Stuart Kernohan

Movie Rater - ML Project

Stuart Kernohan, Mason Ho, Ishaan Bhardwaj, Harsha Vegiraju, Adyant Mishra

Introduction:

Literature Review

To predict movie success, machine learning algorithms like Random Forest Regression, linear regression, and Gradient Boosting were shown to be promising.

Random Forests are beneficial for complex datasets, because they combine many decision trees to prevent overfitting through the process of cross-validation [1].

Multiple Linear Regression is an essential tool as it provides simplicity and clarity of interpretation of results in complex, multivariable systems. [3]

For datasets with possible outliers, missing values, and higher cardinality categorical values, Gradient Boosting can provide powerful insight into finding non-linear relationships.[2]

Dataset Description:

This dataset contains data on 5000 movies, collected from The Movie Database (TMDB). There are four columns in this dataset: movie_id, title, cast, and crew. We will gather additional data on these movies including genre, average user rating, and user rating count. Database Links:
Kaggle
TMDB

Problem Definition:

Problem:

We will use data from TMDB to predict the success of a given movie. We will use inputs such as cast, crew, and genre to make this prediction. Success will be determined based on the average rating of users on TMDB. The issue is figuring out the extent to which each of those variables may affect a film's success level.

Motivation:

Lots of major film companies spend substantial time and money towards a film. Thus it would be extremely helpful if there was a way to predict the success the movie may have before starting production so that resources are not needlessly wasted.

Methods

Data Preprocessing:

First, we explored the ID's and shapes of our datasets, to ensure that every ID is unique. The next step was to get the popularity score of the lead actors in these movies. The lead actors were extracted from the cast column in the credits dataset, and then an api call was used for each actor to get their popularity score. We are utilizing the three most important leads for each movie, and rows missing this data were dropped (~40 records). A similar process was done to add in the director popularity score. Once this was done, we converted the release date column into 'years_since_release', a form that our ML models can handle. After all of this, we joined the columns necessary between the movie and credits datasets.

Implemented ML Algorithms:

The method we have implemented:

- Linear Regression Linear regression is an ideal starting point as it is a simple model and its features are easy to implement. This will give us a baseline to compare the other models to.
- Random Forest Regression Random forest regression is an ensemble model that can capture nonlinear relationships between the features and the target variable. It can also handle high-dimensional data which we will have with numerous actors and crew members.
- Gradient Boosting Gradient Boosting / Extreme Gradient Boosting is the last method we plan on trying, and can improve accuracy by iteratively minimizing errors in prediction. Similar to random forest, it is also good for feature rich datasets.

Results and Discussion

Quantitative Metrics Overview

There are three quantitative metrics used. 1. Mean absolute error (MAE), which makes it easy to understand how far off on average our predicted values are from actual values. 2. Mean squared error (MSE) which is good for more heavily penalizing large errors, and 3. R-squared which explains the percentage of variance in the target explained by the features of our model.

Linear Regression:

1. Mean absolute error (MAE): 0.632
2. Mean squared error (MSE): 0.889

3. R-squared: 0.329

   Random Forest:

4. MAE: 0.576
5. MSE: 0.634
6. R-squared: 0.524

   Extreme Gradient Boosting (xgboost):

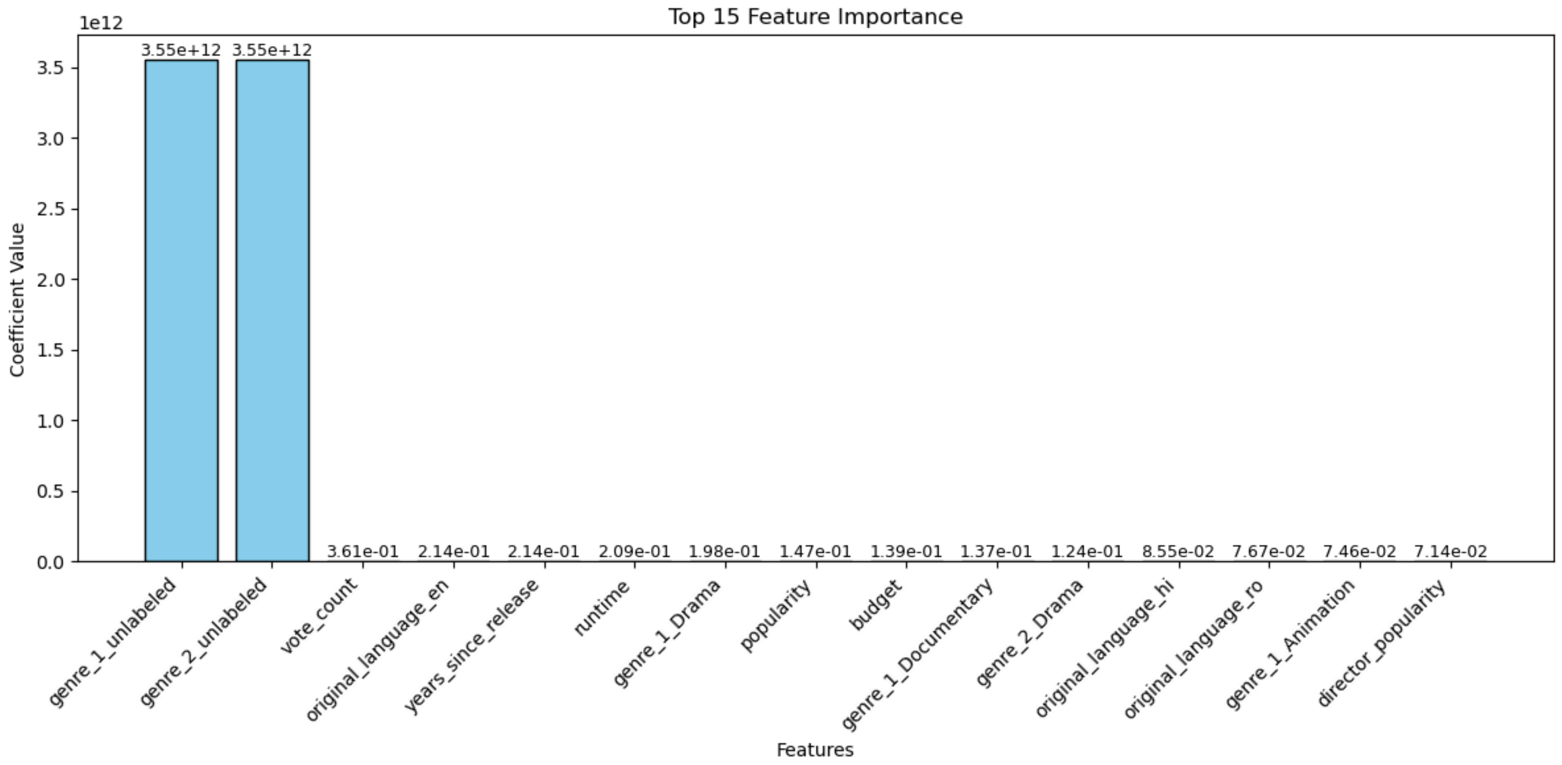
7. MAE: .500
8. MSE: .494
9. R-squared: .565

