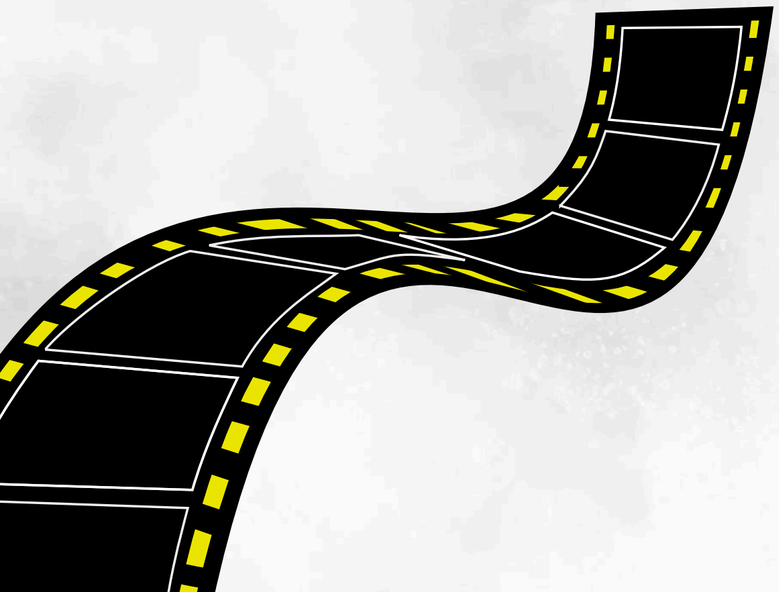
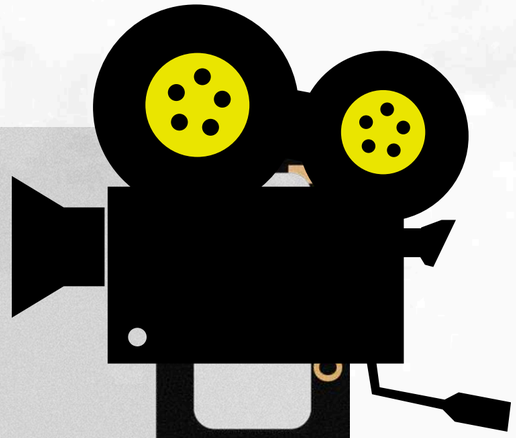
# Pre-release Movie Success prediction

Varun K N, Ashmith K P,
Swaminath Reddy

# Problem statement

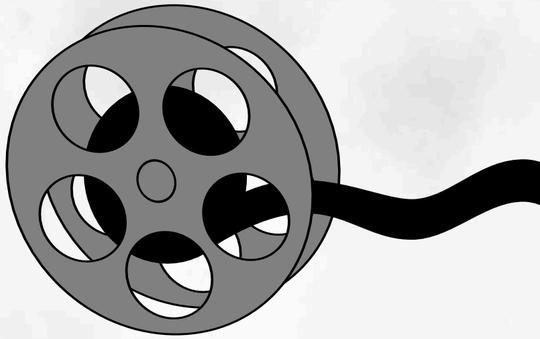


How might we predict movie commercial success using pre-release data through machine learning classification models ?

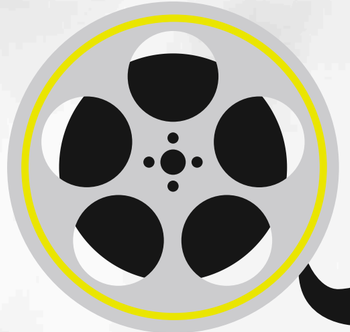


# Significance

- Enables **OTT platforms and satellite rights** buyers to evaluate a movie's potential success before release.
- Helps producers make data-driven **investment decisions** and reduce financial risk.
- Assists filmmakers and distributors in **optimizing marketing, promotion**, and release strategies.



# Literature survey



# Movie Success and Rating Prediction Using Data Mining

Published in: Journal of Scientific Research and Technology (JSRT), 2024

**Ambresh Bhadrashetty1, Surekha Patil2**

## Overview

- Uses data mining + ML to predict:
  - Movie success binary (hit/flop)
  - Movie rating (out of 10)
- Focuses on combining:
  - Static features (budget, cast, etc.)
  - Dynamic features (social media trends, hashtags)

## Performance

- Overall accuracy: > 70% accuracy across test sets (both)

## Methodology

- Dataset : IMDb metadata
- Features: Budget ,Genre ,Actors, director, producer, Release date
- Algorithms used: Random Forest, Naïve Bayes

## Limitations

- Only 5 features used : loses important complexity
- Actor/director just used as basic attributes
- Weak evaluation : Only accuracy reported

# Bollywood Movie Success Prediction using Machine Learning Algorithms

**Ashutosh Kanitkar**

Published in: IEEE Conference, 2018

## Overview

- Predicts Bollywood movie success before release
- Solves:
  - Classification (success category)
  - Regression (box office revenue)
- Uses pre-release features

## Methodology

- Dataset : 250 Bollywood movies (2014–2017)
- 9 Classes used for target variable
- Features: Budget, screens ,Actor, director, producer, Hit songs, Sequel, remake
- Final feature size: ~965 features
- Algorithms used: Logistic, KNN, RF, DT, SVM, NB, ANN

## Performance

| Name of Algorithm | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| Logistic Regression | 45.33 | 0.64 | 0.45 | 0.51 |
| KNN | 49.33 | 0.86 | 0.49 | 0.62 |
| Random Forest | 49.33 | 0.41 | 0.43 | 0.37 |
| Decision Tree | 31.77 | 0.45 | 0.43 | 0.39 |
| Naive Bayes | 24 | 0.26 | 0.24 | 0.24 |
| SVM | 44 | 0.34 | 0.44 | 0.36 |
| ANN | 50 | 0.5 | 0.5 | 0.5 |

## Limitations

- Too many features (965) → overfitting risk
- Small dataset (250 movies)

# A Machine Learning Approach to Predict Movie Box-Office Success

**Nahid Quader, Md. Osman Gani**

Published at: IEEE ICCIT 2017

## Overview

- Builds a decision support system for investors
- Predicts movie success using profit
- Uses both:
  - Pre-release features
  - Post-release features
- 5-class classification based on profit

## Performance

- Accuracy
  - 48.41% (pre-release)
  - 58.41% (all features)

## Methodology

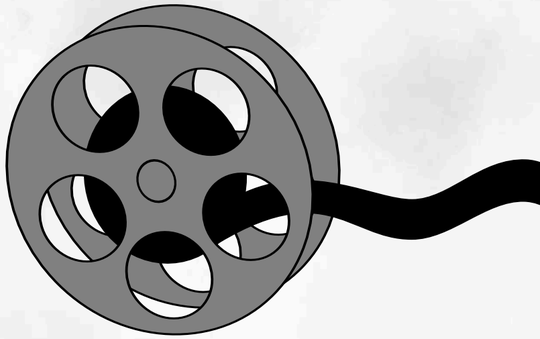
- Dataset :
  - 755 movies (after heavy filtering)
  - Sources: IMDb, Rotten Tomatoes, Metacritic
- Features: Budget, screens, release month, Ratings, reviews, sentiment
- Algorithms used: Neural network , SVM

## Limitations

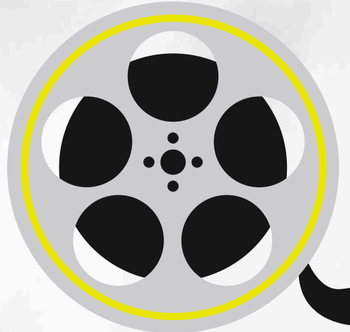
- Uses post-release features
- Star power is oversimplified → just total past revenue
- Ignores - Genre
- Dataset shrinkage

# Our improvements

- 20+ meaningful pre-release features (balanced, no overfitting)
- Improved star power modeling (actor + director + genre influence)
- Pure pre-release prediction → no data leakage
- Larger dataset (~5000 movies) → better generalization
- Better evaluation (Precision, Recall, F1 — not just accuracy)



# Data set & Feature Preprocessing



# Dataset

## Kaggle dataset

Dataset = tmdb_5000_movies.csv + tmdb_5000_credits.csv.

