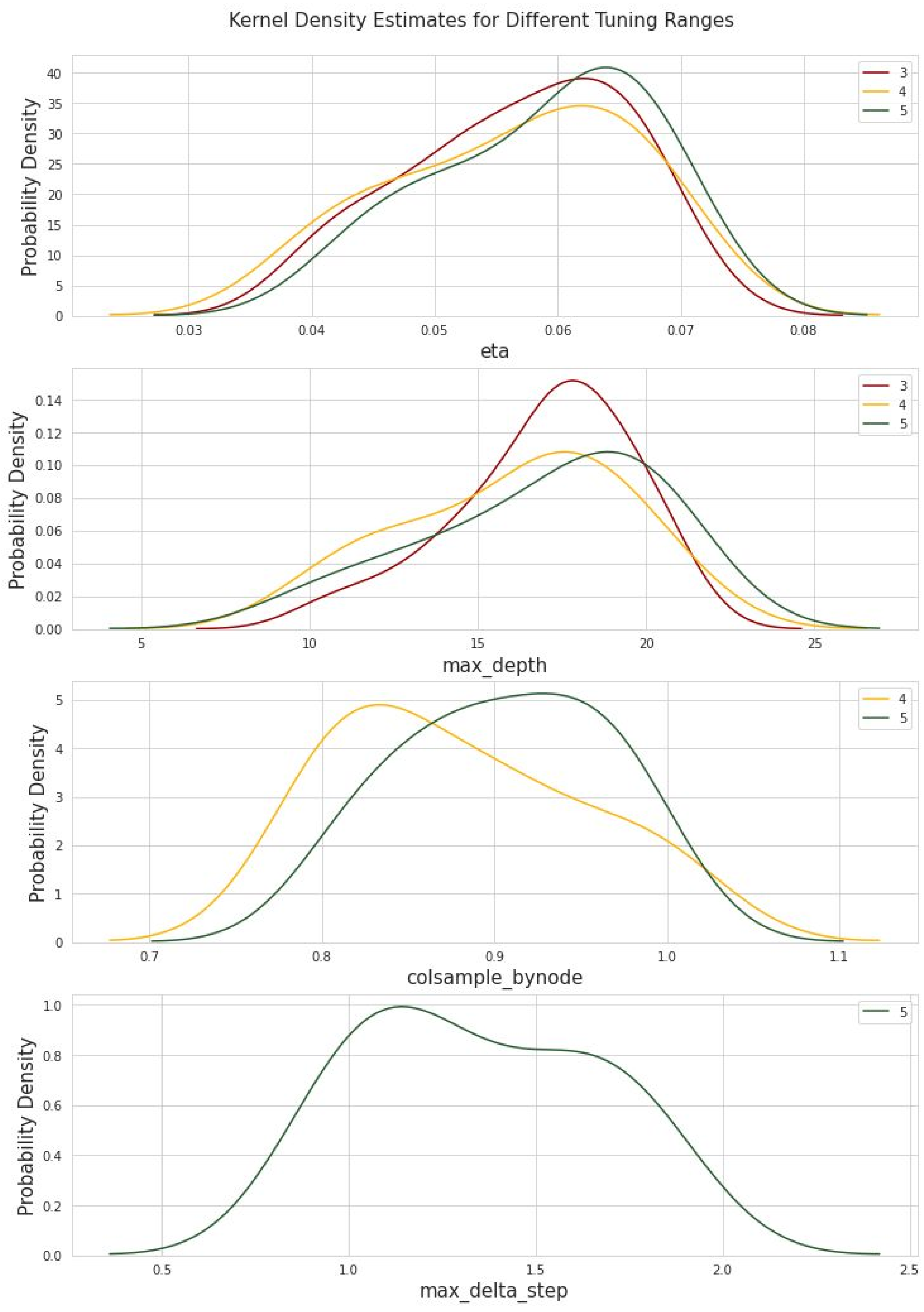
The XGBoost training module required that the maximum number of boosting rounds be predetermined. The initial value was set to be 500 rounds. With early stopping set to 100 rounds, this allowed the models to become quite complex. The tuning of regularization hyperparameters would balance this complexity.

