Project Report | Movie Rating Prediction

1. Introduction

This project aims to develop a machine learning model that can predict IMDb movie ratings based on various attributes, exploring the strengths and limitations of different algorithms in tackling this classification task. By analyzing the performance of Support Vector Machines, Random Forest, Gradient Boosting, AdaBoost, and Ensemble models, I seek to identify the most effective approach for movie rating prediction.

2. Methodology

2.1. Data Pre-processing

The initial dataset comprised of various features. I prepared the data for model training by:

- **Data Cleaning:** String columns were removed as various models cannot interpret that.

- **Handling Missing Values:** Missing values in the dataset were imputed using the median values of the respective columns. This ensured that the dataset remained complete without introducing significant bias.

- **Normalization:** To ensure that all features contributed equally to the model training process, numerical features were normalized using MinMaxScaler. This scaled the features to a range between 0 and 1, which helps in improving the performance of various machine learning algorithms.

2.2. Model Selection and Training

I explored various machine learning models to predict IMDB movie ratings:

- Support Vector Machine (SVM): Effective in high-dimensional spaces and for cases where dimensions exceed samples.

- Random Forest Classifier: Constructs multiple decision trees and outputs the mode for classification, known for high accuracy and handling large, high-dimensional datasets.

- Gradient Boosting Classifier: Builds an additive model in stages, optimizing differentiable loss functions, renowned for high predictive accuracy.

- AdaBoost Classifier: Combines multiple weak classifiers, assigning weights to instances and adapting to difficult ones over iterations.

- Voting Classifier: Combines predictions from multiple models (RF, Gradient Boosting, SVM) based on majority voting, leveraging individual model strengths to enhance overall accuracy.

2.3. Evaluation

The performance of each model was evaluated using accuracy as the primary metric. Cross-Validation was employed to ensure the robustness of the models. The dataset was split into training and validation sets, with 70% used for training and 30% for validation.

3. Result

The performance of the classifiers was evaluated using accuracy as the primary metric.

Model	Train Accuracy	Validate Accuracy	Overall Accuracy
SVM	0.690771	0.691796	0.691079
Random Forest	0.997146	0.710643	0.911119
Gradient Boosting	0.813035	0.721729	0.785619
AdaBoost Classifier	1.000000	0.696231	0.908788
Ensemble Classifier	0.895814	0.711752	0.840546

Table 1 - Classifier Performance Comparison

- Train Accuracy: The accuracy of the model on the training set. This metric indicates how well the model learned from the training data.

- Validate Accuracy: The accuracy of the model on the validation set. This metric provides an indication of the model's performance on unseen data during training.

- Overall Accuracy: The accuracy of the model on the entire dataset. This metric shows the overall performance of the model across all data points.

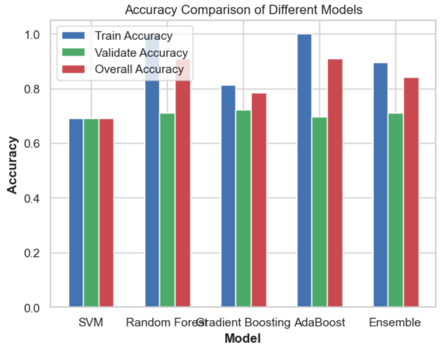

Figure 1 - Accuracy Comparison of Different Models

Model	Mean Accuracy	Standard Deviation
SVM	0.6703	0.0222
Random Forest	0.6822	0.0221
Gradient Boosting	0.6960	0.0268
AdaBoost Classifier	0.6827	0.0225
Ensemble Classifier	0.6808	0.0249