Raw rows: **~5,000 movies**
final cleaned ≈ **(3229, 10)**

```
original_title          0
budget                  0
runtime                 2
release_date            1
genres                  0
production_companies    0
spoken_languages        0
cast                    0
crew                    0
revenue                 0
dtype: int64
```

## TMDB dataset (Indian)

Dataset excluding Music directors.
Raw rows: **~5,000 movies**
final cleaned ≈ **(4300, 12)**

Dataset including Music directors.
Raw rows: **~5,000 movies**
final cleaned ≈ **(3000, 13)**

```
['movie_id',
 'title',
 'language',
 'release_date',
 'genres',
 'director',
 'actors',
 'producers',
 'popularity',
 'vote_average',
 'Budget.1',
 'Revenue.1']
```

&

## TMDB dataset (Hollywood)

Raw rows: **~5,000 movies**
final cleaned ≈ **(4800, 12)**

```
['movie_id',
 'title',
 'release_date',
 'genres',
 'director',
 'actors',
 'producers',
 'budget',
 'revenue',
 'popularity',
 'vote_average',
 'vote_count',
```

# Hybrid Dataset

**TMDB dataset (Indian + Hollywood)**

Raw rows: **~9000 movies**
final cleaned ≈ **(8300, 11)**

```
['title',
 'release_date',
 'language',
 'genres',
 'director',
 'actors',
 'producers',
 'popularity',
 'vote_average',
 'Budget.1',
 'Revenue.1',
```

# Pre processing Pipeline

**Dataset collection**

**Data Cleaning**

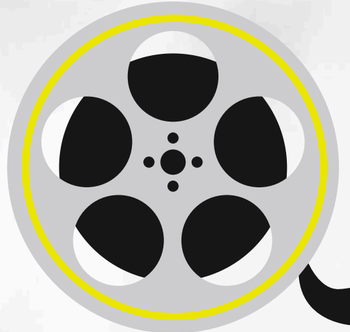
**Feature Engineering**

- **Hollywood Movies**:
  - Used TMDb as the primary source
  - Supplemented with manual web scraping for missing values
  - OMDb provided limited additional data
  - IMDb lacked a suitable public API for bulk extraction
- **Indian Movies**:
  - Initial data collected from TMDb
  - Faced significant missing values (especially budget and revenue)
  - Resolved by scraping Wikipedia to enrich and complete the dataset

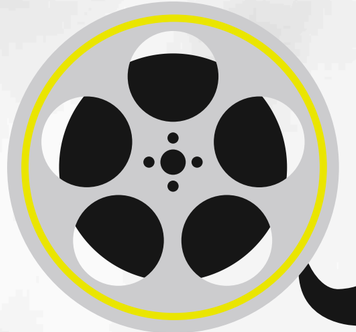
- **Handled Missing Data:**
  - Remaining missing values were minimal after enrichment and were removed
  - vote_average = 0 treated as missing
  - Created has_rating flag and imputed missing ratings using median
- **Feature Encoding:**
  - Applied one-hot encoding to genre and language
  - Removed movies with identical title and release year
- **Data Filtering:**
  - Removed very low-budget films (adjusted budget < 100,000) to reduce noise
  - Dropped placeholder records where both budget and revenue ≤ 1
- **ROI Processing** & **Label Creation**: both 2 and 3
- **Outlier Treatment** : Applied IQR-based filtering on adjusted budget

- **Success-Based Features (Historical ROI):**
  - Engineered Director Success, Actor Star Power, and Producer/Production Company Success
  - Based on past performance using ROI-driven metrics
- **Temporal Features:**
  - Extracted release month from release_date to capture seasonality effects
- **Indian Dataset model 2:**
  - Created Music Director Popularity/Success feature
  - Computed using historical performance trends