Summary of Model, Results, and Key Metrics

Linear Regression

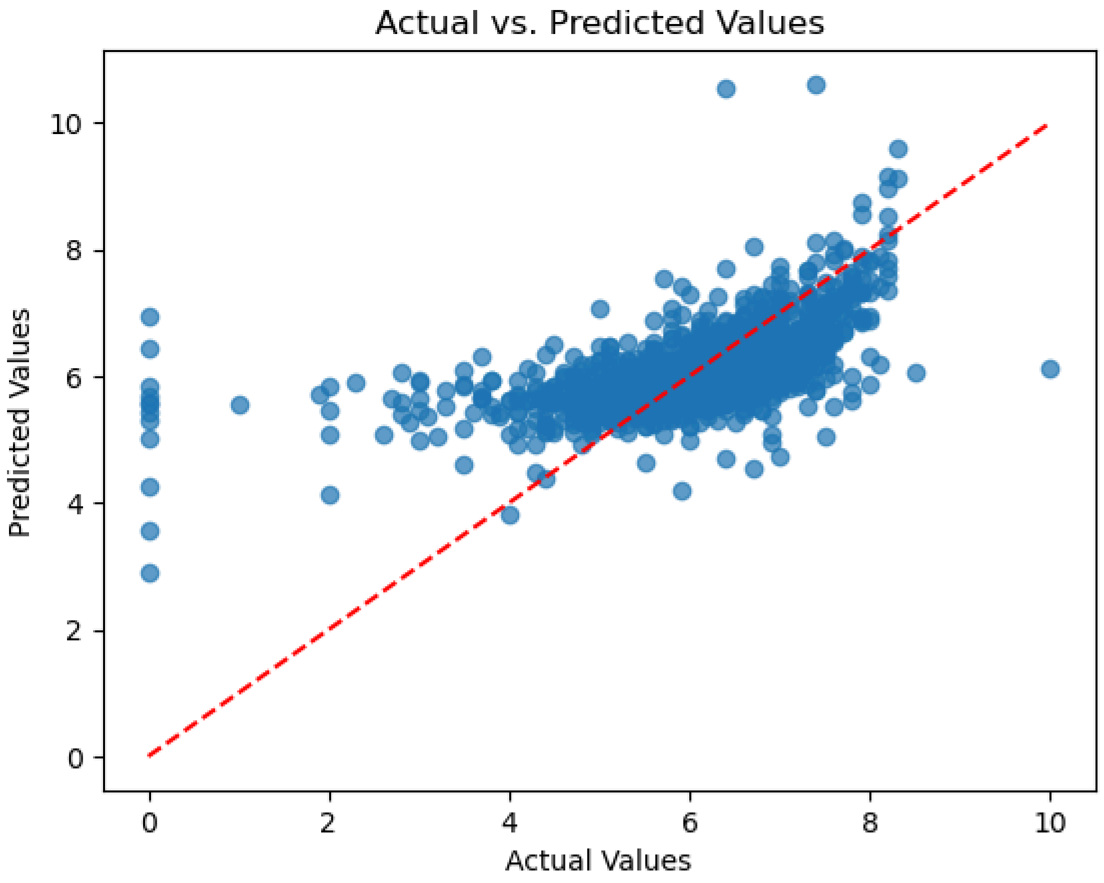
Visualizations

Top 15 Features



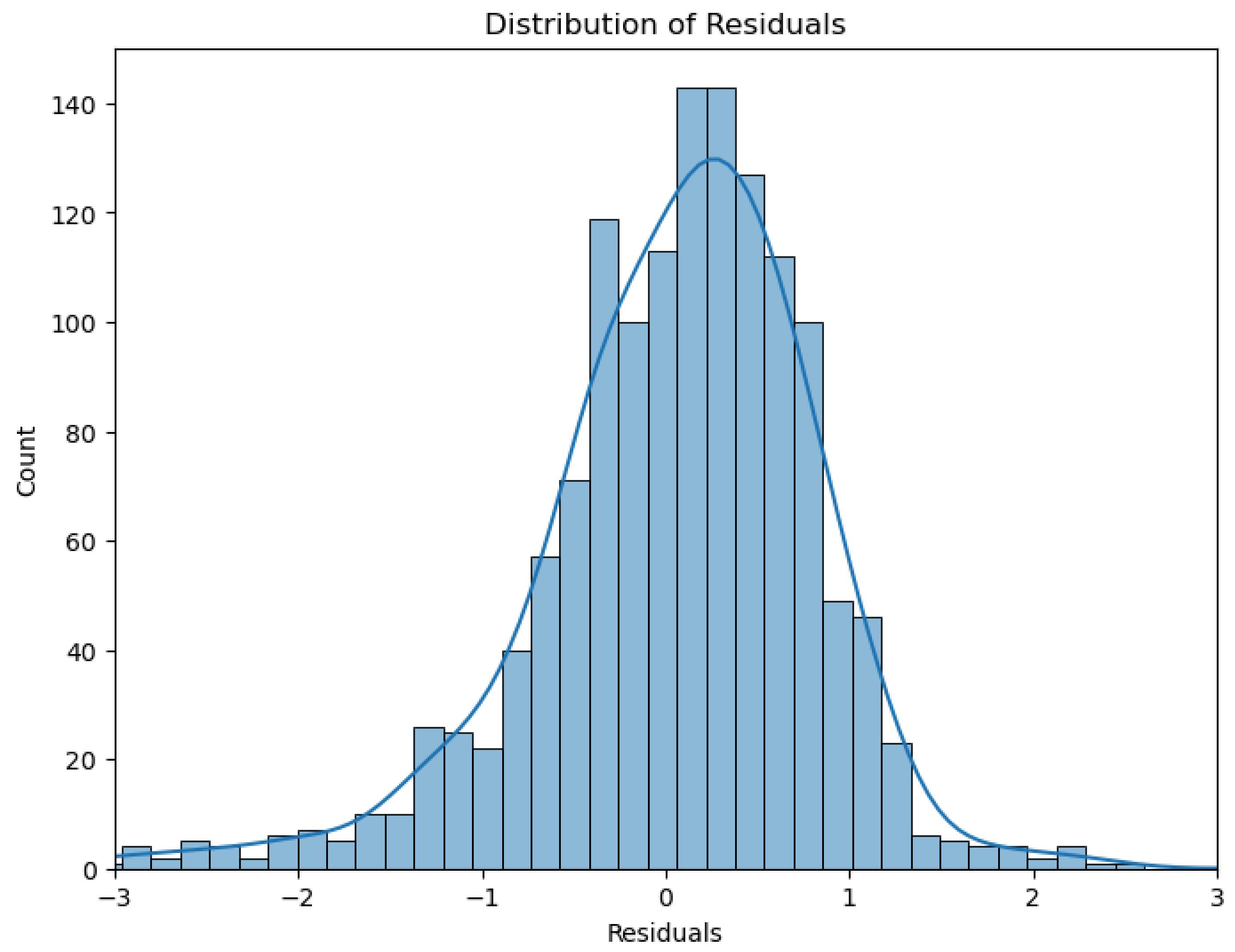
We can see that actually the linear regression gets very confused here as it believes that `genre_1_unlabeled` and `genre_2_unlabeled` are features that contribute the most to the model's predictions. This is clearly not good as the bias is high. It is not actually detecting the patterns in the data at all and is assuming that the movies without a genre for some reason do much better than movies with a genre. This implies that the model is truly nonlinear and trying to fit a linear model into something that is inheriently nonlinear will never work.

Predicted vs Actual



We can see that the model's predictive accuracy actually is not the best as there is a curve upwards as well as a lot of variance. This also implies that the data may actually be nonlinear and a simple regression model is not able to capture such data.

Distribution of Residuals



We can see that the distribution of residuals is not actually centered at zero and is actually slightly leaning to the right. This implies that the model underguesses the actual value frequently as that means that y_actual - y_predicted > 0. There is also a high amount of variance in this graph which implies that there is a lot of inaccuracies in the predictions. This is in line with the MAE and R2 errors that we saw previously.

Quantitative Metrics

Mean Absolute Error (MAE):

The MAE of 0.632 shows that there is actually a lot of error between our predicted and actual ratings. This is in accordance with the charts shown above and implies that this model is not very good as perhaps our dataset is nonlinear.

Mean Squared Error (MSE):

The MSE of 0.889 shows the average squared difference between the predicted and actual ratings. This is again a very high value and this implies that there is a lot of bias still left in the model. This was already made apparent from the features importance graph with the model capturing unlabeled genres as somehow the most important thing.

R-Squared Error (R2):

R-squared (R^2) of 0.329 suggests that the model captures about 33% of the variability in movie ratings. This is extremely low and implies that there is an extreme amount of bias as the linear regression model is too simple for the actual dataset. This shows that more advanced and nonlinear approaches might have to be considered.

Model Analysis

The linear regression model is the second model we considered for the dataset. We wanted to see if it was possible to try to use a simpler model to learn our data as we saw that our random forest model was prone to a high degree of overfitting.