For each round of tuning, the choices of hyperparameters selected by the informed search algorithm were plotted, along with their associated test errors.



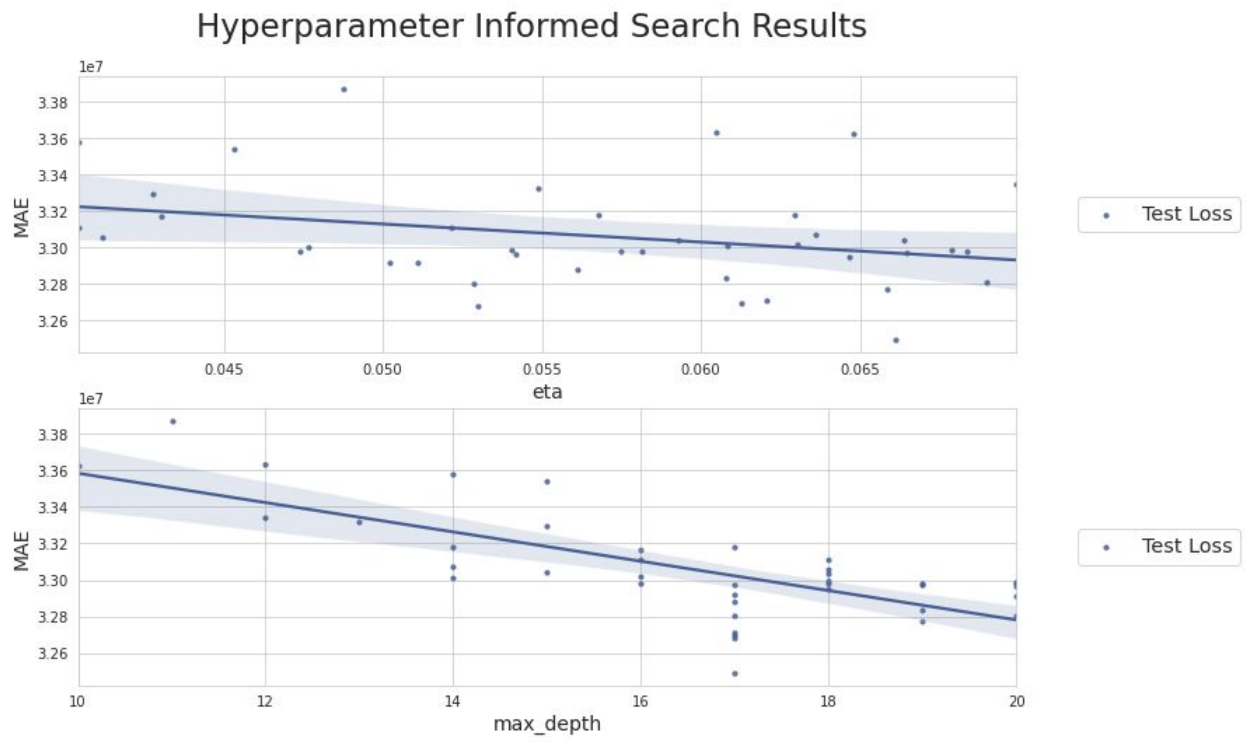
These plots were examined to determine if the search spaces needed to be expanded, as occasionally trial runs produced many models that preferred values on the cusps of their allowed ranges.

The kernel density estimates of the distributions of chosen hyperparameter values were plotted, as well.