# ML Methodology



# I. Model for Indian Movies

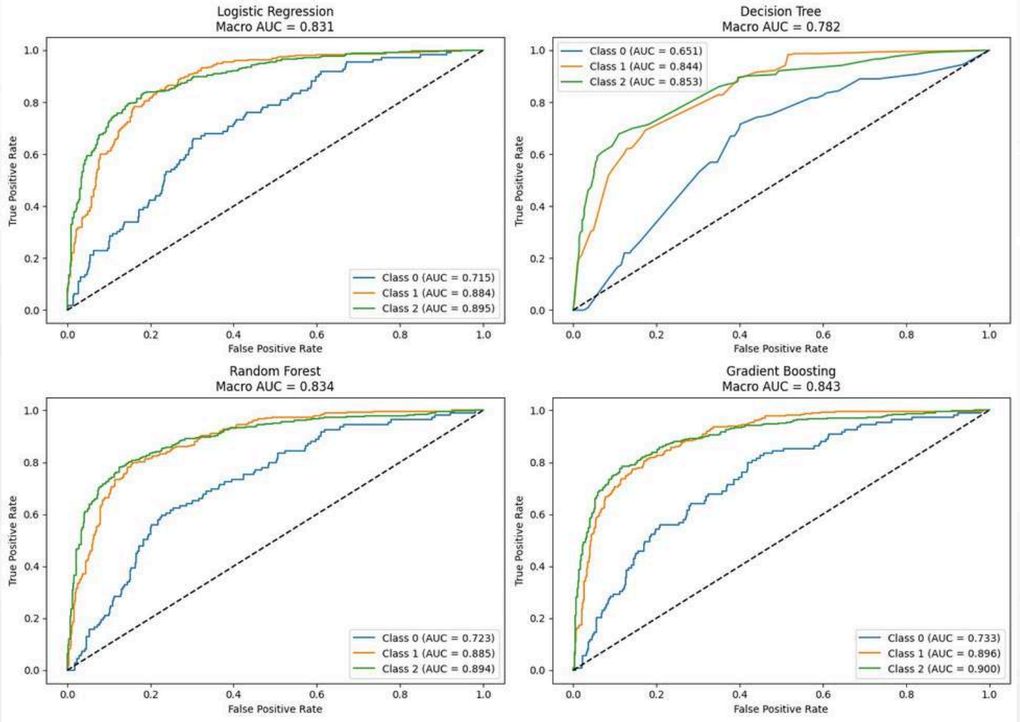


# Predicting ROI category: Flop / Avg / Hit

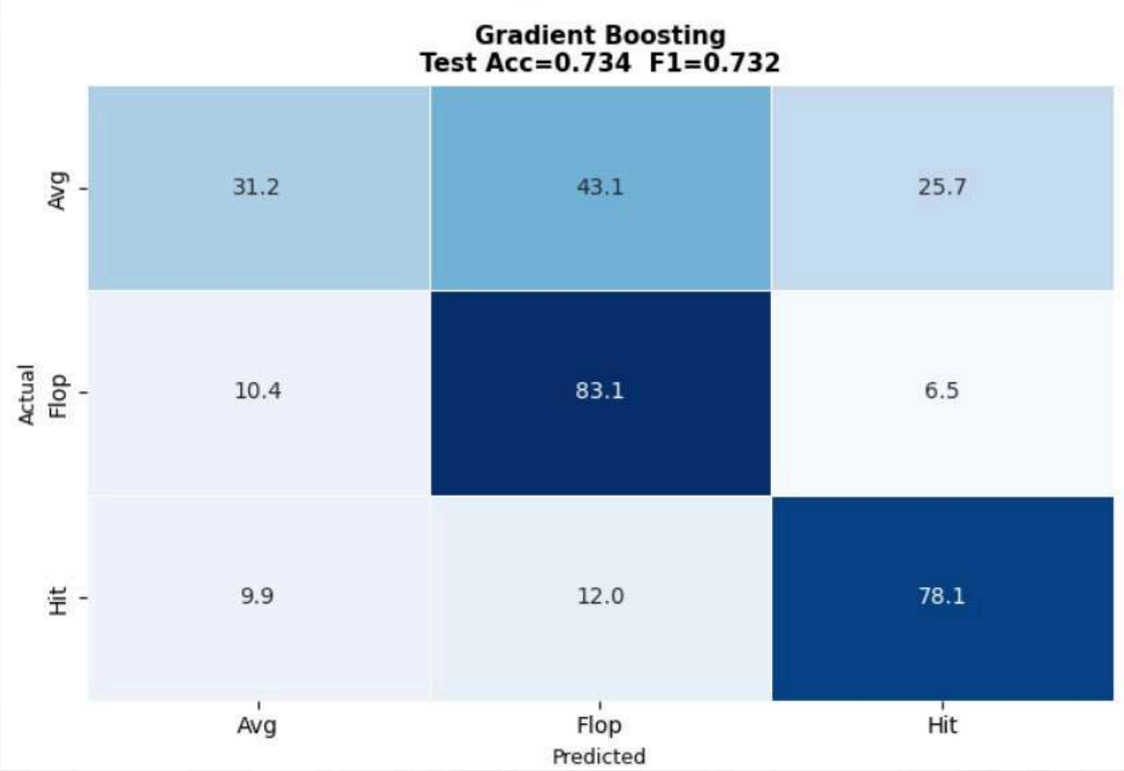
The Classification values were :

**ROI < 0 → Flop**
**0 ≤ ROI < 0.9 → Average**
**ROI ≥ 0.9 → Hit**

- ROC curves show Gradient Boosting and Random Forest outperform other models
- Gradient Boosting performs best across most metrics




| Model | Accuracy | Precision | Recall | F1 Score | AUC |
|---|---|---|---|---|---|
| Gradient Boosting | 0.733860 | 0.643935 | 0.641266 | 0.641140 | 0.842866 |
| Random Forest | 0.707510 | 0.616995 | 0.615307 | 0.613771 | 0.834218 |
| Logistic Regression | 0.677207 | 0.618114 | 0.615417 | 0.609475 | 0.831161 |
| Decision Tree | 0.608696 | 0.618708 | 0.594049 | 0.572718 | 0.782449 |




Random Forest

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.34 | 0.31 | 0.33 | 109 |
| 1 | 0.74 | 0.83 | 0.79 | 308 |
| 2 | 0.85 | 0.78 | 0.81 | 342 |
| accuracy | | | 0.73 | 759 |
| macro avg | 0.64 | 0.64 | 0.64 | 759 |
| weighted avg | 0.73 | 0.73 | 0.73 | 759 |

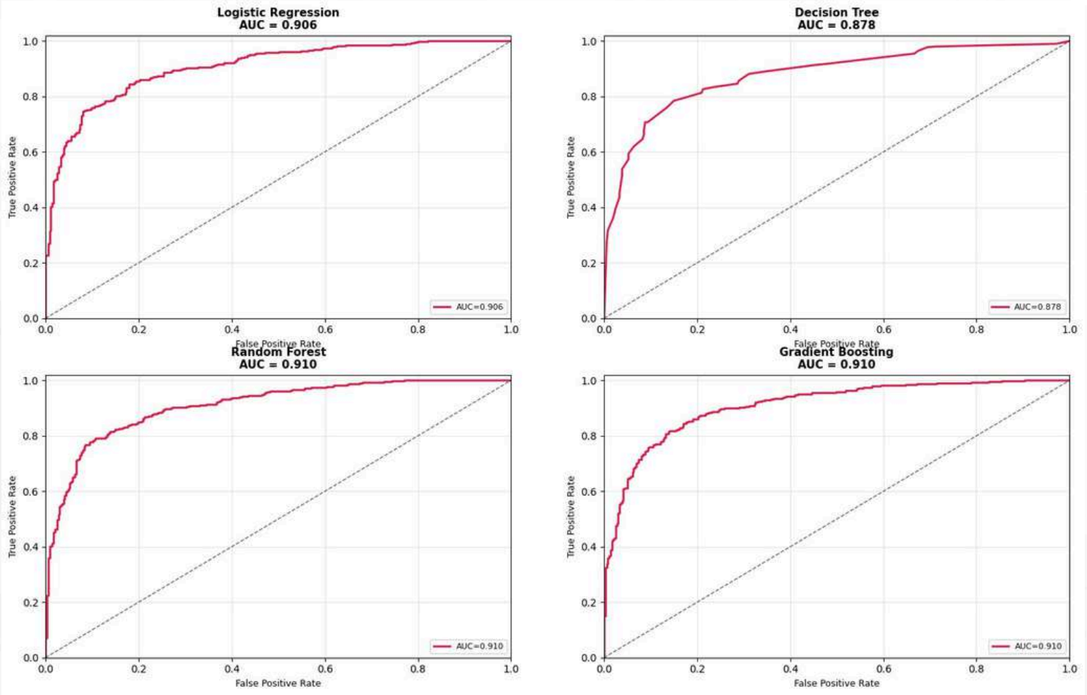
Best Model: Gradient Boosting
Accuracy 73%

# Predicting ROI category: Flop / Hit

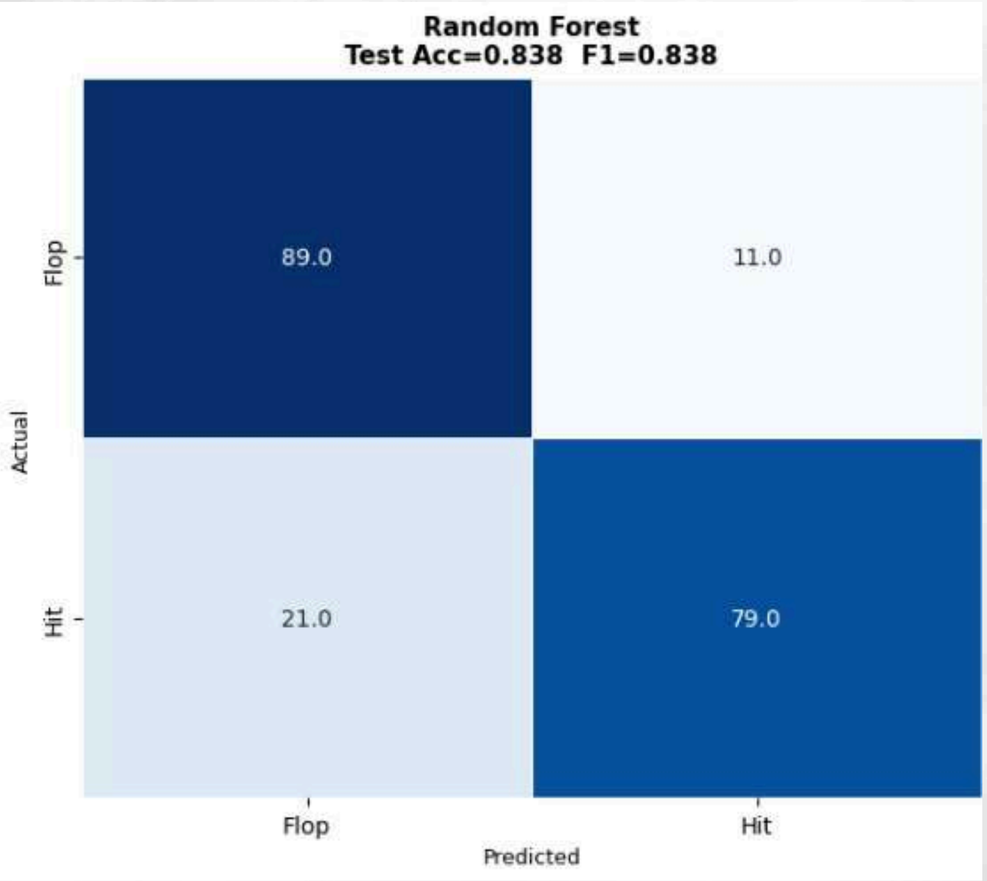
The Classification values were :

