

DATA MINING FINAL PROJECT

Movie Success & Revenue Prediction

GROUP MEMBERS:

DHARMIK KURLAWALA – 2886995

NITEESH SINGH – 2886321

ADVISOR:

DR. SUNNIE CHUNG





Movie Success & Revenue Prediction

Welcome to our data mining project on predicting movie success and revenue. We've developed a machine learning system that can classify films as hits or flops before release and estimate their box office performance.

Our approach combines artificial neural networks(ANN) with gradient boosting to analyze pre-release attributes like budget, cast star power, and genre. This presentation will walk you through our end-to-end process from data collection to deployment.



Project Overview



Problem

Only a fraction of movies turn profitable. Can we predict a movie's success or flop before release to help studios minimize financial risk?



Objective

Use machine learning to classify movies as "Success" vs "Flop" and predict box-office revenue based on pre-release attributes.



Approach

Implement an end-to-end ML pipeline from data collection to predicting hit or flop.

Data Collection

Data Sources

We utilized two public Kaggle datasets from The Movie Database (TMDb):

- Dataset 1: Main movie metadata (title, budget, revenue) for ~10k movies
- Dataset 2: Additional details (cast, crew, IMDb ratings) for ~1M movies

Merge Strategy

Used unique movie ID as primary key to join datasets, enriching each movie entry with cast, director, writers, etc.

Result: A single combined dataset with extensive features per movie (TMDB_movie_dataset_Full.csv)

Dataset Description

- **Raw Dataset Size:** ~10,000 movies (after merging)
- **Features (Pre-cleaning):**
 - **Numeric:** Budget, Revenue, Runtime, Popularity, Ratings (TMDB vote_average/count, IMDb rating/votes)
 - **Categorical/Text:** Title, Release date, Genres, Languages, Overview (plot summary), Production companies & countries, Cast, Director
- **Initial Issues:** Missing values in some fields (e.g. budget, runtime), some movies with zero runtime or negligible budget
- **Class Definition:**
 - Defined "Success" as revenue > 1.5× budget (ROI > 150%), otherwise "Flop"
 - Binary label = 1 (Success) or 0 (Flop) created accordingly
 - 8,835 movies remained after dropping invalid entries (runtime 0, missing budget/revenue)

Data Cleaning & Preprocessing

Filtering

Removed movies with incomplete data (dropped NAs for key fields) and unrealistic values (budget < \$1000, revenue ≤ \$0)

Label Creation

Created binary label using budget vs. revenue (1 = success, 0 = flop)

Log Transformation

Added log_budget and log_revenue to reduce skew (many movies have low revenue, a few very high)

Train/Test Split

Stratified split (70% train, 30% test) ensuring similar success/flop ratio in each

Scaling

Applied Quantile Transformer to numeric features (budget, runtime, etc.) – maps them to a normal distribution for stable model training

Final Features Selected

1 Star Power Features:

cast_power, director_power,
writers_power, producers_power,
production_companies_power,
production_countries_power

2 Movie Attributes:

runtime (movie length), log_budget
(log-transformed budget),
overview_sentiment (sentiment
polarity of the synopsis),
release_year, release_month,
is_holiday_release (Nov/Dec
releases), is_sequel (sequel
indicator from title), big_studio
(major studio involvement),
director_hit_ratio (director's past
success rate)

3 Text Features:

50 TF-IDF features extracted from
movie overviews (tfidf_0 to
tfidf_49)

4 Genre Features (Multi-hot encoded):

genre_action, genre_comedy, genre_drama,
genre_adventure, etc. (~20 genre features)

5 Language Features (One-hot encoded):

lang_english, lang_french, lang_spanish, lang_hindi, etc.
(~80+ languages)

Feature Encoding

Genres & Languages

Multi-label binarization (Multi-hot encoding) for 19 genres and 114 languages (each becomes a binary feature)



Text Overview

Extracted TF-IDF features from movie description (50 top terms) to capture text information

Final Feature Set

198 features per movie (after encoding), ready for modeling (stored in movie_features.csv)



Feature Engineering



Star Power

Cast_power, Director_power, Writers_power, Producers_power –
Numeric scores assessing the past success of top cast/crew



