Audience Rating PREDICTION

By Elamathi. R

TABLE OF CONTENTS

Problem STATEMENT

- With the given dataset, build a model to predict '**audience_rating**'. Demonstrate the working of the pipeline with a notebook, also validate the model for its accuracy.

project OBJECTIVE

- The objective of this project is to predict audience ratings for movies using machine learning models, based on various features like cast, genre, and directors.

- The goal is to identify the most accurate model to understand how these factors influence audience sentiment and improve decision-making in the entertainment industry.

- Random Forest Regressor
- Gradient Boosting Regressor
- AdaBoost Regressor
- Linear Regression
- Ridge Regression
- Decision Tree Regressor
- Lasso Regression
- Support Vector Regressor (SVR)

Machine Learning Models for Audience Rating Prediction

Machine learning, a subset of artificial intelligence, enables systems to learn and improve from data without explicit programming.

In this project, machine learning models are used to predict audience ratings by analyzing historical data and evaluating performance using metrics like R2 score, MSE, and MAE.

CODE EXPLANATION

Data Preprocessing: The dataset is cleaned by handling missing values and transforming categorical features using one-hot encoding and scaling numerical features.

Exploratory Data Analysis (EDA): Basic statistics, correlation analysis, and feature distributions are examined to identify patterns and relationships in the data.

Model Selection: Various machine learning models, including Random Forest, Gradient Boosting, and Linear Regression, are trained to predict audience ratings.

Model Evaluation: Performance metrics like MSE, MAE, and R2 are used to evaluate and compare model effectiveness.

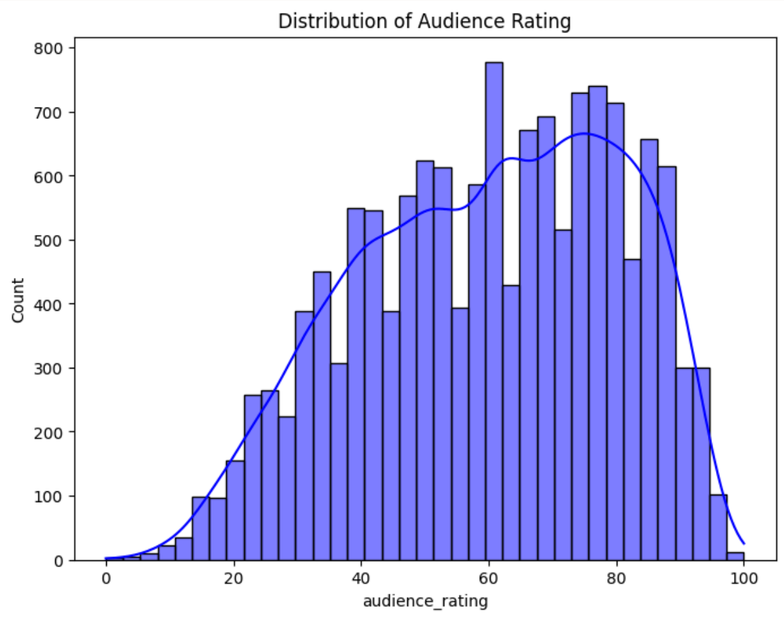
Results: The models' performance is visualized, and the best model is selected based on the highest R2 score.

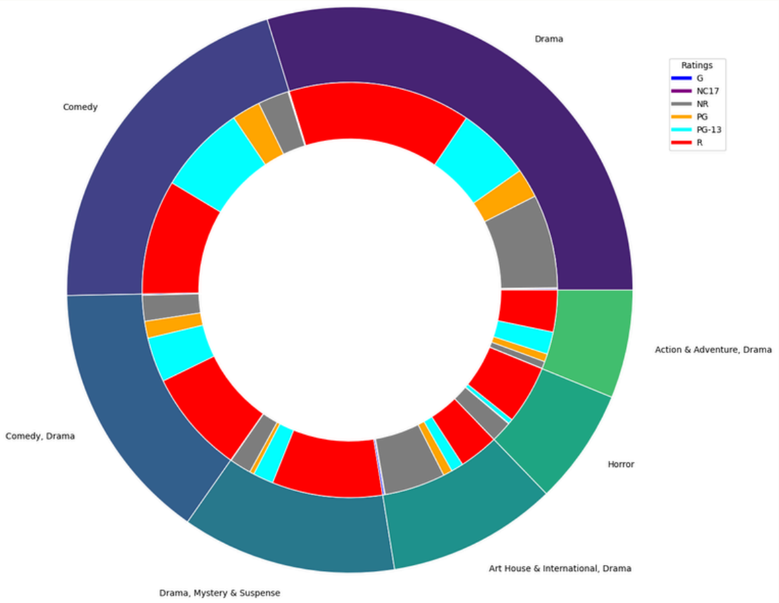
```python
# Calculate median for numerical columns
median_runtime = audience_data['runtime_in_minutes'].median()
median_rating = audience_data['audience_rating'].median()
```