**ROI < 0.5→ Flop**
**ROI ≥ 0.5 → Hit**

- Random Forest achieves highest AUC (~0.91)
- All models perform significantly better than tertiary classification
- Balanced Precision & Recall




| Model | Accuracy | Precision | Recall | F1 Score | AUC |
|---|---|---|---|---|---|
| Random Forest | 0.837945 | 0.843107 | 0.837945 | 0.837824 | 0.910232 |
| Gradient Boosting | 0.831357 | 0.834336 | 0.831357 | 0.831366 | 0.910280 |
| Logistic Regression | 0.823452 | 0.828285 | 0.823452 | 0.823336 | 0.906350 |
| Decision Tree | 0.810277 | 0.815869 | 0.810277 | 0.810080 | 0.877626 |




```
Random Forest
==========================================================================
              precision    recall  f1-score   support

           0       0.80      0.89      0.84       363
           1       0.89      0.79      0.84       396

    accuracy                           0.84       759
   macro avg       0.84      0.84      0.84       759
weighted avg       0.84      0.84      0.84       759
```

Best Model: Random Forest

Accuracy 84%

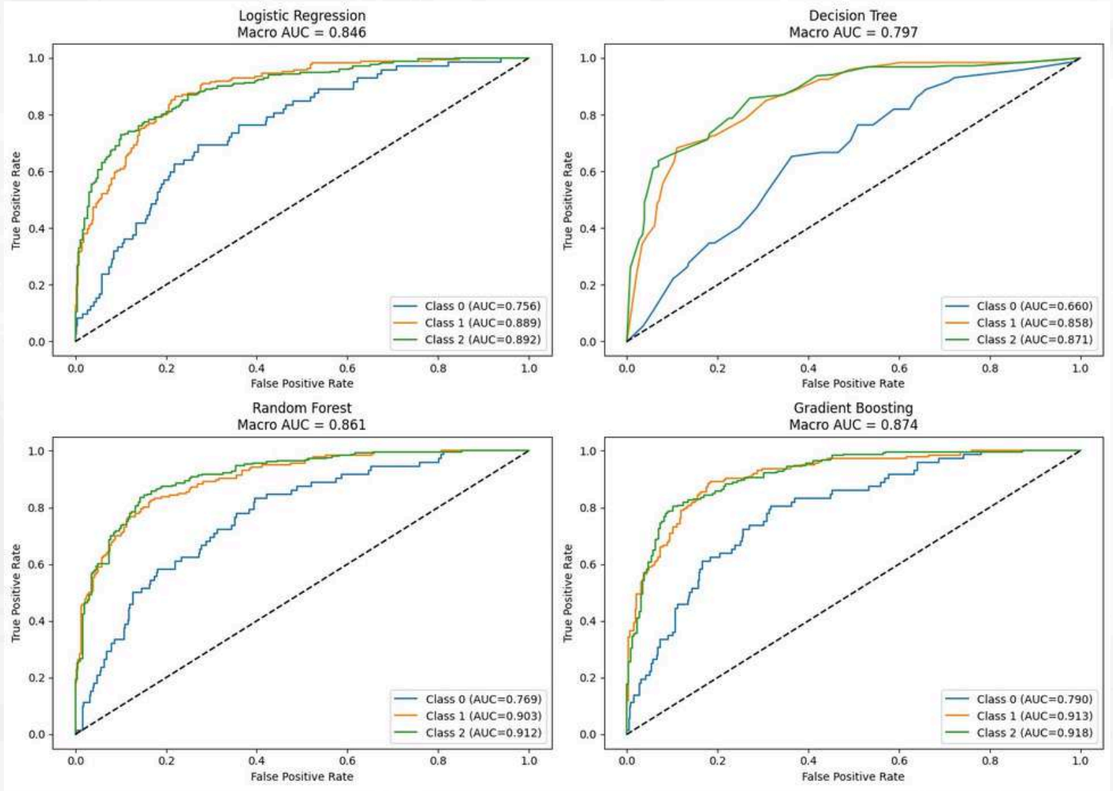
# Impact of Music Director on ROI Prediction

## Predicting ROI category: Flop /Average/ Hit

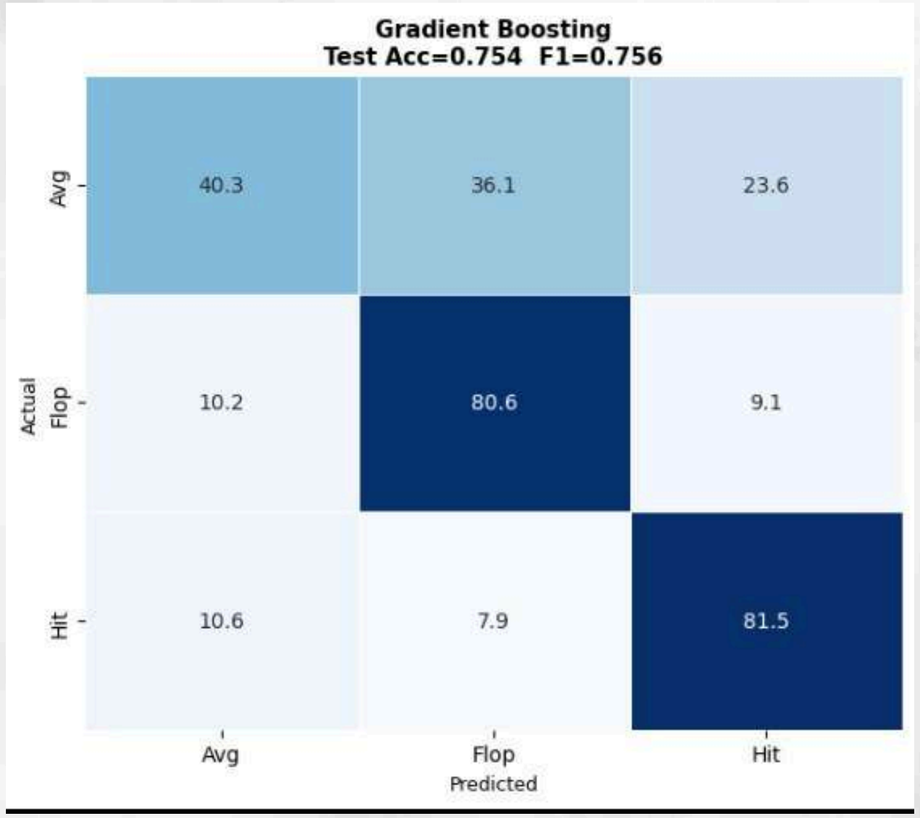
The Classification values were :