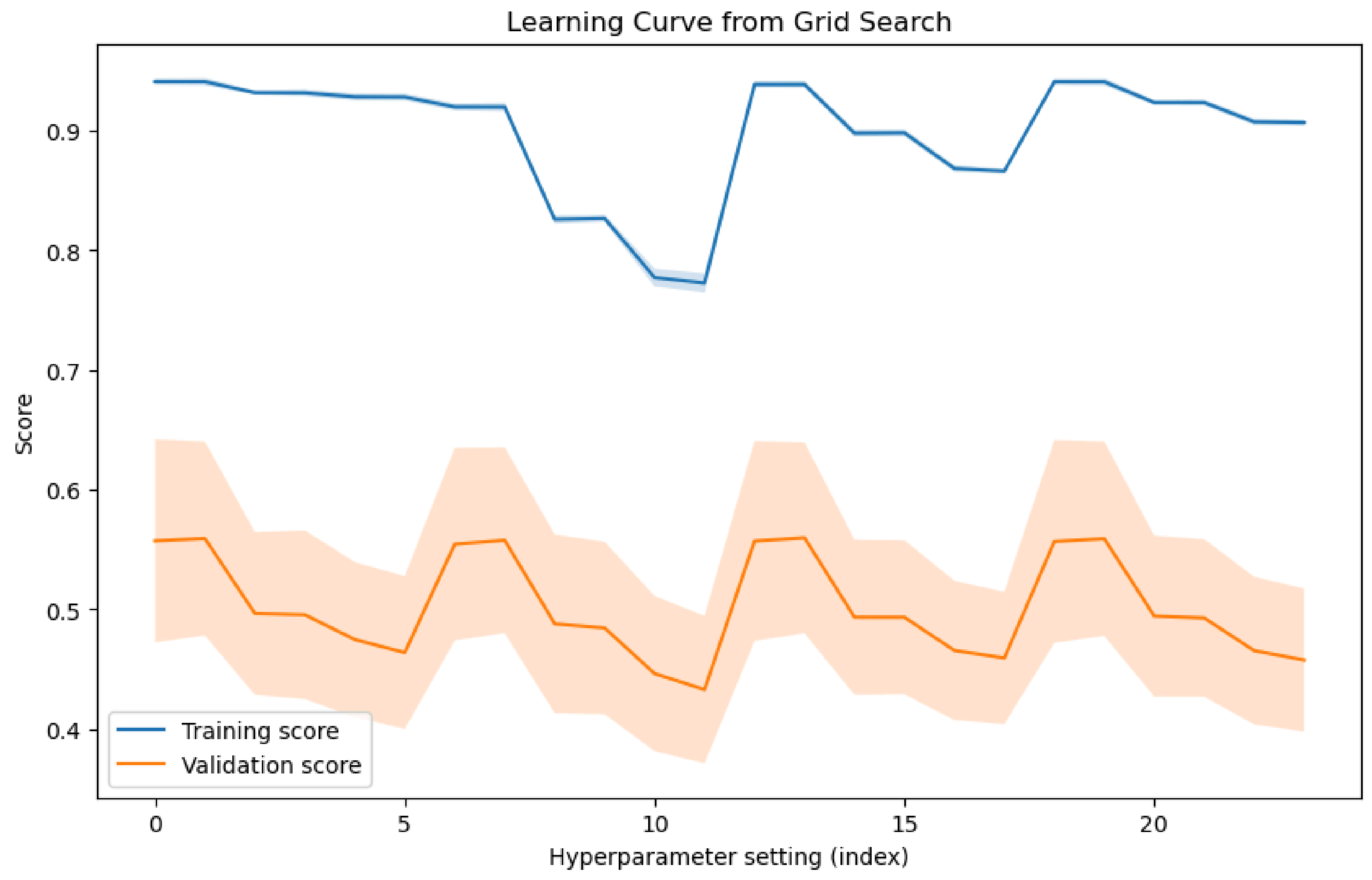
The way that linear regression works is it tries to give each feature a weight and based on that weight, it will use the linear combination of features (and weights) to try and predict the output.

Unfortunately, it was very quickly evident that the linear regression model is too weak of a model as it failed to capture any sort of meaningful variance and thus had a high amount of bias. This meant that the output that we got from the model was pretty much random garbage and that it was time to move on to a stronger model.

