



UNIVERSITÀ DI PISA



2nd Project Report

Giuseppe Muschetta

Master's Degree Program in Computer Science
Specialization in Data Science and Business Informatics

University of Pisa, Academic Year 2024/2025

Contents

1	Data Understanding and Preparation	1
1.1	Initial Dataset Assessment	1
1.2	Exploratory Data Analysis (EDA)	1
1.2.1	Raw Numerical Feature Distributions	1
1.2.2	Categorical Feature Distributions	2
1.3	Correlation and Redundancy Analysis	2
1.3.1	Categorical Feature Association (Cramér's V)	2
1.3.2	Numerical Feature Correlation (Pearson & Spearman)	3
1.4	Data Imputation	4
1.5	Feature Engineering	5
1.6	Final Data Transformation	5
2	Outlier Detection	6
2.1	Methodology	6
2.1.1	Isolation Forest	6
2.1.2	Local Outlier Factor (LOF)	6
2.1.3	Cluster-Based Local Outlier Factor (CBLOF)	6
2.2	Comparative Analysis of Results	6
2.3	Outlier Treatment and Qualitative Validation	7
3	Imbalanced Learning: A Methodological Investigation	8
3.1	Baseline Model: Decision Tree on Imbalanced Data	8
3.1.1	Baseline Model Performance	8
3.1.2	Baseline Feature Importance	8
3.2	Oversampling Approach: The SMOTE-Enhanced Model	8
3.2.1	SMOTE Model Performance	9
3.2.2	SMOTE Feature Importance	9
3.3	Undersampling Approach: The RandomUnderSampler Model	9
3.3.1	Undersampling Model Performance	9
3.3.2	Undersampling Feature Importance	9
3.4	Final Comparative Analysis and Verdict	9
3.4.1	In-depth Discussion	9
3.5	Final Verdict	10
4	Advanced Classification	10
4.1	Target Variable Definition	10
4.2	Methodology and Evaluation Framework	11
4.3	Linear Models: Logistic Regression and LinearSVC	11
4.3.1	Hyperparameter Tuning	11
4.3.2	Performance Evaluation	11
4.4	Ensemble Models: Random Forest and XGBoost	11
4.4.1	Hyperparameter Tuning	12
4.4.2	Feature Importance Analysis	13
4.5	Deep Neural Network (Weighted Softmax Classifier)	13
4.5.1	Architecture and Training Strategy	13
4.5.2	Performance Evaluation	13
4.6	Comparative Analysis and Final Conclusions	14
4.6.1	Conclusions	14
5	Explainability Random Forest Classifier	14
5.1	Global Feature Importance Analysis	14
5.2	Local Explainability: Case Study Analysis	15
5.3	Decision Pathway Analysis	16
5.4	Implications and Model Insights	16

6	Advanced Regression	17
6.0.1	Target Variable Selection	17
6.0.2	Feature Space Construction	17
6.0.3	Model Selection	17
6.0.4	Hyperparameter Optimization	17
6.1	Model Performance and Quantitative Evaluation	17
6.1.1	Evaluation Metrics	17
6.1.2	Results	18
6.1.3	Comparative Analysis	18
6.1.4	Diagnostic and Residual Analysis	18
6.2	Comparative Feature Importance Analysis	19
6.3	Conclusion	19
7	Time Series Data Understanding and Preparation	19
7.1	Statistical Analysis of Revenue Distribution	19
7.2	Revenue Feature Engineering and Correlation Analysis	20
7.2.1	Correlation Analysis	20
7.3	Global Pattern Discovery	20
7.3.1	Decay Patterns	21
7.4	Data Pre-processing and Representation	21
8	Motifs, Discords, and Shapelets: A Unified Temporal Analysis	22
8.1	Introduction and Methodology	22
8.2	Motif and Discord Discovery	22
8.2.1	Global Motif Analysis	22
8.2.2	Global Discord Analysis	22
8.3	Relationship with Shapelets	23
8.3.1	Hypothesis-Driven Shapelet Discovery	23
8.3.2	Learned Shapelets and Interpretation	23
8.3.3	Final Synthesis: Connecting Motifs and Shapelets	23
9	Clustering: K-Medoids and Hierarchical	24
9.1	K-Medoids (Partitioning Around Medoids - PAM)	24
9.2	Agglomerative Hierarchical Clustering	24
9.3	Optimal Number of Clusters (k)	24
9.4	Analysis and Visualization of the Cluster Partition	24
9.4.1	Dimensionality Reduction for Visualization	24
9.5	Characterization and Interpretation of the Discovered Clusters	25
9.5.1	Cluster 0 (312 films): The Front-Loaded Blockbuster	25
9.5.2	Cluster 1 (110 films): The High-Variance Performer	25
9.5.3	Cluster 2 (107 films): The Stable & Sustained Success	25
9.5.4	Cluster 3 (491 films): The Normative Long-Tail Performer	26
9.5.5	Cluster 4 (114 films): The Slow-Burn Niche Success	26
10	Time Series Classification	26
10.1	Evaluation Metrics	26
10.2	Model 1: K-Nearest Neighbors (k-NN) with Euclidean or Manhattan Distances	26
10.2.1	Performance Analysis	26
10.3	Model 2: K-Nearest Neighbors (k-NN) with Dynamic Time Warping (DTW)	27
10.3.1	Performance Analysis	27
10.4	Model 3: Shapelet-based Model	27
10.4.1	Performance and Interpretability Analysis	27
10.5	Model 4: MiniROCKET	29
10.5.1	Performance Analysis	29
10.6	Comparative Analysis and Discussion	29
10.7	Conclusion and Recommendations	29