**ROI < 0 → Flop**
**0 ≤ ROI < 0.9 → Average**
**ROI ≥ 0.9 → Hit**

- Performance improves compared to base tertiary model (AUC ~0.84 → ~0.87)
- Gradient Boosting remains the best-performing model
- Random Forest shows competitive performance




| Model | Accuracy | Precision | Recall | F1 Score | AUC |
|---|---|---|---|---|---|
| Gradient Boosting | 0.753906 | 0.670298 | 0.674730 | 0.672087 | 0.873637 |
| Random Forest | 0.730469 | 0.667057 | 0.683601 | 0.668355 | 0.861191 |
| Logistic Regression | 0.677734 | 0.624386 | 0.630978 | 0.617327 | 0.845694 |
| Decision Tree | 0.630859 | 0.593085 | 0.576700 | 0.571304 | 0.796509 |



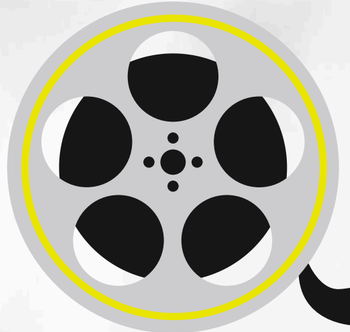

```
Gradient Boosting
==========================================================================
              precision    recall  f1-score   support

           0       0.39      0.40      0.39        72
           1       0.77      0.81      0.79       186
           2       0.86      0.81      0.84       254

    accuracy                           0.75       512
   macro avg       0.67      0.67      0.67       512
weighted avg       0.76      0.75      0.76       512
```

Best Model: Gradient Boosting
Accuracy 75%

# II. Model for Hollywood Movies

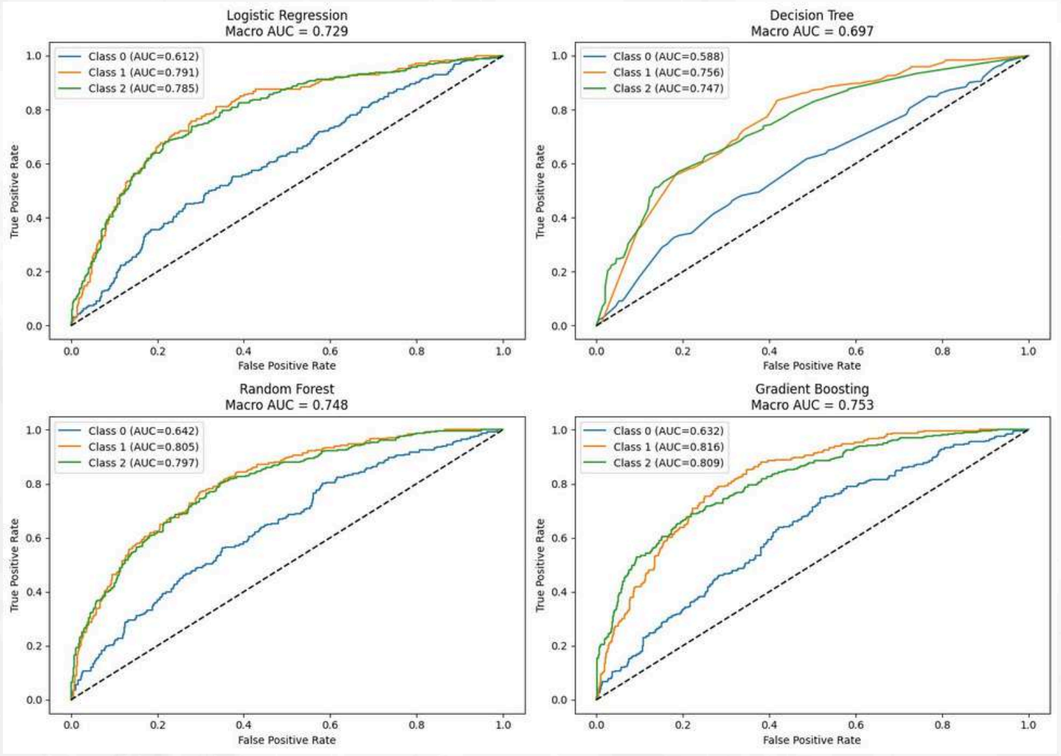


## Predicting ROI category: Flop / Avg / Hit

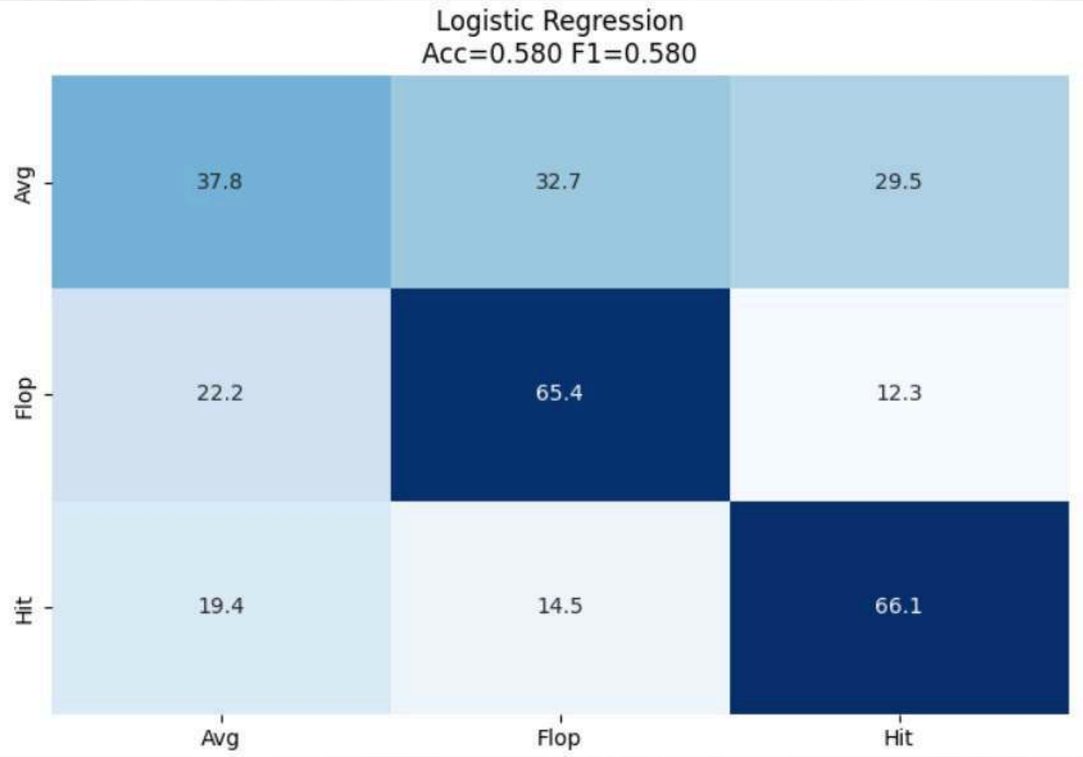
The Classification values were :

**ROI < 0 → Flop**
**0 ≤ ROI < 1.5 → Average )**
**ROI ≥ 1.5 → Hit**

- Models trained on engineered features perform better on Indian dataset than Hollywood dataset
- Overall performance is lower across all models




| Model | Accuracy | Precision | Recall | F1 Score | AUC |
|---|---|---|---|---|---|
| Logistic Regression | 0.580466 | 0.555949 | 0.564580 | 0.557346 | 0.729459 |
| Random Forest | 0.574917 | 0.545877 | 0.553097 | 0.546944 | 0.747779 |
| Gradient Boosting | 0.573807 | 0.541745 | 0.546750 | 0.543398 | 0.752615 |
| Decision Tree | 0.542730 | 0.531577 | 0.532165 | 0.529929 | 0.696860 |




```
Logistic Regression
                   precision    recall  f1-score   support

                0       0.42      0.38      0.40       251
                1       0.53      0.65      0.59       243
                2       0.72      0.66      0.69       407

         accuracy                           0.58       901
        macro avg       0.56      0.56      0.56       901
     weighted avg       0.58      0.58      0.58       901
```