Table 2 – Cross-Validation Result Comparison

4. Discussion and Analysis

Imbalanced Distribution of IMDb Scores

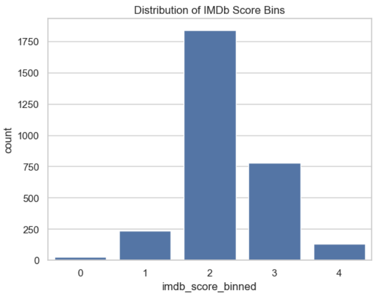

Figure 2 - Distribution of IMDb Score

The histogram of IMDb scores shows an imbalanced distribution, with most movies clustered in ratings 2 and 3. This imbalance can bias model training towards majority classes. To address this, I applied class weight balancing to the RF and SVM models. The RF model improved accuracy by 1.1%, likely due to its ability to focus on minority classes through its ensemble nature. However, the SVM model's accuracy dropped by 28.94% because class weight adjustments disrupted the hyperplane, leading to poorer performance on imbalanced data.

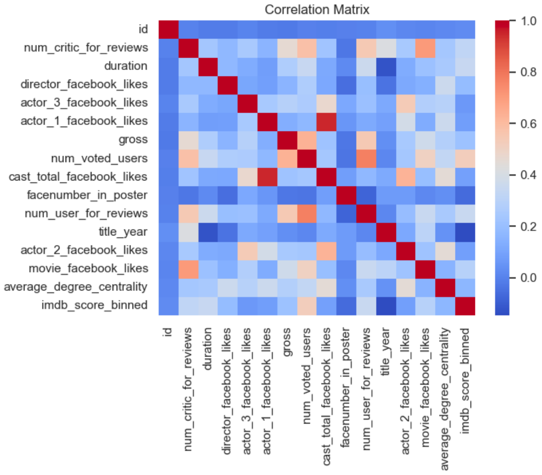

Figure 3 – Correlation Matrix of Features

The correlation matrix shows strong relationships, particularly between cast_total_facebook_likes and actor_1_facebook_likes, and title_year and duration, indicating potential multicollinearity. While SVM models suffer from multicollinearity, tree-based models like RF, Gradient Boosting, AdaBoost, and Ensemble models are less affected. Removing multicollinear variables in SVM models resulted in negligible performance difference (<1%).

String Features Extraction

In an attempt to enhance model performance, I tried incorporating additional features extracted from .npy files, including actor1_features, actor2_features, director_features, plot_features, and title_features, generated using various methods like count vectorization, Doc2Vec, and FastText. Despite these efforts, the inclusion of these pre-extracted features did not lead to any significant improvement in the models' accuracy or overall performance.

Outliers

I initially examined box plots to identify potential outliers and calculated z-scores for each feature, filtering out rows with z-scores greater than 3.5 for movies with IMDb scores of 2 and 3. Due to the scarcity of other ratings, I focused on these scores. This process removed significant outliers, but it led to overfitting in the RF and AdaBoost models. The reduced variability made the RF model too specialized in the training data, failing to generalize to the validation set. AdaBoost adapted too well to the cleaned data, focusing on hard instances that didn't represent the general population. This experience underscores the importance of balancing data cleaning with preserving enough variability to ensure models can generalize effectively.

4.1. Support Vector Machine (SVM)

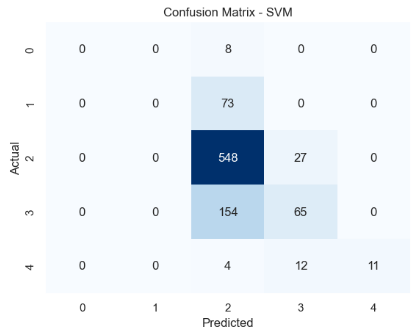

Figure 4 - Confusion Matrix for SVM Classifier

Properties: SVMs find a hyperplane that best separates data into classes, maximizing the margin between the classes. This margin maximization improves generalization on unseen data. SVMs are effective in high-dimensional spaces and robust to overfitting, especially when features outnumber samples. However, they can struggle with imbalanced datasets and are sensitive to the choice of kernel and regularization parameters.

Cross-Validation Results: The SVM model achieved a mean cross-validation accuracy of 67.03% with a standard deviation of 2.22%. The relatively close values between training, validation, and overall accuracies (69.08%, 69.18%, and 69.11%, respectively) indicate that the SVM model did not overfit the training data, as expected.