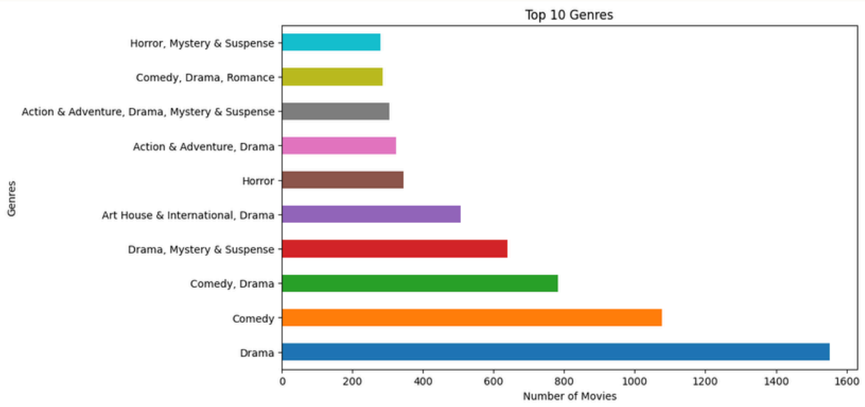
```python
# Convert the 'in_theaters_date' column to datetime format and extract the release year
# This helps in aggregating data by release year
audience_data['release_year'] = pd.to_datetime(audience_data['in_theaters_date'], format='%d-%m-%Y').dt.year

# Creating a histogram to visualize the number of movies released each year
plt.figure(figsize=(10, 6))

# Counting movies for each release year and sorting by year
release_year_counts = audience_data['release_year'].value_counts().sort_index()
release_year_counts.plot(kind='bar', color='skyblue')

# Adding plot title and axis labels
plt.title("Number of Movies by Year of Release")
plt.xlabel("Year")
plt.ylabel("Number of Movies")

# Customizing x-axis labels to show every 5th year for better readability
years = release_year_counts.index  # Get unique years from the data
plt.xticks(
    ticks=range(0, len(years), 5),  # Adjust tick spacing to every 5th year
    labels=[str(year) for year in years[::5]]  # Display labels for every 5th year
)

# Display the plot
plt.show()
```

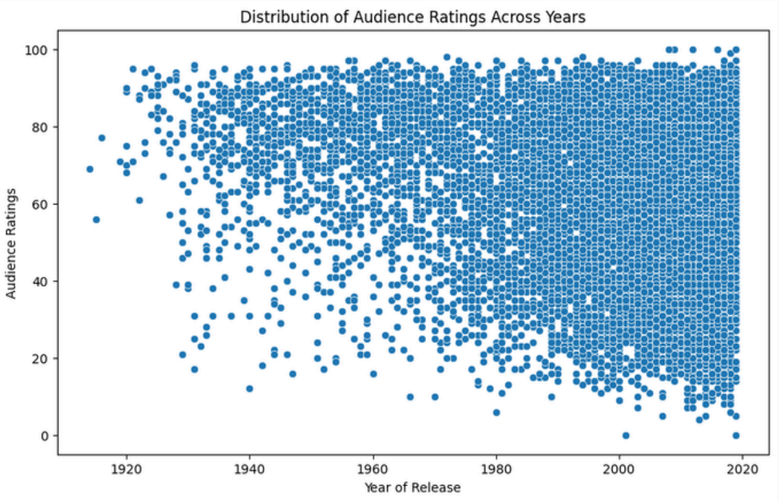
```python
# Dictionary to store different regression models
models = {
    'RandomForest': RandomForestRegressor(random_state=42),
    'GradientBoosting': GradientBoostingRegressor(random_state=42),
    'AdaBoost': AdaBoostRegressor(random_state=42),
    'LinearRegression': LinearRegression(),
    'Ridge': Ridge(),
    'Lasso': Lasso(),
    'DecisionTree': DecisionTreeRegressor(random_state=42),
    'SVR': SVR()
}
```

EDA ANALYSIS OUTPUT









Model BUILDING

Data Split: The dataset is split into training and testing sets using an 80-20 ratio, ensuring that the model is trained on a large portion of the data and tested on unseen data for validation.

Training Pipeline: A pipeline is used to preprocess the data (handling missing values, scaling numerical features, one-hot encoding categorical features) and then train the model.