Director Hit Ratio

Fraction of a director's past films that were successes (e.g., if a director made 10 films and 6 were hits, director_hit_ratio = 0.6)



Movie Indicators

is_sequel (Binary flag if title suggests a sequel), big_studio (if any production company is a major studio), is_holiday_release (1 if released in Nov/Dec)



Sentiment Analysis

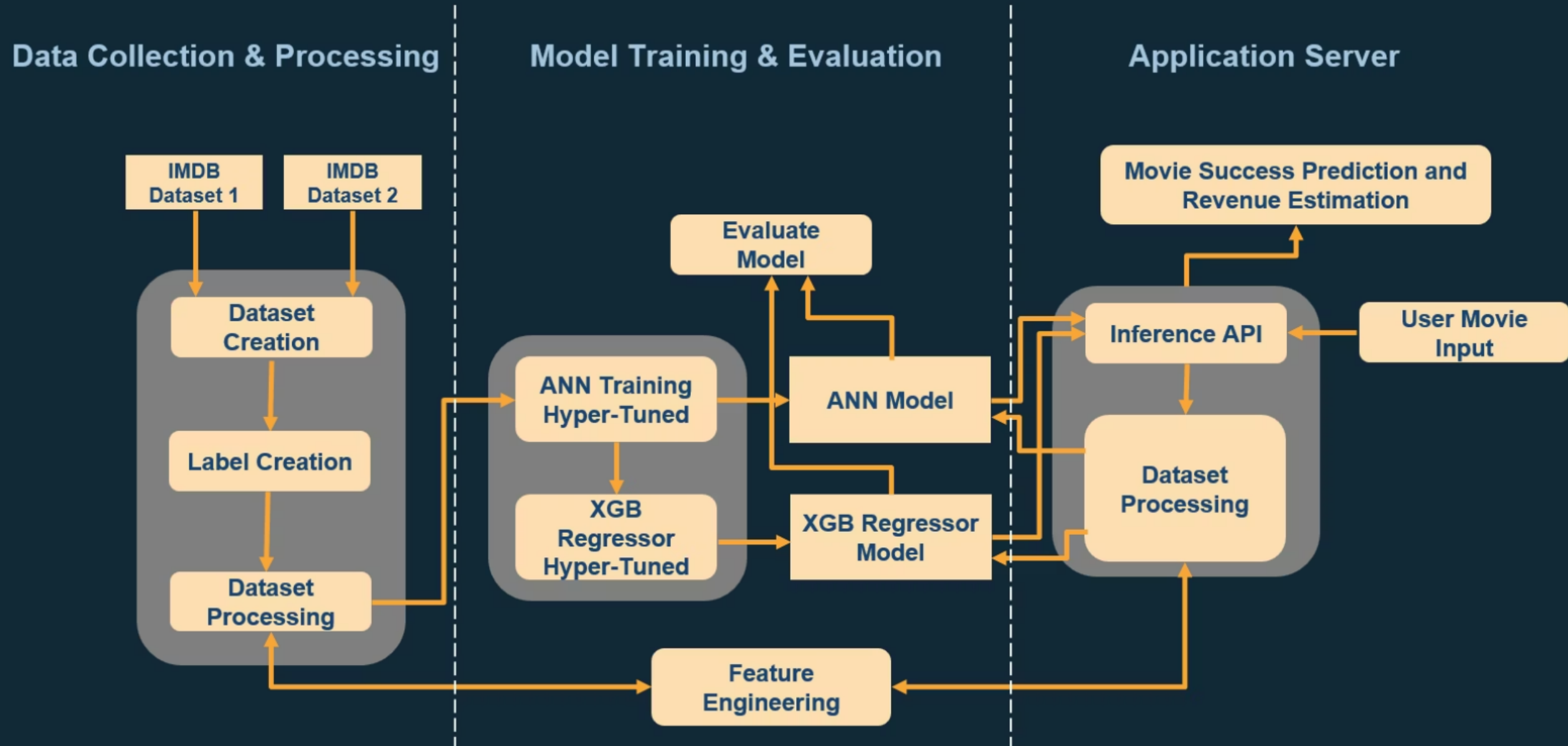
overview_sentiment – Sentiment score of the plot summary (range clipped to [-0.5, 0.5]), gauging positive/negative buzz

Feature Comparison

• **Result:** Our enhanced feature matrix systematically captures critical domain knowledge including star power, director track record, sequel status, and content sentiment. These domain-informed transformations significantly improve model predictive capacity beyond raw data.