Error Analysis: The SVM model achieves high accuracy for the most common class (rating 2), correctly classifying 548 instances. However, it struggles with misclassifications, notably for ratings 3 and 4. This aligns with SVM's sensitivity to imbalanced datasets. Introducing class weights led to a significant 28.94% drop in overall accuracy, highlighting SVM's sensitivity to such adjustments, as it relies on finding an optimal hyperplane that maximizes the margin between classes, disrupted by adjusted class weights, especially with highly imbalanced data.

4.2. Random Forest

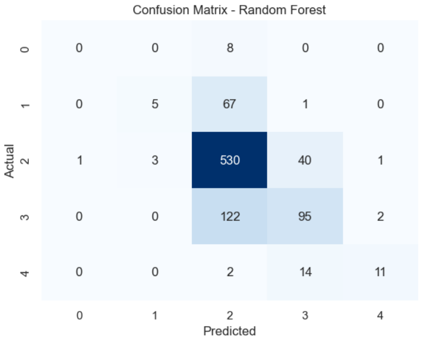

Figure 5 - Confusion Matrix for RF Classifier

Properties: Random Forests create multiple decision trees from random data subsets, averaging predictions (regression) or taking the majority vote (classification). This reduces variance and helps avoid overfitting. They handle high-dimensional data, are less affected by multicollinearity, and are robust to outliers and noise. However, they can be computationally expensive and may overfit if trees are too deep.

Cross-Validation Results: The RF model achieved a mean cross-validation accuracy of 68.22% with a standard deviation of 2.21%. The significant difference between training accuracy (99.71%) and validation accuracy (71.06%) indicates overfitting.

Error Analysis: The RF model excels at identifying the majority class (rating 2), with 530 correct predictions. However, it struggles with misclassifications for ratings 3 and 4. This indicates a bias towards the majority class and difficulty handling less frequent classes. While RF are powerful and handle high-dimensional data well, these misclassifications suggest the model may be overfitting the training data and not generalizing well to unseen data without proper tuning. The tendency to overfit can be attributed to its ability to learn complex patterns in the training data, which may not represent the underlying distribution of the test data.

4.3. Gradient Boosting

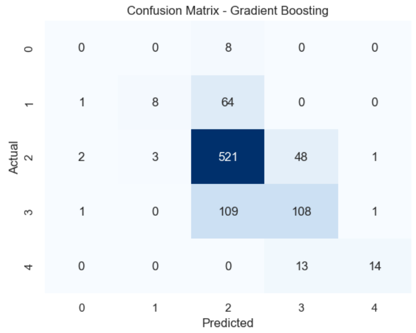

Figure 6 - Confusion Matrix for Gradient Boosting

Properties: Gradient Boosting builds models sequentially, correcting errors from previous ones and optimizing a loss function by adding weak learners (decision trees) in stages. It has high predictive accuracy and handles various data types, capturing complex patterns. However, it can overfit with too many stages, is computationally intensive, and requires careful hyperparameter tuning.

Cross-Validation Results: The Gradient Boosting model achieved a mean cross-validation accuracy of 69.60% with a standard deviation of 2.68%. The moderate difference between training accuracy (81.30%) and validation accuracy (72.17%) suggests some overfitting but still generalizes well to the validation set.

Error Analysis: Figure 6 shows a more balanced performance across classes compared to RF. It correctly classifies 521 instances of rating 2 but still exhibits notable misclassifications, especially between ratings 2 and 3. This indicates that while Gradient Boosting improves generalization by sequentially correcting errors, it struggles with distinguishing similar classes. This difficulty likely stems from the complexity of the task and the subtle differences between adjacent ratings, challenging even advanced algorithms like Gradient Boosting.