Kernel Density Estimates for Different Tuning Ranges
Probability Density
eta
3
4
5
0.03
0.04
0.05
0.06
0.07
0.08
0
5
10
15
20
25
30
35
40
max_depth
5
10
15
20
25
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
colsample_bynode
0.7
0.8
0.9
1.0
1.1
0
1
2
3
4
5
max_delta_step
0.5
1.0
1.5
2.0
2.5
0.0
0.2
0.4
0.6
0.8
1.0

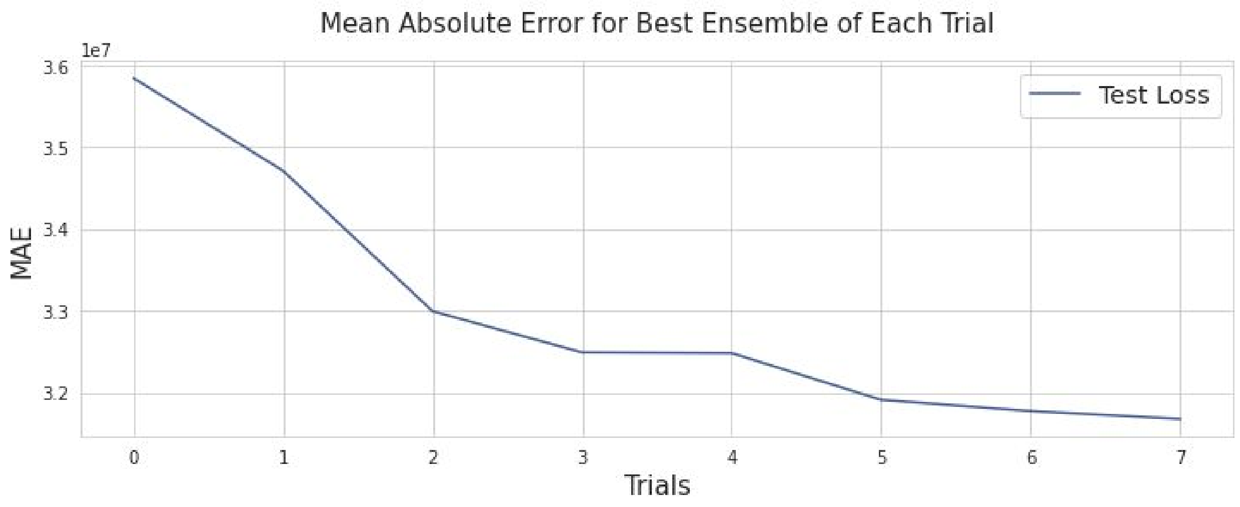
These plots complemented the search space visuals to confirm any needs for hyperparameter range adjustments.

Finally, regression lines were placed on scatterplots of the hyperparameter values vs. the test errors.



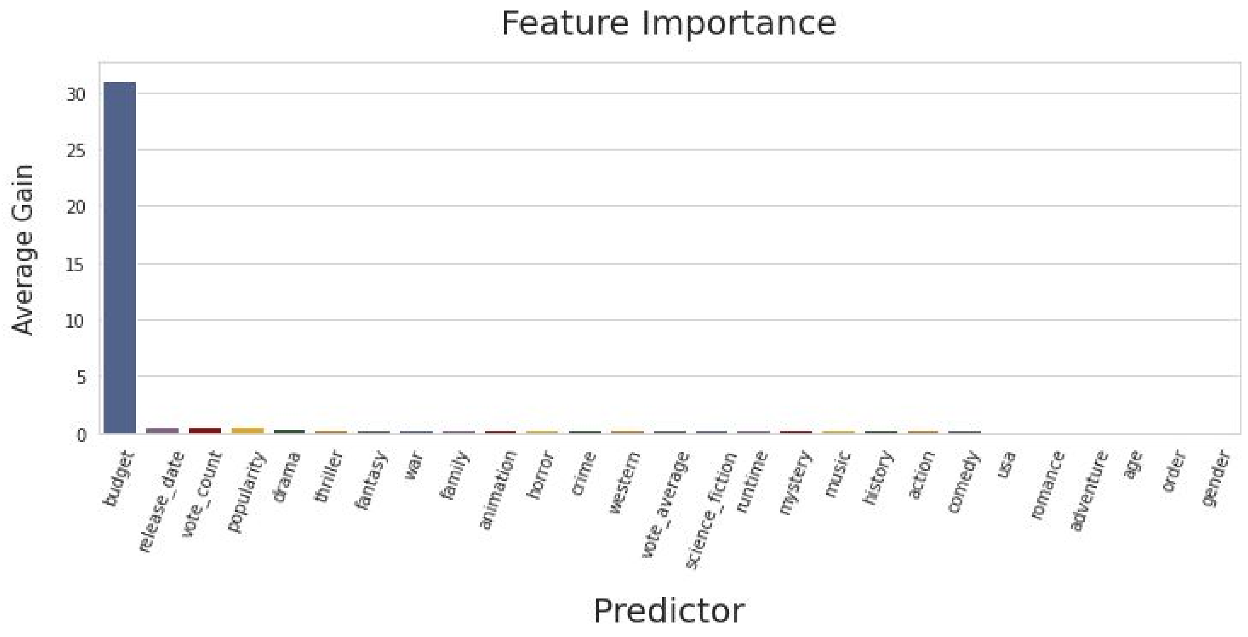
The vertical tilting of these lines would indicate the possibility of finding better performing models at hyperparameter values in that direction, possibly beyond the current range for that round.