Random Forest

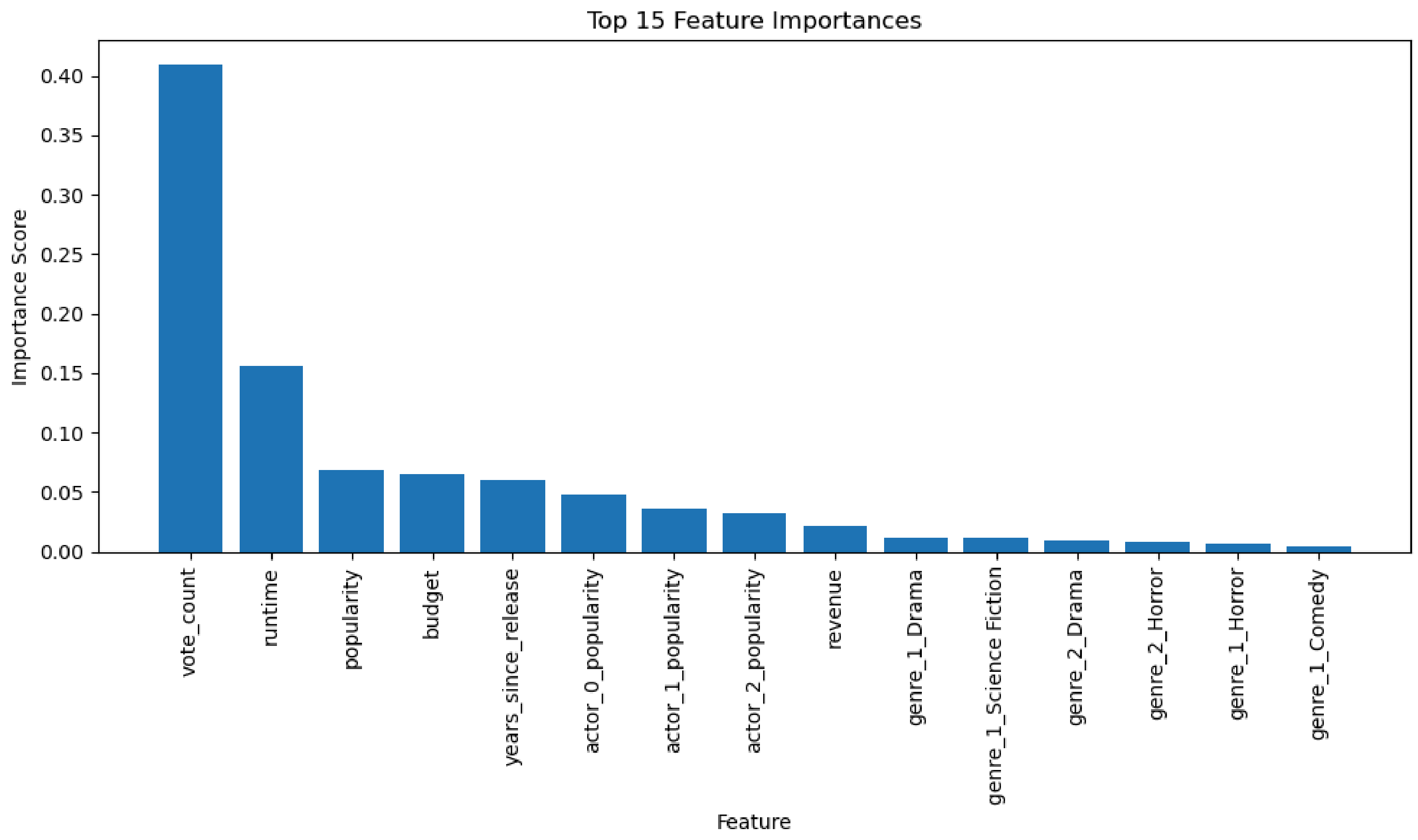
Visualizations

Learning Curve



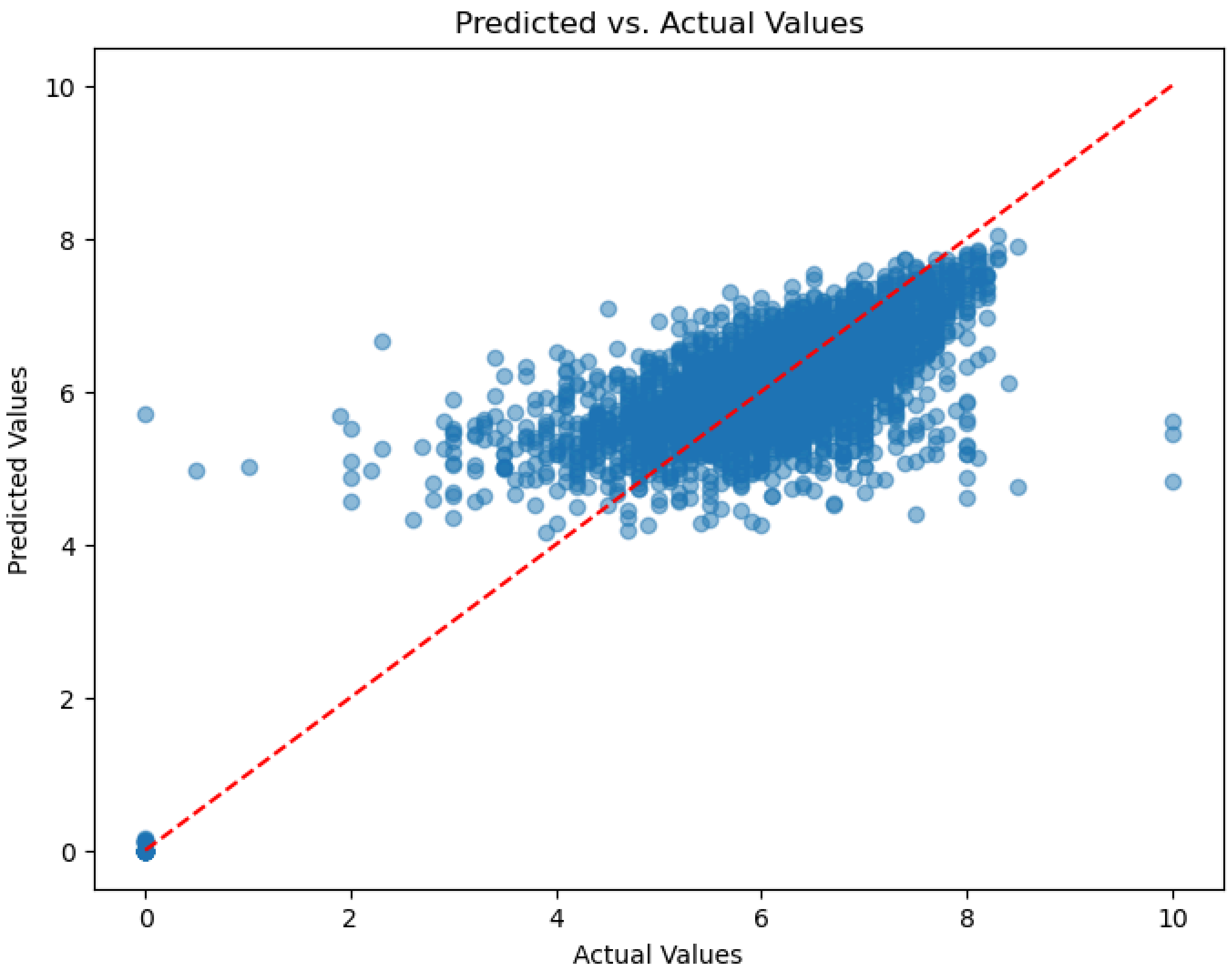
The learning curve shows a high training score (near 1.0), indicating that the model fits the training data very well, but the significantly lower validation score (around 0.5) suggests overfitting, as the model struggles to generalize to unseen data. The gap between training and validation scores, along with fluctuations in validation performance across different hyperparameter settings, reinforces this overfitting tendency. The wide error band for the validation score indicates variability, suggesting instability in model performance across different data splits. To improve generalization, steps such as applying regularization, reducing model complexity, or adding more representative data could be beneficial.

Top 15 Features



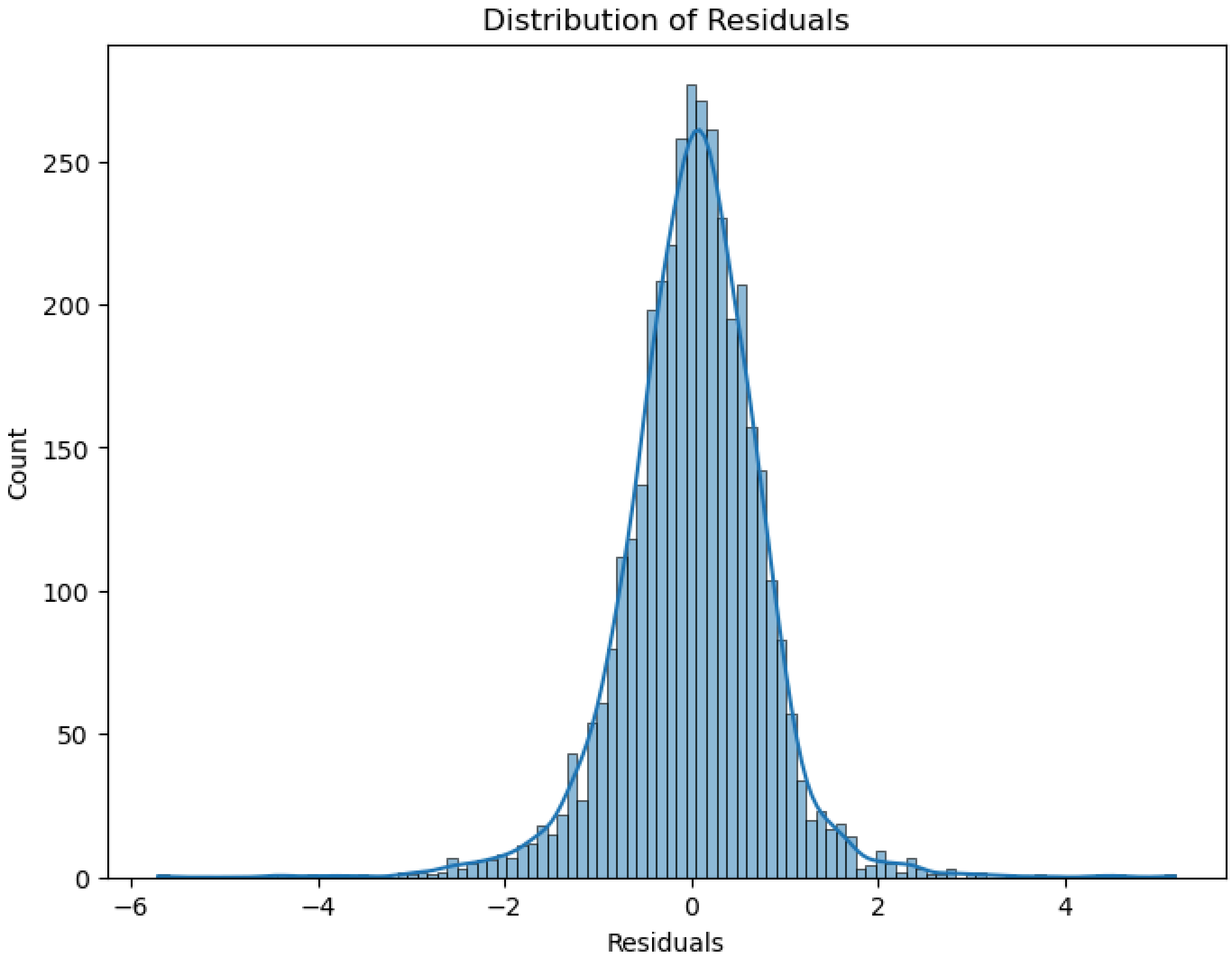
The feature importance plot highlights the top 15 features contributing to the model's predictions, with `vote_count` standing out as the most influential feature by a large margin, followed by `runtime`, `popularity`, and `budget`. This suggests that the number of votes a movie receives significantly impacts the model's predictions, likely because vote count serves as a proxy for viewer engagement or popularity, which often correlates with higher ratings. Secondary features, such as `runtime` and `popularity`, also play substantial roles, indicating that factors related to a movie's appeal and production scale are relevant. Genre-related features like `genre_1_Drama` and `genre_1_Science Fiction` appear further down the list, contributing modestly to the prediction. Overall, the plot underscores the importance of viewer engagement metrics and runtime in predicting ratings, while genre and other minor features have less influence.