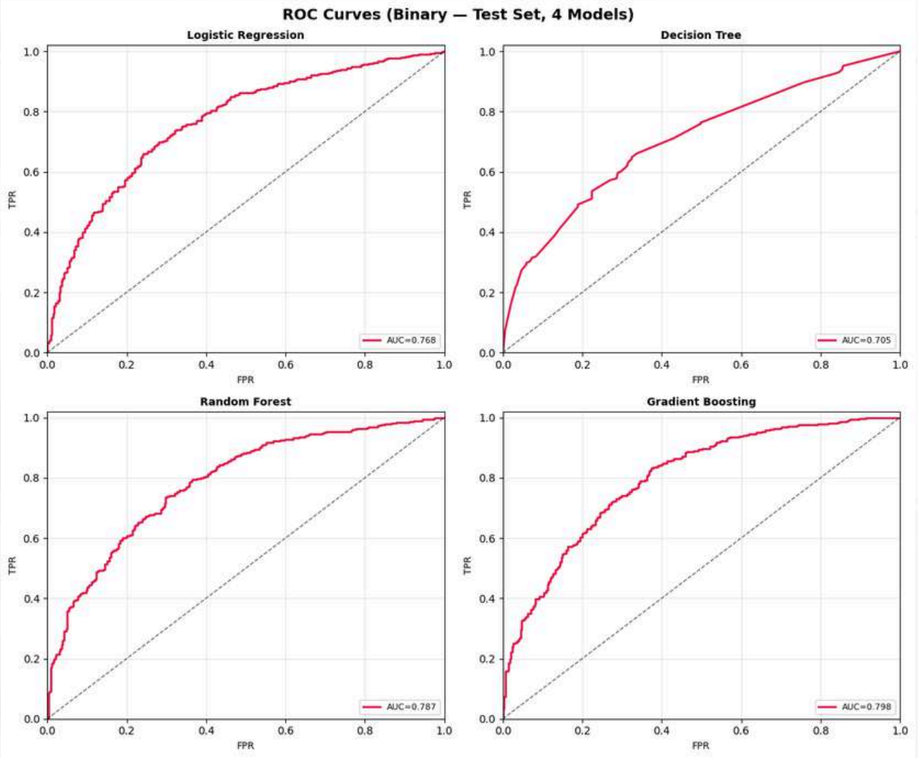
Best Model: Logistic regression
Accuracy 58%

# Predicting ROI category: Flop / Hit

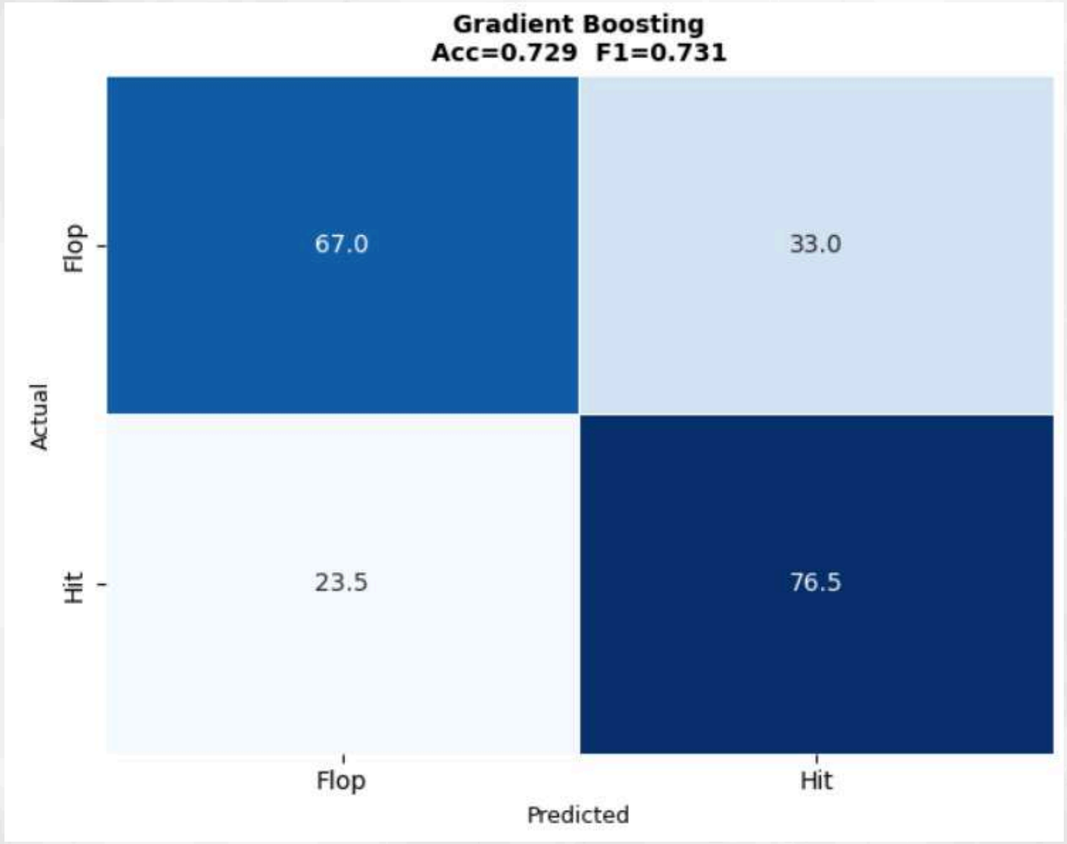
The Classification values were :

**ROI ≥ 0.5 → Hit**
**ROI < 0.5 → Flop**

- Gradient Boosting performs best with balanced accuracy and AUC (~0.84).
- Binary setup improves model stability and performance




| Model | Accuracy | Precision | Recall | F1 Score | AUC |
|---|---|---|---|---|---|
| Gradient Boosting | 0.733860 | 0.643935 | 0.641266 | 0.641140 | 0.842866 |
| Random Forest | 0.707510 | 0.616995 | 0.615307 | 0.613771 | 0.834218 |
| Logistic Regression | 0.677207 | 0.618114 | 0.615417 | 0.609475 | 0.831161 |
| Decision Tree | 0.608696 | 0.618708 | 0.594049 | 0.572718 | 0.782449 |



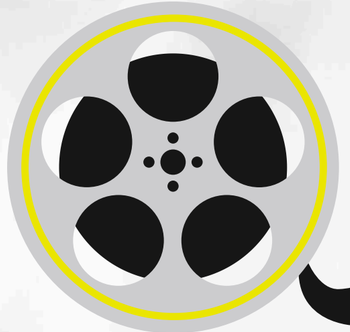

```
Gradient Boosting
==========================================================
              precision    recall  f1-score   support

        Flop       0.63      0.67      0.65       339
         Hit       0.79      0.77      0.78       562

    accuracy                           0.73       901
   macro avg       0.71      0.72      0.71       901
weighted avg       0.73      0.73      0.73       901
```