This optimization algorithm was performed for 11 hyperparameters. Only 4 of them yielded improved models from the previous tuning round. Two of them, eta, also known as learning rate, and max delta step increased the regularization of the residual errors at the leaf nodes. One of them, colsample by node, adjusted a sampling method, which selected a subset of features to use at every leaf node. The last one, max depth, controlled how well the models learned the structure of the dataset, by allowing the individual trees to make more decisions, as they grew deeper. A hard cap of 20 layers was set on this hyperparameter, as deeper trees became extremely time cost inefficient. After the last hyperparameter was tuned, the limit was raised on the maximum number of boosting rounds to 1,000. This, set with an optimal number of early stopping rounds of 200 iterations, provided one last push to squeeze out any last bit of performance.



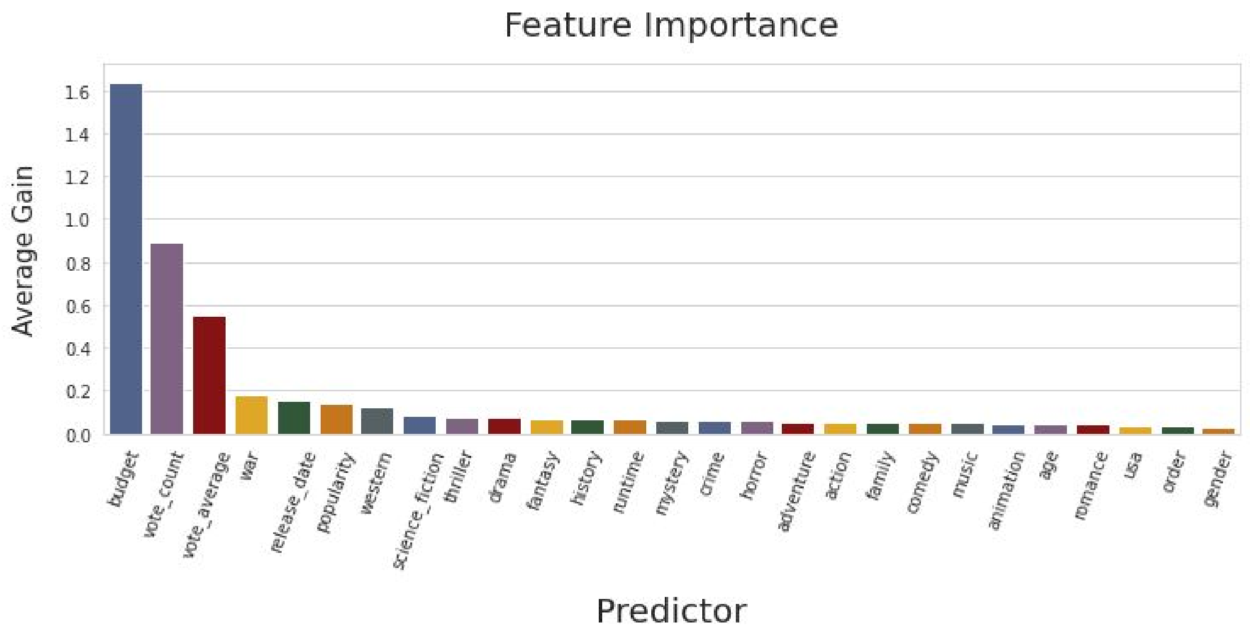
Tuning the hyperparameters gave an improvement of an additional 17%. The average test error was decreased to $31,678,760 for the final tuned model, using an optimized target variable and superior optimization function. The total reduction in the average 10 fold CV test MAE, from start to finish, was 44.2%.