Predicted vs Actual



The **Predicted vs. Actual Values** plot illustrates the model's predictive accuracy, with most points clustering around the red dashed line, indicating that the model generally performs well in predicting ratings between 4 and 8. However, there is a noticeable spread, especially at the lower and higher ends of the scale, where the model shows some deviations from the actual values, leading to both underestimations and overestimations. This spread suggests that while the model captures the central trend reasonably well, it struggles with extreme ratings, possibly due to unmodeled complexities. Overall, the model demonstrates good predictive accuracy, though these deviations indicate that further tuning or additional features could help improve its ability to handle outliers and more accurately capture the full range of ratings.

Distribution of Residuals



The **Distribution of Residuals** plot shows that the residuals (differences between predicted and actual values) are approximately normally distributed around zero, with most residuals clustering closely around the center. This symmetric, bell-shaped distribution suggests that the model's predictions are generally unbiased, as it neither consistently overestimates nor underestimates the actual values. The narrow spread indicates that most predictions are close to the actual values, reflecting good accuracy. However, there are some outliers on both sides, which represent occasional larger prediction errors. Overall, this residual distribution implies that the model performs well for most data points, though there are a few instances where predictions deviate more substantially from the actual values.

Quantitative Metrics

Optimal Hyperparameters:

The best parameters identified—max_depth=16, max_features=1.0, min_samples_leaf=1, min_samples_split=2, and n_estimators=150—suggest that a moderately deep forest with a sufficient number of trees was necessary for capturing patterns in this dataset. By using all features (max_features=1.0) at each split, the model leveraged all available information, balancing complexity with accuracy.

Out-of-Bag Score (OOB):

With an Out-of-Bag (OOB) Score of 0.58, the model demonstrates moderate generalization capabilities. This OOB score, obtained from out-of-sample data during training, indicates that about 58% of the variance in movie ratings is explained by the model, which aligns well with the R-squared value and reinforces the consistency of its performance.

Mean Squared Error (MSE):

The MSE of 0.56 shows the average squared difference between the predicted and actual ratings. While a lower MSE would indicate higher precision, the current value suggests that some variability in the predictions exists. This might be due to complex or nonlinear relationships within the data that were not fully captured by the current features. In practical terms, this suggests the model sometimes deviates noticeably from the true ratings, hinting that certain movies might be harder for the model to predict accurately. This could result from complex relationships that the current model and features aren’t fully capturing.

R-squared (R^2):

R-squared (R^2) of 0.56 suggests that the model captures about 56% of the variability in movie ratings (vote_average). While this indicates the model does have predictive power, the R^2 value also tells us there’s room for improvement, as nearly half of the variance remains unexplained. This could mean that other relevant factors—possibly nuanced aspects of movies not captured by the current features (such as director or script quality)might play a role in ratings.

Mean Absolute Error (MAE):

The Mean Absolute Error (MAE) of 0.54 provides an intuitive measure of the model’s prediction accuracy by representing the average absolute difference between predicted and actual movie ratings. In practical terms, this means that if a movie has an actual rating of 7.0, the model’s prediction will, on average, fall within the range of 6.46 to 7.54. This close range implies that the model’s predictions are relatively reliable and that it performs well in approximating ratings, despite inherent complexities in the data.

Model Analysis

Our first model is a random forest model. It is a learning technique that combines multiple decision trees to try and improve predictive performance and reduce overfitting. Each independent decision tree in the forest makes its own prediction, and the random forest will take the average (in regression) or majority vote (in classification) of the predictions. This results in a more accurate model.

The random forest algorithm takes in a random subset of the data to build each tree (bootstrap aggregating). Additionally, it will select a random subset of features at each split within each tree, which will ensure that no single feature will dominate in the decision-making process. This randomness helps to reduce variance in predictions making the model more robust to overfitting.

In terms of performance, our random forest model demonstrated reasonable predictive accuracy with the chosen hyperparameters. Key metrics indicate that it captures patterns within the data effectively, especially for the primary target variable, vote_average (movie ratings). Although the model performed well overall, with consistent accuracy and unbiased predictions, there were some deviations and outliers in certain cases, suggesting that further tuning or feature enhancement could improve results further.

Significance of Key Features

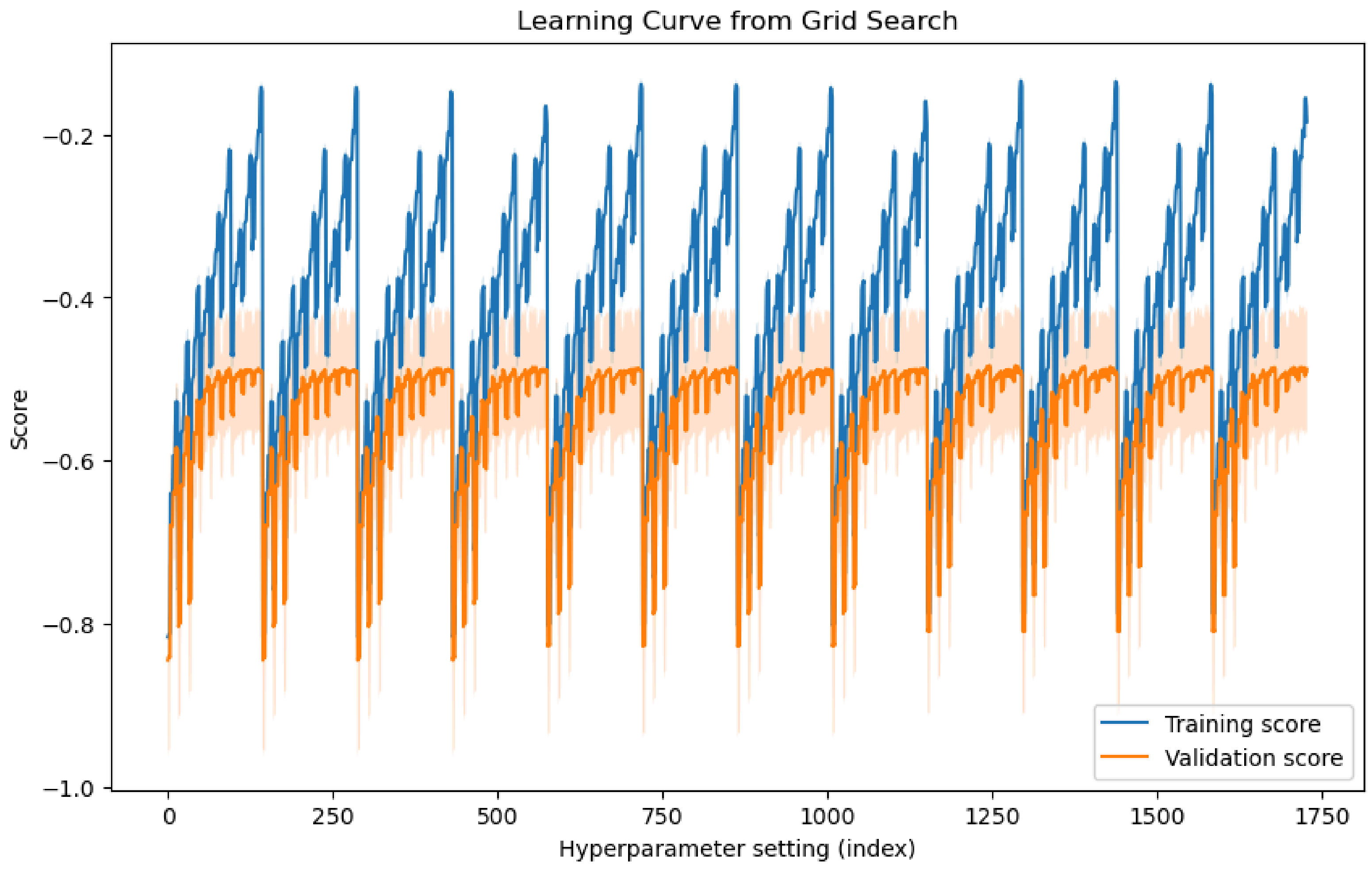
The model's feature importance rankings indicate that variables such as budget, popularity, and revenue are crucial in predicting movie ratings. This aligns well with real-world expectations, where higher-budget and popular movies often receive more attention and, potentially, better ratings. This finding supports the idea that movie-related financial and popularity metrics are solid predictors of audience perception.