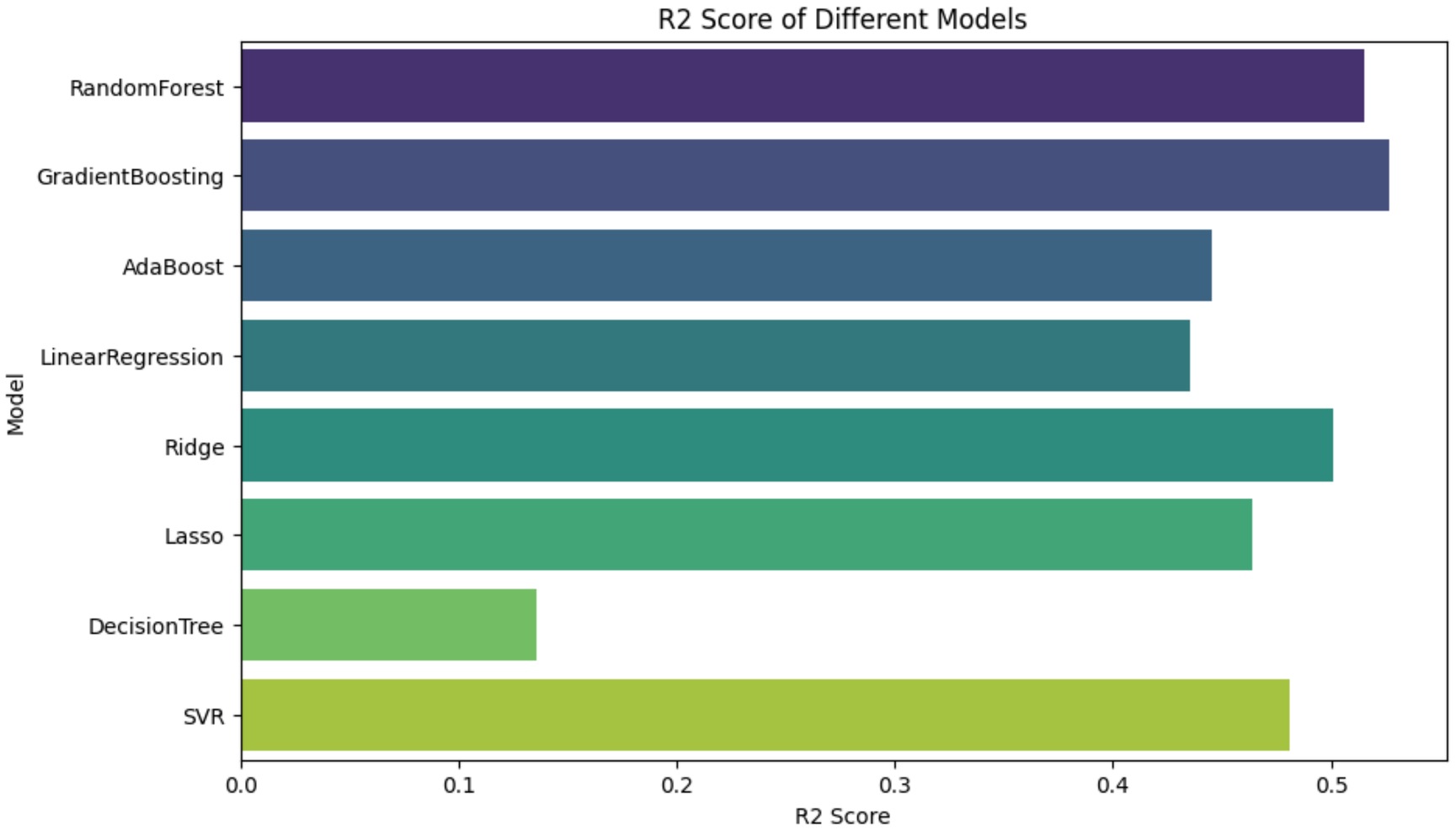
Performance Evaluation: Each model is evaluated using performance metrics such as:

- Mean Squared Error (MSE)
- Mean Absolute Error (MAE)
- R-squared (R2) score

Cross-Validation: Cross-validation is performed for the best model to assess its generalizability.

Best Model: The model with the highest R2 score (Gradient Boosting) was selected for further analysis and deployment.

OVERALL MODEL PERFORMANCE



CONCLUSION

Best Model

GradientBoostingRegressor achieved the highest R2 score of 0.53.

Weakest Model

DecisionTree performed poorly with an R2 of 0.14.

FUTURE WORK

Model Tuning

Fine-tune hyperparameters of the GradientBoostingRegressor and RandomForest models for improved performance.

Feature Engineering

Explore additional features and advanced techniques to enhance predictive accuracy.

Ensemble Methods

Combine multiple models using ensemble techniques to boost model robustness and performance.