The initial boosted tree ensemble, before the start of the hyperparameter tuning, was composed of many weak learners.



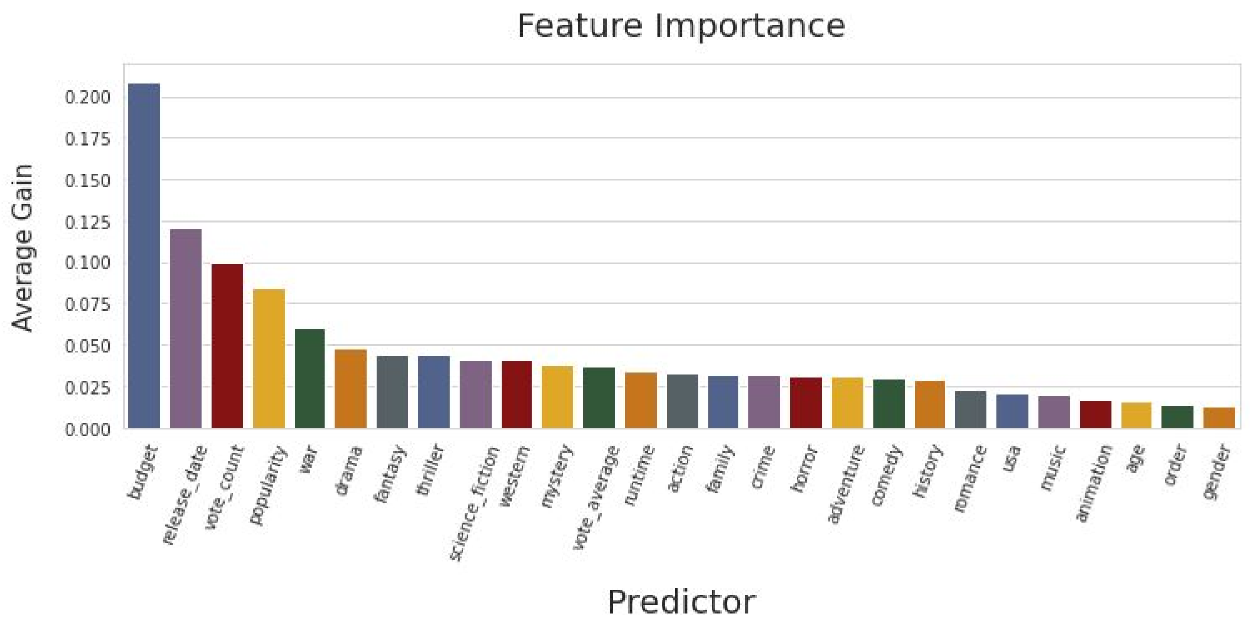
Nearly all of the decisions made by the untuned model were based on values of the budget feature.

As the hyperparameter selections progressed, the trees were able to find more structure in the information.



After tuning the first 3 hyperparameters, the best model was using a richer variety of the features to make its predictions.

By the end of the optimization algorithm, the best model was very complex.



Four of the features were being used extensively to predict, and they were all used to some significant degree.

After the final model was selected, it was used to make predictions on the validation data to observe the models performance on out-of-sample data. The average 10 fold CV validation error was $38,963,668. The final model was 77% as accurate as it was on the CV training data. This decrease was not surprising, as the size of the validation dataset was 25% of the size of the training set.