Less important features, such as specific genres or languages, might indicate that these factors have a more subtle influence on ratings or that their effects are already absorbed by more general variables like budget and popularity.

Extreme Gradient Boosting:

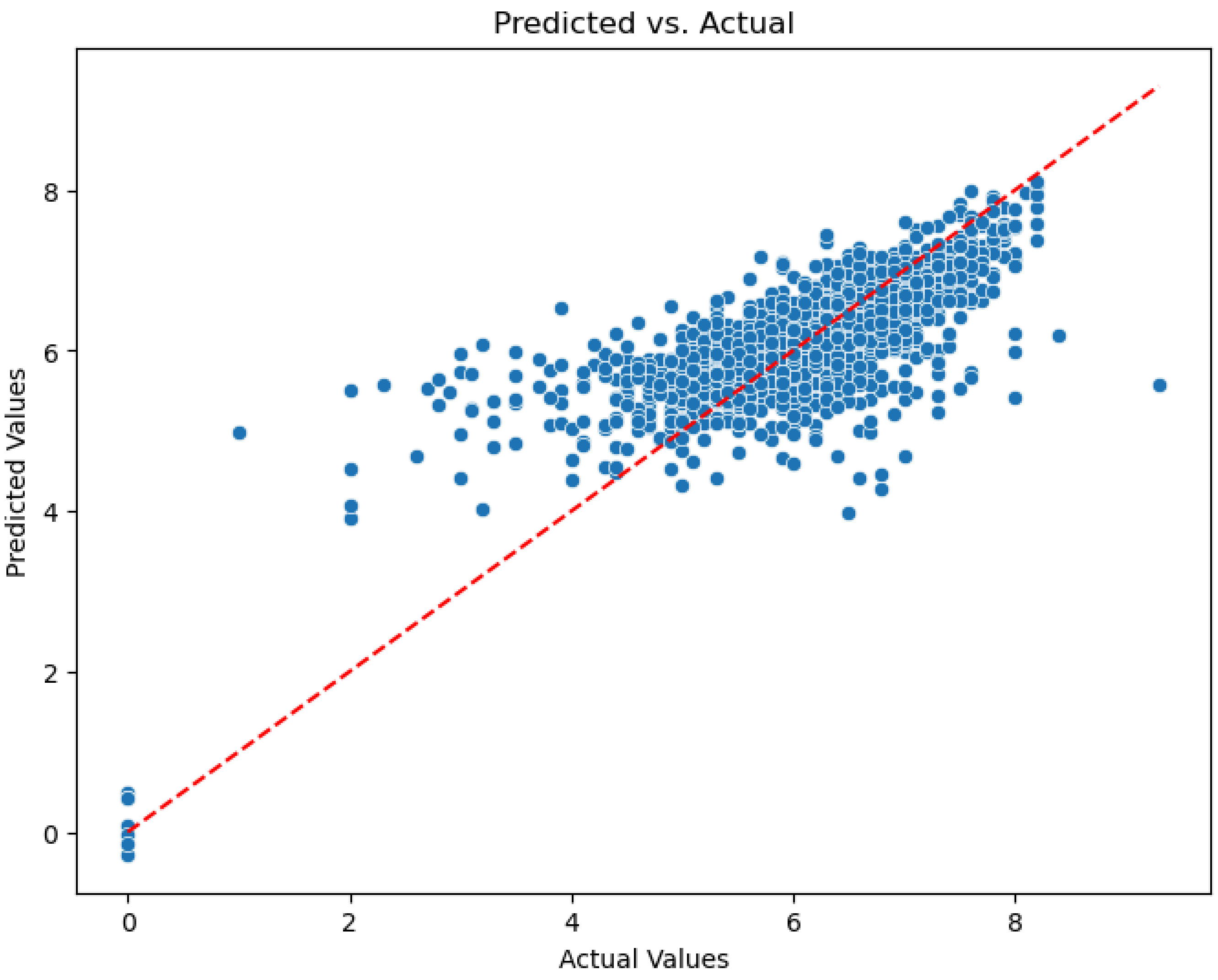
Visualizations:

Learning Curve:



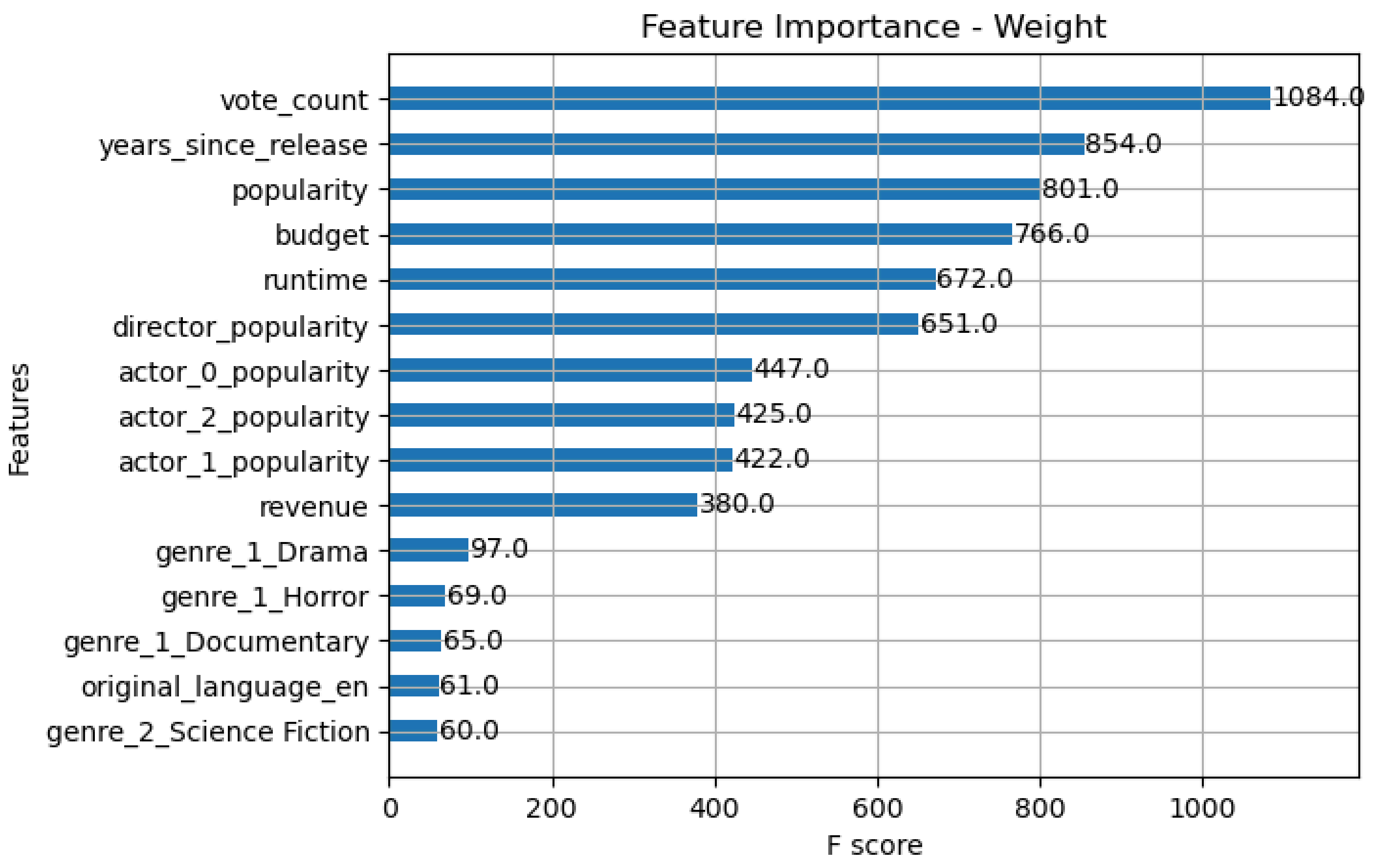
The cyclical pattern indicates that specific hyperparameters are very important, and you see the large drop everytime These parameters are set to specific values. Unfortunately, there is a massive differnce between the validation and training scores which indicates a high degree of overfitting. This ultimately means that alternative strategies will need to be considered such as regularization, increasing the amount of data by perhaps bagging, or perhaps tuning hyperparameters to increase bias and therefore decrease overfitting.