Original Feature	Engineered Feature(s)	Description
Cast (list of actors)	cast_power (numeric)	Cumulative influence score of top 3 actors based on their previous box office performance
Director (name)	director_power, director_hit_ratio	Director's commercial influence score; proportion of successful films in their portfolio (0–1)
Title	is_sequel (0/1)	Binary indicator identifying if the title suggests a sequel or franchise installment
Overview (text)	overview_sentiment, tfidf_O...tfidf_49	Emotional tone score of plot summary; Vector of 50 key terms capturing content themes
Genres (list)	genre_* (Multi-hot for each genre)	Binary features for each genre category (e.g., genre_Action=1 if film is Action)
Spoken Languages (list)	lang_* (Multi-hot for each language)	Binary features for each language (e.g., lang_English=1 if English is spoken)

System Architecture



System Architecture

Data Collection & Processing

- 1 Star Power Features**
Scores for the cast, director, writers, producers, production companies, and production countries
- 2 Movie Attributes**
Runtime, log-transformed budget, sentiment of the synopsis, release year/month, sequel/holiday flags, major studio involvement, director's past success rate
- 3 Text Features**
50 TF-IDF features extracted from the movie overviews
- 4 Genre & Language Features**
One-hot encoded genre and language indicators

Model Training

The engineered features are used to train two machine learning models:

- 1 ANN Classifier**
A neural network is trained to predict the probability of a movie being a "Success" vs a "Flop".
- 2 XGBoost Regressor**
An XGBoost model is trained to predict the log-transformed revenue of a movie, using the engineered features plus the ANN classifier's probability output.

Hyperparameter tuning is performed on the ANN model to find the optimal architecture and training settings.

Evaluation

The performance of both models is evaluated on a held-out test set:

- 1 Classifier Metrics**
Accuracy, ROC-AUC, Precision-Recall, Calibration curve
- 2 Regressor Metrics**
Mean Absolute Error (MAE) and R-squared (R^2)

The best-performing model weights and calibration parameters are saved for deployment.

ANN Classifier Model

Input Layer

198 engineered features

Hidden Layers

- Layer 1: 128 neurons, ReLU activation, L2 regularization ($\lambda=1e-4$)
- Layer 2: 64 neurons, ReLU + Batch Norm
- Layer 3: 32 neurons, ReLU

Output Layer

1 neuron with Sigmoid activation (success probability)

Model Training

1 Optimizer & Loss

Trained the ANN classifier using the **Adam** optimizer and **Binary Cross entropy** loss function.

2 Class Balancing

Handled the slight class imbalance by applying **class weights**, ensuring a balanced training focus on both success and flop predictions.

3 Early Stopping

Monitored the validation loss and stopped training when there was no improvement for 8 epochs, avoiding model overfitting.

4 Training Details

Trained the model for up to 60 epochs, with an effective ~20 epochs until convergence. Used a batch size of 64.

The final trained classifier model (`cls_model.keras`) was saved for evaluation and later use in the deployment pipeline.

XGBoost Regressor Model



Objective

Predict the log of revenue, enabling direct prediction of box-office receipts in dollars (after exponential transform)



Features

All engineered features plus the ANN's predicted success probability, enriching the regressor's input with an estimate of movie quality