Best Model: Gradient Boosting
Accuracy 73%

# III. Model for Hybrid movies

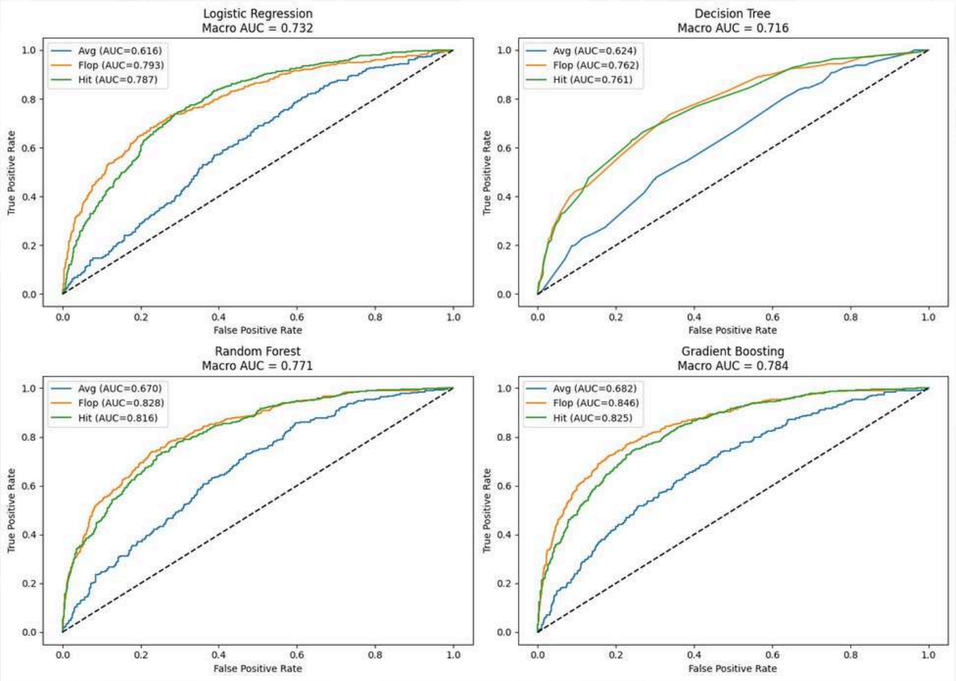


## Predicting ROI category: Flop / Avg / Hit

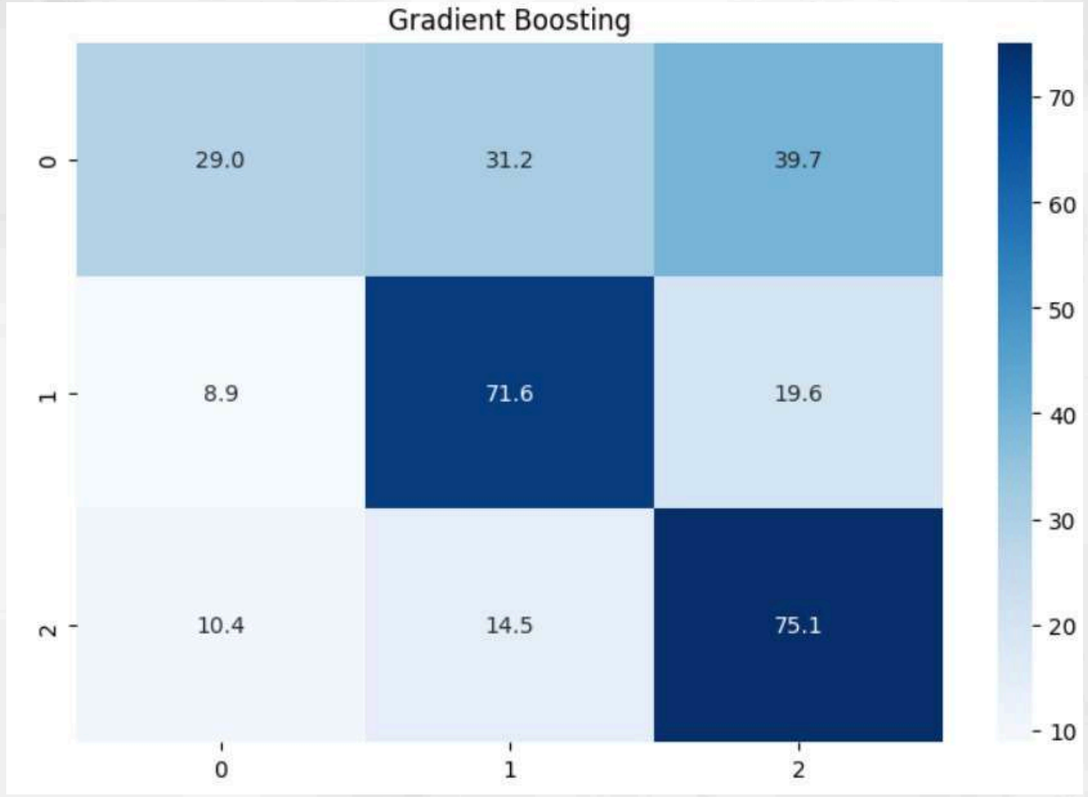
The Classification values were :

**ROI < 0 → Flop**
**0 ≤ ROI < 0.9 → Average**
**ROI ≥ 0.9 → Hit**

- Model performance decreases on combined dataset (AUC ~0.78), indicating higher variability across industries.
- Gradient Boosting still performs best




| Model | Accuracy | Precision | Recall | F1 Score | AUC |
|---|---|---|---|---|---|
| Gradient Boosting | 0.663860 | 0.589423 | 0.585638 | 0.585595 | 0.784438 |
| Random Forest | 0.619363 | 0.551175 | 0.553562 | 0.551337 | 0.771099 |
| Logistic Regression | 0.552014 | 0.528241 | 0.527289 | 0.514215 | 0.731699 |
| Decision Tree | 0.578473 | 0.510530 | 0.512870 | 0.505308 | 0.715774 |




Gradient Boosting

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.37 | 0.29 | 0.32 | 272 |
| 1 | 0.66 | 0.72 | 0.68 | 552 |
| 2 | 0.74 | 0.75 | 0.75 | 839 |
| accuracy | | | 0.66 | 1663 |
| macro avg | 0.59 | 0.59 | 0.59 | 1663 |
| weighted avg | 0.65 | 0.66 | 0.66 | 1663 |

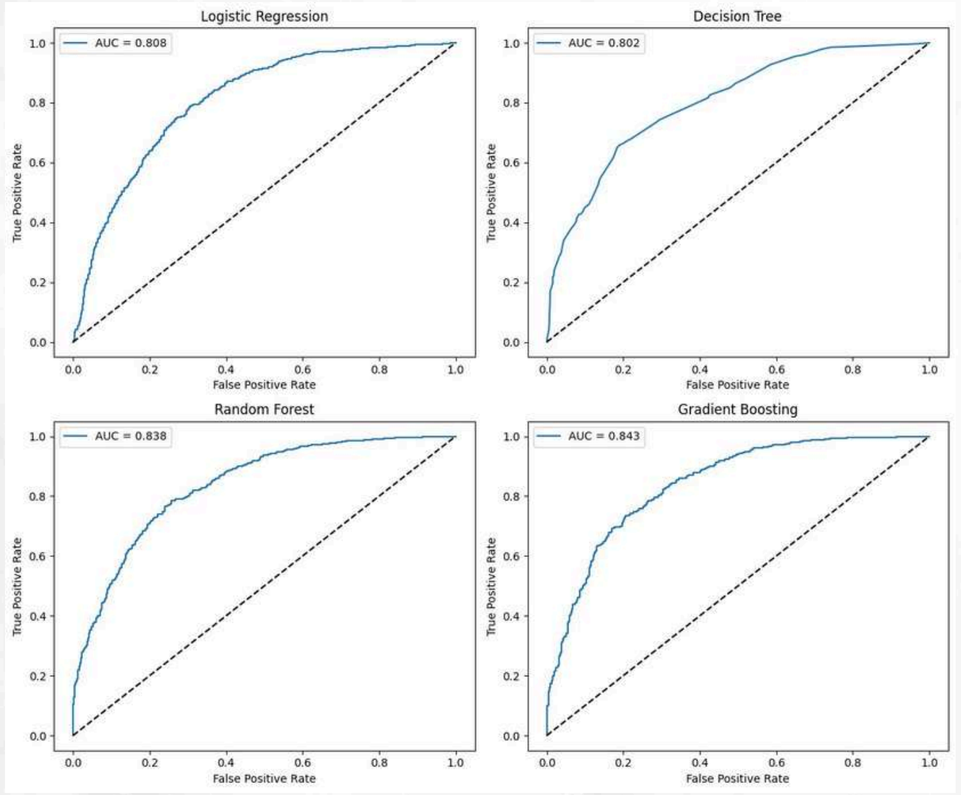
Best Model: Gradient Boosting
Accuracy 66%