Predicted vs. Actual



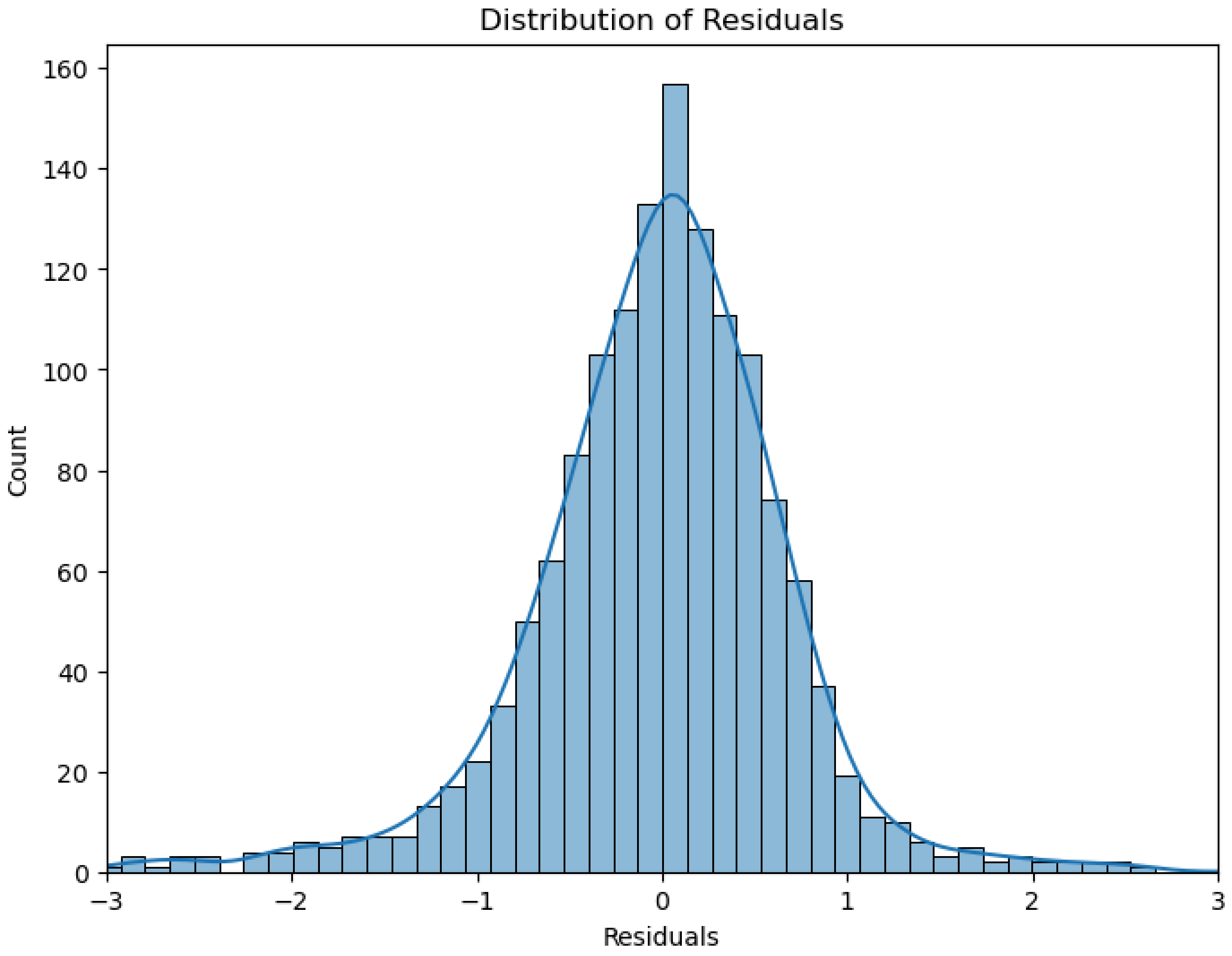
This plot shows the graph of actual vs predicted values, with the red line corresponding to accurate predictions. Values near the means are generally predicted well, though the model struggles with values at lower scores. With large outliers, the model performs poorly which is what leads our MSE to being larger than MAE.

Top 15 Features:



Similar to random forest, we see vote count as an important feature. This makes sense, as if a movie is very good (or very bad) they are going to be more likely to review the movie. Additionally things like years_since release is going to impact highly along with popularity (TMDB metric based off votes for the day, views for the day, marked as favourite that day, watchlist for the day, total votes, release date, and the previous day's score) are going to be highly influential for our model. We find genres of drama, comedy, and horror to be the most influential in our model

Distribution of Residuals



The **Distribution of Residuals** plot shows that the residuals (differences between predicted and actual values) are approximately normally distributed around zero, with most residuals clustering closely around the center. This symmetric, bell-shaped distribution suggests that the model's predictions are generally unbiased, as it neither consistently overestimates nor underestimates the actual values. The narrow spread indicates that most predictions are close to the actual values, reflecting good accuracy. However, there are some outliers on both sides, which represent occasional larger prediction errors. Overall, this residual distribution implies that the model performs well for most data points, though there are a few instances where predictions deviate more substantially from the actual values.

Quantitative Metrics Discussion

Optimal Hyperparameters

The optimal hyperparameters are n_estimators = 350, colsample_bytree = .8, gamma = 0, learning_rate = .03, max_depth = 5, and subsample = .8. N_estimators represents the number of trees in the model, learning_rate controls how much each tree contributes to to the final prediction, max_depth determines the maximum depth of each tree, subsample controls the percentage of rows used for each tree, colsample_bytree controls the percentage of columns considered for each tree, and gamma is the minimum loss reduction required to make a furhter partition in a tree.

MAE

MAE of .500 indicates that our predictions are off by .498 on average.

MSE

MSE of .494 indicates that the square of our prediction error is .512 on average. In this model, MSE and MAE are almost identical, meaning we do not have many large outliers, and the residuals are generally small.

R-squared

R-squared of .565 means that 56.5% of variance in the target is explained by the features of our model, and the remaining 43.5% of variance is unexplained. This means that additional features may be beneficial to improving the model.

Model Analysis

The gradient boosting model did much better than the random forests. We believe that this is because of the iterative error minimization process as well as the built in regularization.

XGboost does a lot of things, so it's a bit difficult to talk about everything that it does. But essentially it's a stronger version of gradient boosting which adds things such as regularization, tree pruning, parallel processing, and handling missing values. [4]

While this model did do better than the random forest model, it is still important to note that there is still a sizable MAE and MSE. This means that there is still more than should be done to improve the model that go outside the scope of simply picking a better model. These things could include: more data preprocessing, dimensionality reduction, or even getting more data.

Model Comparison

While linear regression is simple and quick to compute, it also performed the worst, with the highest MAE and MSE, and the lowest R-Squared. We believe that this is due to its inability to capture nonlinear relationships. This was an indication that it was time to move on to stronger and more nonlinear models.

The random forest model was the second best, with lower MAE and MSE than linear regression but still prone to overfitting. This model did much better than linear regression as it was able to capture nonlinear relationships, however, we noticed a high degree of overfitting as well as a longer computation time. This was a good sign that more hyperparameter tuning was going to be needed as to reduce the amount of overfitting.