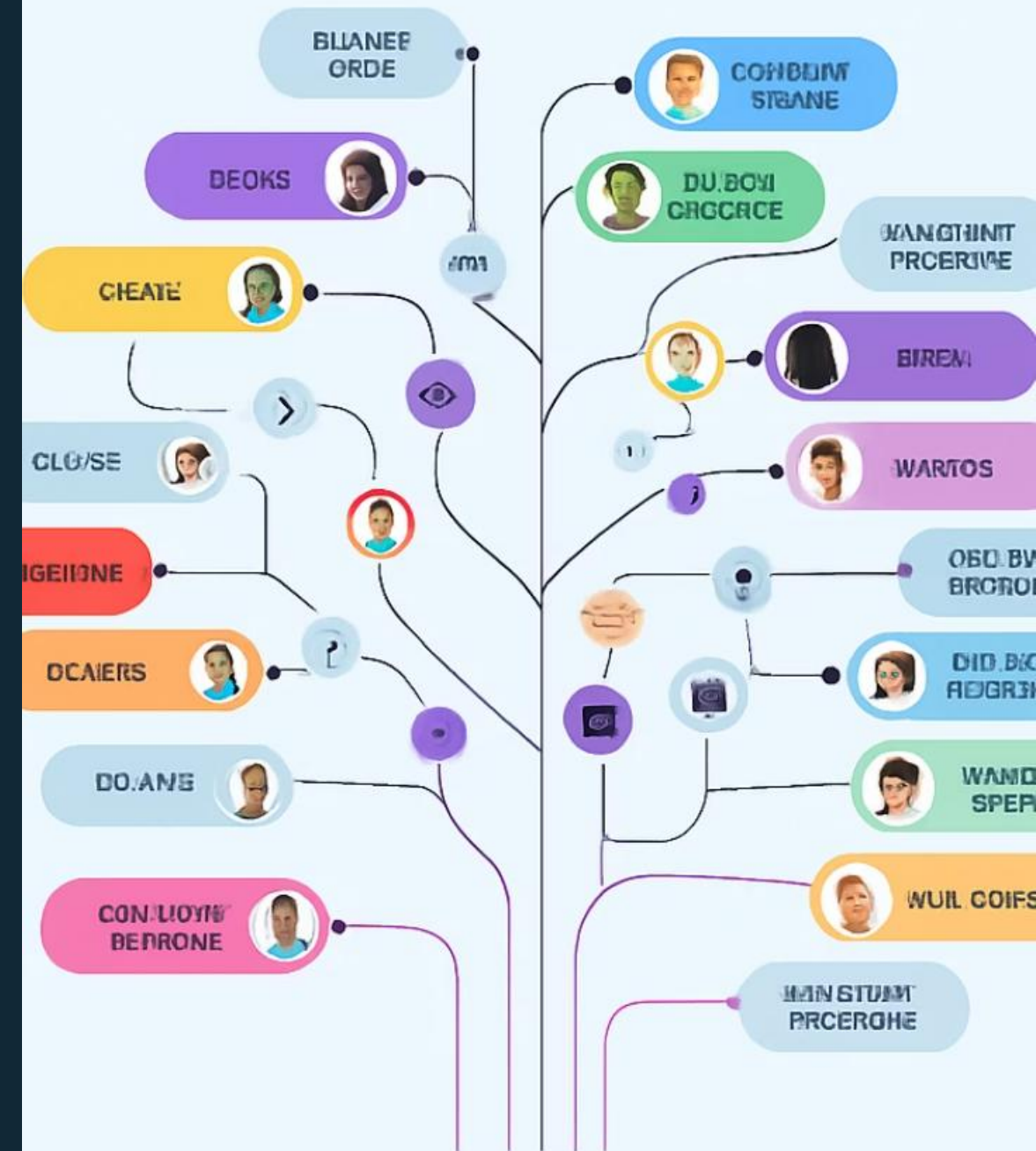
Training Strategy

5-fold Cross-Validation to find optimal parameters: max_depth=6, learning rate eta=0.05, subsample 80%, with early stopping