4.4. AdaBoost Classifier

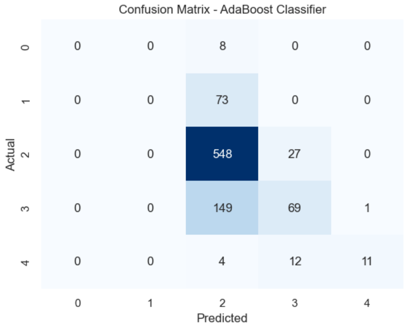

Figure 7 - Confusion Matrix for AdaBoost Classifier

Properties: AdaBoost combines multiple weak classifiers, typically decision stumps, to create a strong classifier. It assigns higher weights to misclassified instances, focusing on difficult cases. It boosts weak learners' performance and adapts over iterations but is prone to overfitting if the base classifiers are too complex and is sensitive to noisy data and outliers.

Cross-Validation Results: The AdaBoost model achieved a mean cross-validation accuracy of 68.27% with a standard deviation of 2.25%. The perfect training accuracy (100.00%) and lower validation accuracy (69.62%) indicate significant overfitting.

Error Analysis: The confusion matrix for the AdaBoost Classifier shows high accuracy for the majority class (rating 2), with 548 correct predictions. However, it misclassifies rating 3 as 2 and rating 4 as 3. This indicates overfitting, as the model focuses too much on hard instances from the training set, failing to generalize well to unseen data. AdaBoost's close adaptation to the training data results in excellent performance on familiar instances but poor generalization, highlighting the challenge of balancing training fit and flexibility for new data.

4.5. Ensemble Classifier

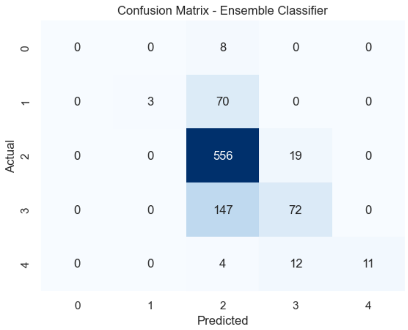

Figure 8 - Confusion Matrix for Voting Classifier

Properties: Ensemble methods like voting classifiers combine multiple models' strengths, reducing bias, variance, and overfitting. They improve performance by leveraging model diversity for robust predictions, enhancing generalization. However, they are computationally intensive and complex to implement, with performance depending on the diversity and accuracy of individual models.

Cross-Validation Results: The Ensemble model achieved a mean cross-validation accuracy of 69.60% with a standard deviation of 2.12%. The closer training accuracy (89.58%) and validation accuracy (71.18%) indicate better generalization compared to individual models.

Error Analysis: The confusion matrix for the Ensemble model demonstrates balanced performance across all classes. It correctly classifies 556 instances of rating 2. However, misclassifications remain, such as rating 3 as 2 and rating 4 as 3. Despite that, the ensemble approach reduces the bias and variance of individual models, leveraging their strengths for more accurate predictions. This balanced performance underscores the effectiveness of combining different classifiers, enhancing overall accuracy and robustness.

Addressing Overfitting

Overfitting is a common issue where a model performs exceptionally well on training data but fails to generalize to unseen data. In my efforts to mitigate overfitting, I employed various techniques, such as balancing class weights and carefully tuning hyperparameters. For instance, in the RF Classifier, I used class weights to handle imbalances and set parameters like n_estimators and max_depth to prevent the trees from becoming too specialized. Despite these efforts, some models still exhibited signs of overfitting, as evidenced by the high training accuracy but lower validation accuracy, particularly in the RF and AdaBoost classifiers.

5. Conclusion

In this project, I explored the application of machine learning techniques to predict IMDb movie ratings. Our comprehensive analysis and comparison of five different models - SVM, Random Forest, Gradient Boosting, AdaBoost, and Ensemble - revealed valuable insights into their strengths and limitations. While each model demonstrated varying degrees of accuracy, the Ensemble model emerged as the top performer, leveraging the diversity of individual models to achieve a balanced and robust prediction.

Our findings highlight the importance of addressing imbalanced datasets, feature correlation, and overfitting in movie rating prediction. The results also underscore the value of ensemble methods in combining the strengths of individual models to achieve improved generalization and accuracy.

6. References