The XGBoost model was the best model, achieving the lowest MAE and MSE, and the highest R-Squared. This success is likely due to its use of iterative error minimization and built-in L1 and L2 regularization [4], which reduce overfitting.

Overall, we learned a lot from each visualization as we got to quickly see that linear regression was too weak of a model and that random forests were massively overfitting.

Next Steps

We plan to improve our by enhancing the preprocessing and also exploring additional machine learning algorithms. First we will perform further feature engineering through our current dataset and the TMDB api, which will include features such as director and producer popularity score. Then, we will perform principal componenet analysis to reduce the dimensionality of the dataset.

If we were to continue improving the model, we would focus on the following strategies to enhance its performance and robustness:

Data Enrichment: We plan to expand the dataset by performing additional feature engineering using the TMDB API. New features, such as director and producer popularity scores, could capture underlying patterns not currently represented in the data.

Dimensionality Reduction: We will apply Principal Component Analysis (PCA) to reduce the dimensionality of the dataset. This approach will help simplify the model, potentially reducing overfitting and improving computational efficiency.

Model Exploration: In addition to our current model, we aim to test alternative algorithms:

XGBoost: A powerful gradient boosting technique that can better address residual errors and potentially outperform our current random forest model. Linear Regression: A simpler baseline model to gauge the performance improvement from more complex algorithms. Hyperparameter Tuning: We will refine the model by experimenting with a wider range of hyperparameter values. Techniques such as grid search and Bayesian optimization will be employed to identify the best configurations.

Cross-Validation: To ensure the model’s robustness, we will use higher cross-validation (CV) fold counts. This will provide a more reliable assessment of its generalization capabilities and minimize the risk of overfitting.

By incorporating these steps, we aim to maximize the model’s accuracy, robustness, and interpretability, delivering a well-rounded solution that balances performance and complexity.

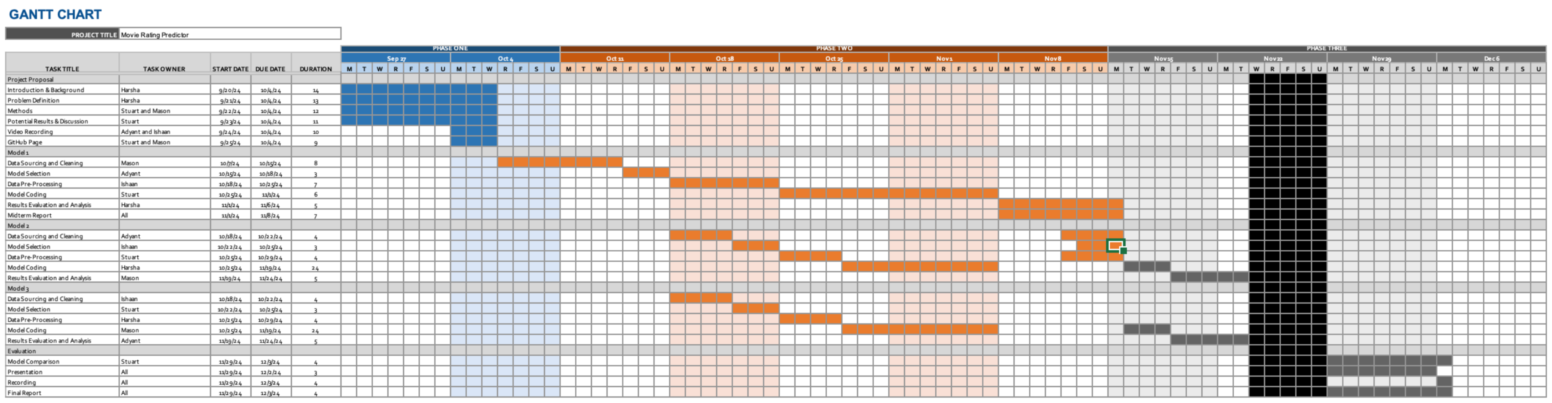
Project Goals:

The primary goal of this project is to build a predictive model that forecasts the average user rating of a movie based cast, crew, and genre.
Regarding sustainability, this pulls from the TMDB5000 database which is a reliable source. Regarding ethical concerns, this tool is only meant to help movie planners before they commit large amounts of resources to a project.

Expected Results:

We expect particular critically acclaimed actors and directors to have a strong positive correlation with user ratings. Additionally, genre and other crew details (director, etc) will also play roles in determining the ratings.

Gantt Chart



GANTT CHART

PROJECT TITLE Movie Rating Predictor

TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION
Project Proposal				
Introduction & Background	Harsha	9/20/24	10/4/24	14
Problem Definition	Harsha	9/21/24	10/4/24	13
Methods	Stuart and Mason	9/22/24	10/4/24	12
Potential Results & Discussion	Stuart	9/23/24	10/4/24	11
Video Recording	Adyant and Ishaan	9/24/24	10/4/24	10
GitHub Page	Stuart and Mason	9/25/24	10/4/24	9
Model 1				
Data Sourcing and Cleaning	Mason	10/7/24	10/15/24	8
Model Selection	Adyant	10/15/24	10/18/24	3
Data Pre-Processing	Ishaan	10/18/24	10/25/24	7
Model Coding	Stuart	10/25/24	11/1/24	6
Results Evaluation and Analysis	Harsha	11/1/24	11/6/24	5
Midterm Report	All	11/1/24	11/8/24	7
Model 2				
Data Sourcing and Cleaning	Adyant	10/18/24	10/22/24	4
Model Selection	Ishaan	10/22/24	10/25/24	3
Data Pre-Processing	Stuart	10/25/24	10/29/24	4
Model Coding	Harsha	10/25/24	11/19/24	24
Results Evaluation and Analysis	Mason	11/19/24	11/24/24	5
Model 3				
Data Sourcing and Cleaning	Ishaan	10/18/24	10/22/24	4
Model Selection	Stuart	10/22/24	10/25/24	3
Data Pre-Processing	Harsha	10/25/24	10/29/24	4
Model Coding	Mason	10/25/24	11/19/24	24
Results Evaluation and Analysis	Adyant	11/19/24	11/24/24	5
Evaluation				
Model Comparison	Stuart	11/29/24	12/3/24	4
Presentation	All	11/29/24	12/2/24	3
Recording	All	11/29/24	12/3/24	4
Final Report	All	11/29/24	12/3/24	4

You may also see the full version here at Github

Contribution Table

</table> ## References: [1] R. Dhir and A. Raj, "Movie Success Prediction using Machine Learning Algorithms and their Comparison," 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC), Jalandhar, India, 2018, pp. 385-390, doi: 10.1109/ICSCCC.2018.8703320.
[2] T. Masui, "All You Need to Know about Gradient Boosting Algorithm - Part 1. Regression," Medium, Feb. 12, 2022. https://towardsdatascience.com/all-you-need-to-know-about-gradient-boosting-algorithm-part-1-regression-2520a34a502
[3] D. Maulud and A. M. Abdulazeez, "A Review on Linear Regression Comprehensive in Machine Learning," Journal of Applied Science and Technology Trends, vol. 1, no. 4, pp. 140–147, Dec. 2020, doi: 10.38094/jastt1457.
[4] J. Crossman-Smith, "XGBoost Explained: A Beginner's Guide - Low Code for Data Science - Medium," Medium, Mar. 24, 2024. https://medium.com/low-code-for-advanced-data-science/xgboost-explained-a-beginners-guide-095464ad418f

Name	Contact	
Stuart Kernohan	Data cleaning, feature engineering, and visualizations, Helped write the report, host the website, and plan the team meetings. XG Boost.	</tr>
Mason Ho	Made initial model with random forest regression pulling from Stuart's dataset, Transcribed the report in markdown. Random Forest	
Ishaan	Wrote Data Report and Analysis on Model, Gannt Chart, Slide Deck, Youtube Video. Regression.	
Harsha Vegiraju	Wrote Data Report + Analysis on results, Wrote the literature review, wrote motivation, did research for the report, updated Gantt chart.	

Adyant Mishra	Wrote Data Report + Analysis on results, Gannt Chart, Slide Deck, Youtube Video. Regression