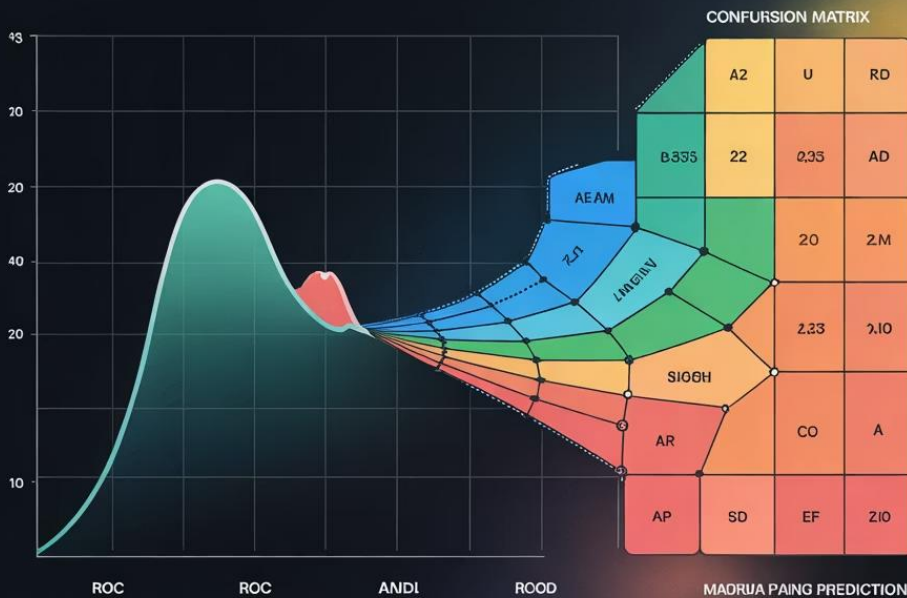


Regularization

Used tree depth limit and column subsampling to prevent overfitting and ensure good generalization



Movie Success Prediction Analysis



Classification Performance

~84%

Accuracy

Percentage of movies correctly classified as success or flop