## Predicting ROI category: Flop / Hit

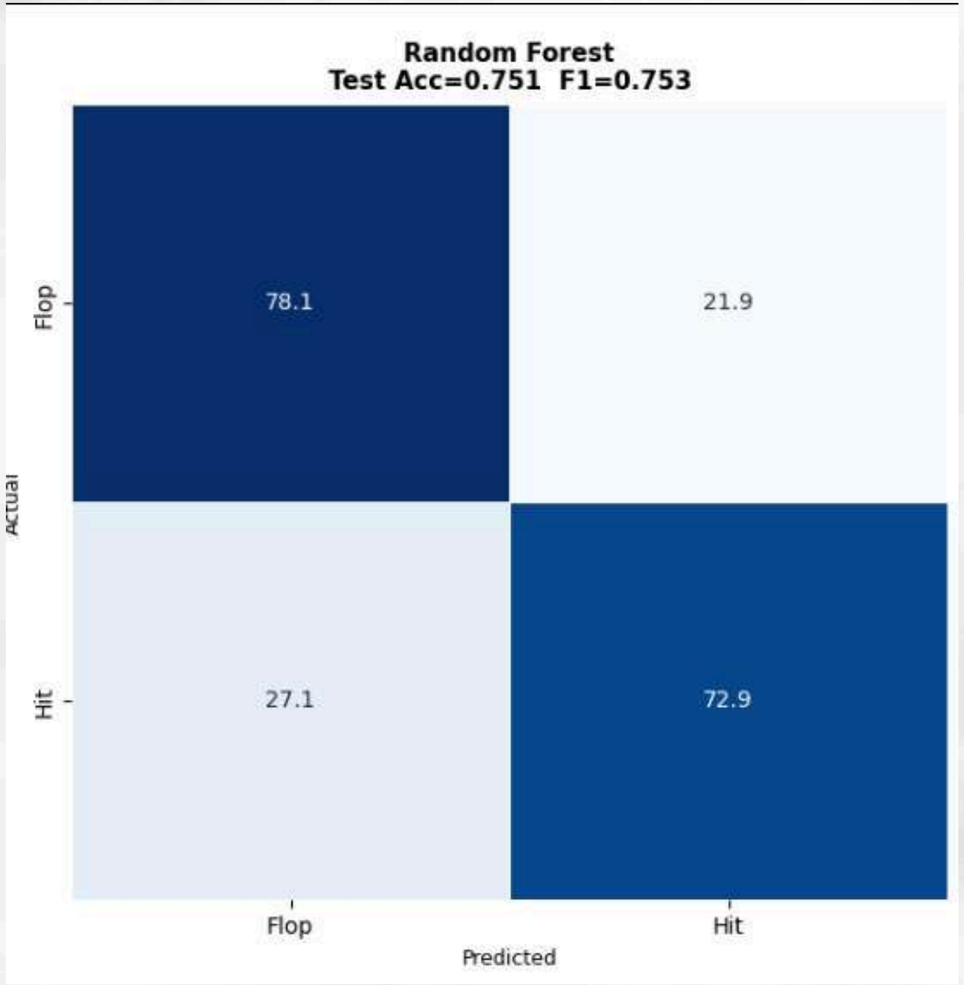
The Classification values were :

**ROI < 0.5→ Flop**
**ROI ≥ 0.5 → Hit**

- Binary classification improves performance on combined dataset (AUC ↑ to ~0.84)
- Simplifying the problem (binary) helps recover performance even with diverse datasets..




| Model | Accuracy | Precision | Recall | F1 Score | AUC |
|---|---|---|---|---|---|
| Gradient Boosting | 0.756464 | 0.808676 | 0.757292 | 0.782141 | 0.842723 |
| Random Forest | 0.751052 | 0.819672 | 0.729167 | 0.771775 | 0.837583 |
| Decision Tree | 0.722189 | 0.825916 | 0.657292 | 0.732019 | 0.801934 |
| Logistic Regression | 0.711966 | 0.806369 | 0.659375 | 0.725501 | 0.808160 |



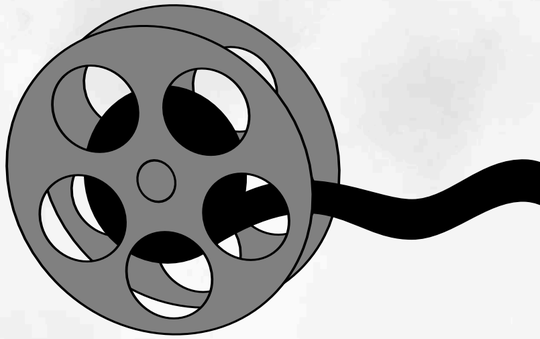

Random Forest

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.68 | 0.78 | 0.73 | 703 |
| 1 | 0.82 | 0.73 | 0.77 | 960 |
| accuracy | | | 0.75 | 1663 |
| macro avg | 0.75 | 0.76 | 0.75 | 1663 |
| weighted avg | 0.76 | 0.75 | 0.75 | 1663 |

Best Model: Random Forest
Accuracy 75%

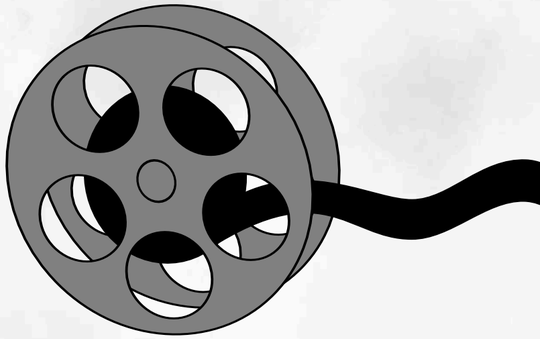
# Challenges Faced

- Limited availability of large and repeated movie data for actors, directors, and producers, which affected the model's ability to learn patterns effectively.
- Overfitting of machine learning models due to limited and insufficient training data



# Future scope

- Increase the dataset size by including more movies for each actor, director, and producer so the model can learn patterns more effectively and improve prediction performance.
- Integrate real-time audience sentiment from social media, trailers, and reviews for dynamic prediction.
- Use deep learning and advanced AI models for better feature extraction and prediction.



# References


- "A Machine Learning Approach to Predict Movie Box-Office Success (Quader et al., 2017)"
- "Movie Success and Rating Prediction Using Data Mining"
- "Predicting Box-Office Markets with Machine Learning Methods" (2022)
- "Bollywood Movie Success Prediction using Machine Learning Algorithms"

# Thank You