0.92

ROC-AUC

Indicates good discrimination between successful and flop movies

~82%

Precision

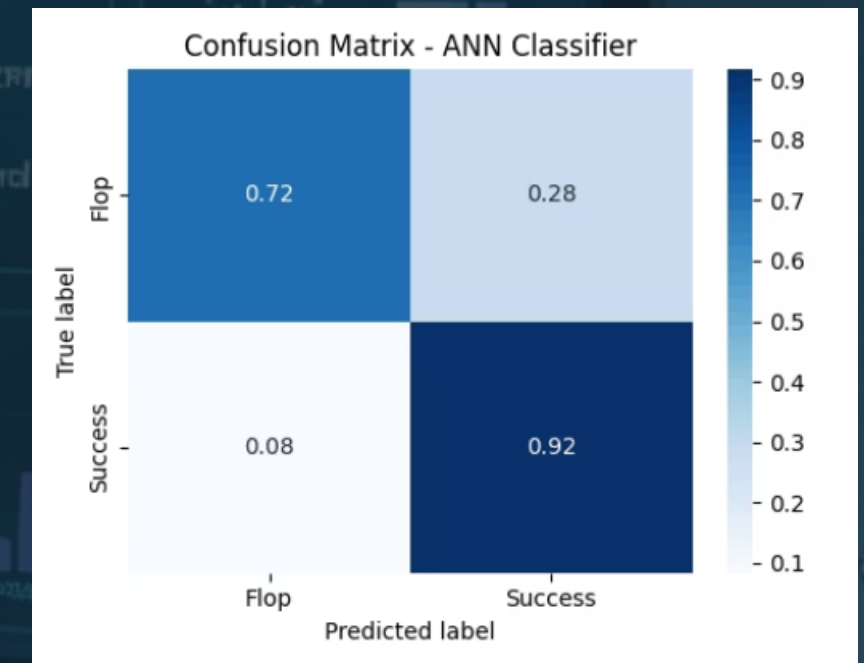
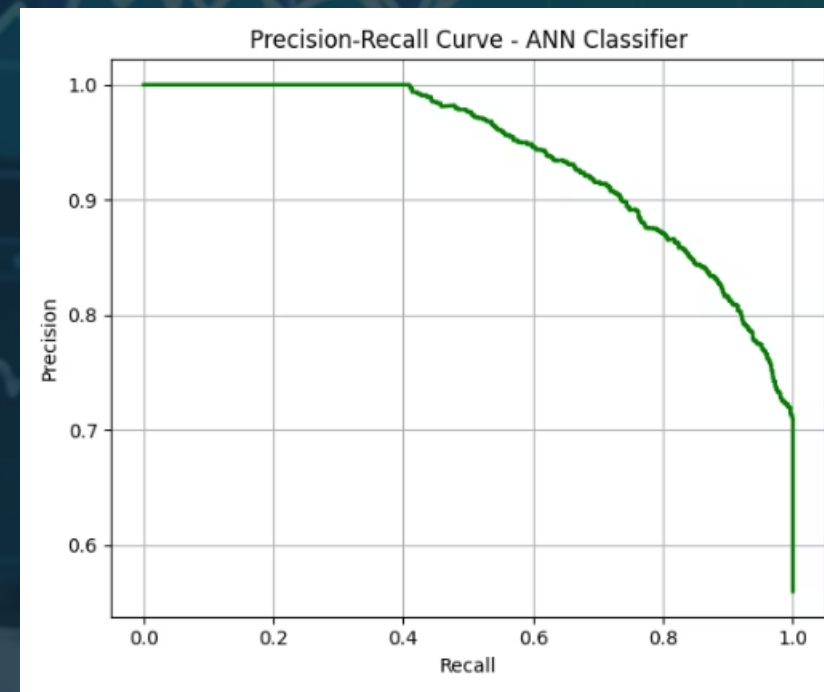
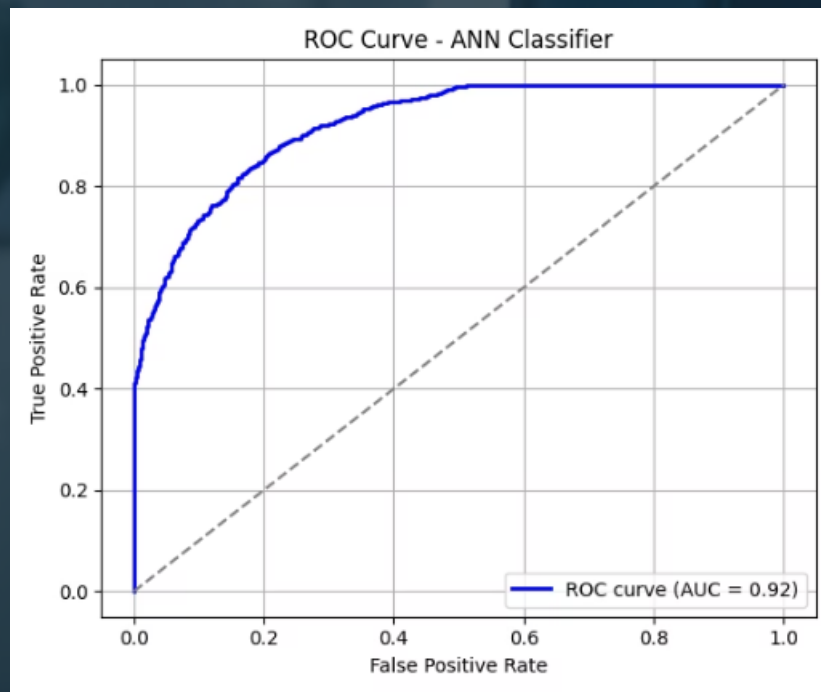
When model predicts success, it's right about 4/5 of the time

~91%

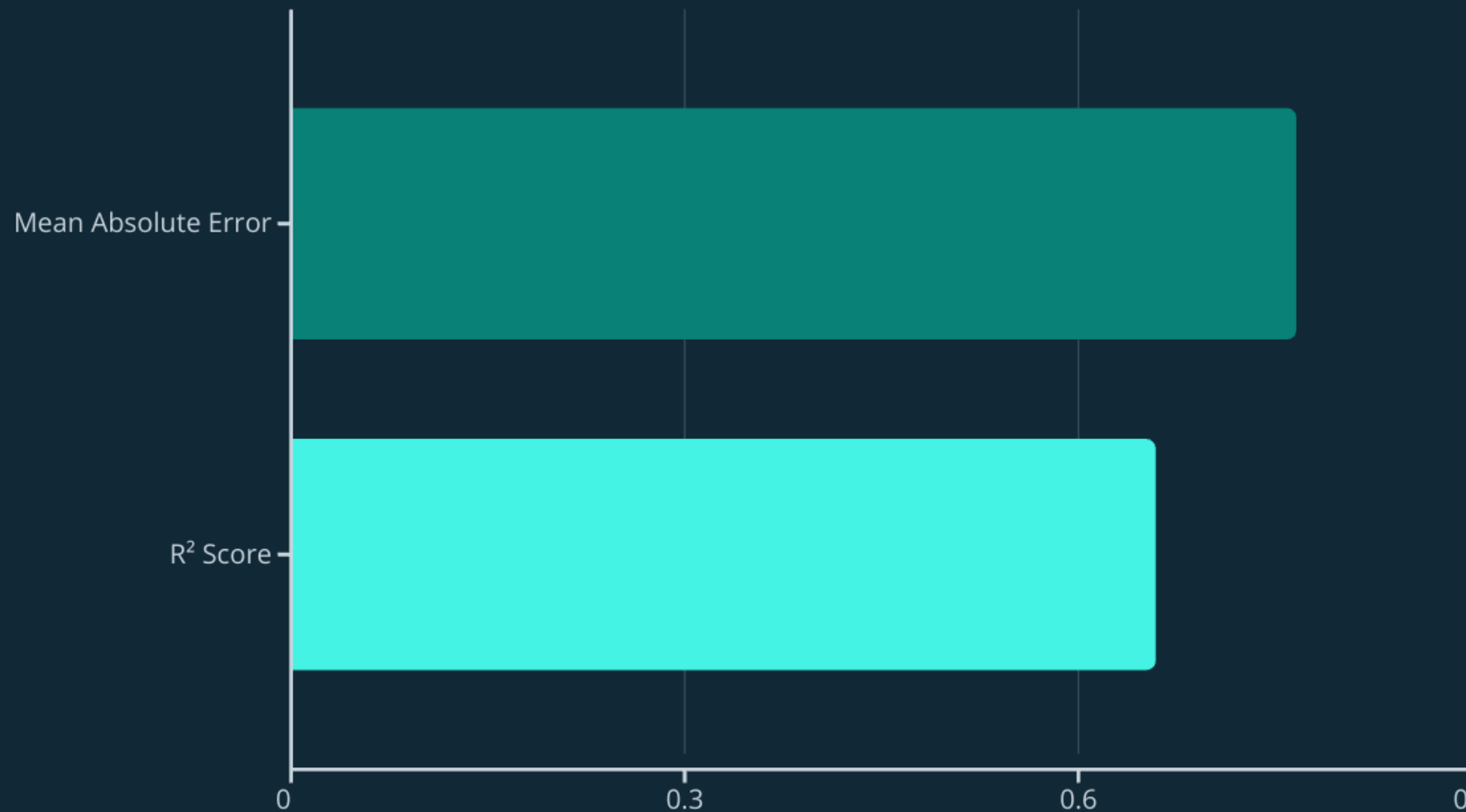
Recall

Model catches about 9/10 of the actual hits

Evaluation Plots



Revenue Prediction Performance



Our XGBoost regressor achieves a Mean Absolute Error of approximately \$33 million in revenue predictions. Considering some blockbusters earn hundreds of millions, this error is reasonably low.

The R² Score of ~0.66 indicates the model explains about 66% of the variance in movie revenues. While not perfect, this provides significant predictive power, especially for an industry with inherent unpredictability.

Challenges Encountered & Solutions



Data Quality Issues

Encountered incomplete values for budget and runtime. Applied strict data cleaning filters and statistical imputation techniques to ensure dataset integrity.



Distribution Skew

Blockbuster outliers created extreme revenue imbalance. Implemented log transformation and leveraged XGBoost's inherent ability to handle skewed distributions.



High Dimensionality

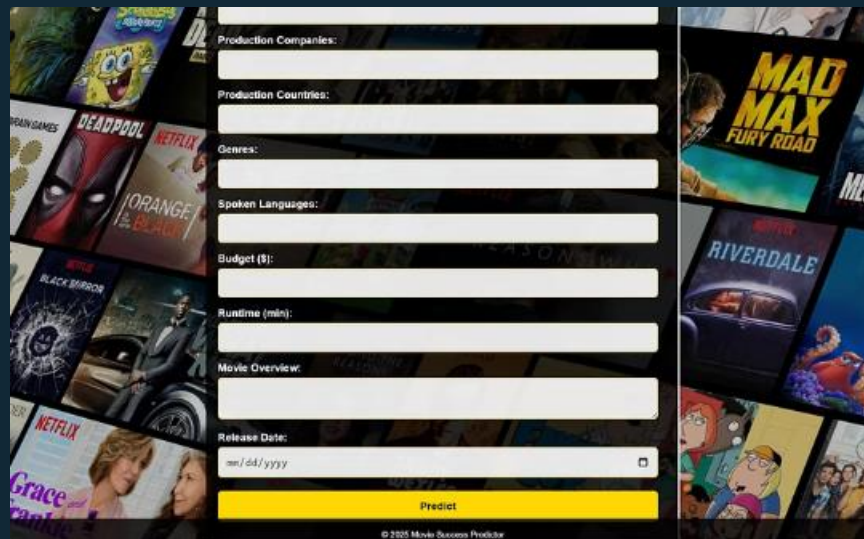
Feature explosion (~200 features) after encoding. Combined regularization techniques with models designed to handle high-dimensional spaces.



Model Optimization

Addressed class imbalance with stratified sampling and implemented careful pipeline separation to prevent data leakage between stages.

Deployment



The screenshot shows the input form of the web application. It features a dark background with movie posters on the sides. The form includes fields for Production Companies, Production Countries, Genres, Spoken Languages, Budget (\$), Runtime (min), Movie Overview, and Release Date (mm/dd/yyyy). A yellow 'Predict' button is at the bottom.

User Interface

A simple web form with fields for key inputs (cast, director, budget, etc.) that allows users to enter information about upcoming movies.

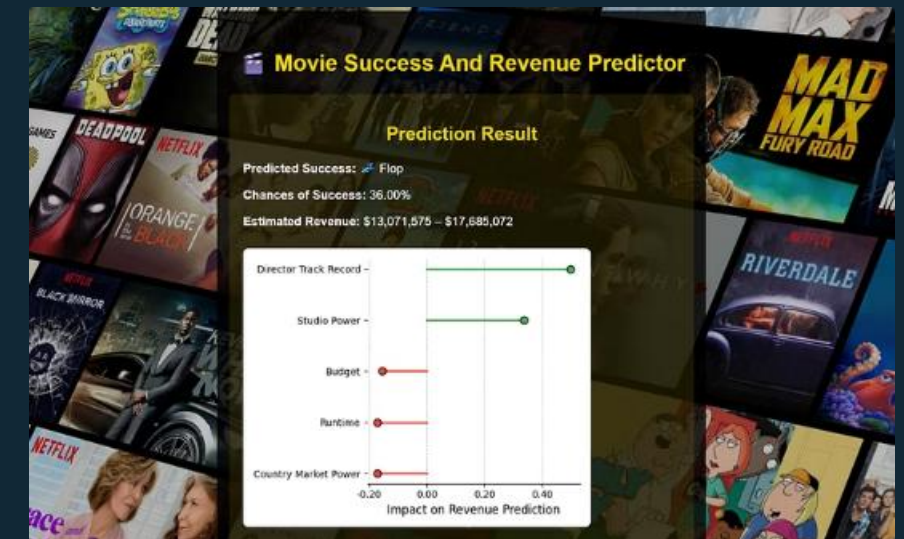


This screenshot shows the same form with data entered. The title is 'Movie Success And Revenue Predictor'. The input data is as follows:

Field	Value
Movie Title:	Drive-Away Dolls
Cast (comma-separated):	Margaret Qualley, Geraldine Viswanathan, Beanie Feldstein, Joey Slotnick
Director:	Ethan Coen
Writers:	Tricia Cooke, Tricia Cooke
Producers:	Tim Bevan, Ethan Coen, Tricia Cooke, Eric Felner
Production Companies:	Focus Features, Working Title Films
Production Countries:	United States of America, United Kingdom
Genres:	Comedy, Crime

Flask Web Application

RESTful API using Flask that processes user input, computes features, and calls models to generate predictions.



Prediction Results

The application returns success probability and estimated box office revenue to help inform production decisions.



Conclusion & Future Work

✓ Project Outcome

Developed a working ML system that can predict with decent accuracy whether a movie will be a hit or flop, and estimate its revenue before release.

💡 Impact

Such a model could serve as a preliminary "greenlighting" tool for studios – supporting budgeting decisions and marketing strategies by identifying high-risk projects early.

⏭ Future Work

Incorporate social media hype, genre trends, and actor popularity indices. Experiment with advanced models and keep the system updated with recent data as the film industry evolves.