1 Data Understanding and Preparation

The IMDb dataset analysis began with a broad statistical overview and visual exploration to detect hidden patterns. This guided thorough data cleaning, including redundancy checks, advanced imputation, feature engineering, and a multi-stage transformation pipeline.

1.1 Initial Dataset Assessment

The dataset contains 149,531 records and 32 features. The initial analysis separated numerical (Table 1) and categorical (Table 2) features.

The **numerical summary** revealed key traits shaping our analysis. Features like `numVotes` and `totalCredits` show extreme positive skew, with max values far above the 75th percentile, typical of popularity metrics. This skew complicates many ML algorithms, requiring robust normalization.

The **categorical summary** exposed quality issues. For example, `runtimeMinutes`, a numerical variable, was misclassified as `object` due to a non-standard placeholder (`\N`) in 26.7% of entries, blocking numeric type inference.

Missing data were notable: `countryOfOrigin` had 26.9% missing values encoded as `np.nan`, while features like `regions` shared the placeholder problem, demanding a consistent missing value strategy.

Multi-label fields such as `genres`, `soundMixes`, `countryOfOrigin`, and `regions` were loaded as raw strings instead of lists, complicating downstream multi-label processing. We developed parsing functions to convert them into usable formats.

Table 1: Statistical Summary of Raw Numerical Features.

	count	mean	std	min	25%	50%	75%	max
<code>startYear</code>	149531	1994.94	24.89	1878	1981	1999	2015	2024
<code>awardWins</code>	149531	0.31	2.91	0	0	0	0	446
<code>numVotes</code>	149531	1085.13	19999.73	5	12	28	107	2.95e+06
<code>totalImages</code>	149531	9.53	101.65	0	1	1	4	20586
<code>totalVideos</code>	149531	0.81	9.48	0	0	0	0	1123
<code>totalCredits</code>	149531	22.31	83.29	0	5	12	22	10649
<code>criticReviewsTotal</code>	149531	6.20	60.18	0	0	0	1	5543
<code>awardNominations...</code>	149531	0.73	5.86	0	0	0	0	569
<code>numRegions</code>	149531	10.45	23.95	0	1	2	8	338
<code>userReviewsTotal</code>	149531	11.23	157.65	0	0	1	4	13180
<code>companiesNumber</code>	149531	5.82	19.66	0	1	2	5	597
<code>averageRating</code>	149531	6.87	1.35	1.0	6.1	7.1	7.8	10.0
<code>writerCredits</code>	149531	2.00	2.47	0	0	1	3	51
<code>directorsCredits</code>	149531	0.95	0.80	0	1	1	1	58
<code>quotesTotal</code>	149531	1.07	7.03	0	0	0	0	1084

Table 2: Descriptive Summary of Categorical Features.

	count	unique	top	freq
<code>originalTitle</code>	149531	133101	Episode #1.1	532
<code>rating</code>	149531	10	(7, 8]	48555
<code>endYear</code>	149531	79	\N	143897
<code>runtimeMinutes</code>	149531	377	\N	40195
<code>titleType</code>	149531	10	movie	69279
<code>countryOfOrigin</code>	109544	2651	['US']	44964
<code>genres</code>	149531	1419	Comedy	12986
<code>regions</code>	149531	26136	['\N']	34823
<code>soundMixes</code>	149531	305	[]	96798

1.2 Exploratory Data Analysis (EDA)

1.2.1 Raw Numerical Feature Distributions

The distribution of raw numerical features confirms earlier stats. Figure 1 shows six features with distinct shapes: `totalImages` follows an exponential pattern, `totalCredits` a Pareto distribution, both with many low values and

extreme right outliers. This strong right-skew justified the normalization approach in Section 1.6. Other numerical features share similar patterns.

`runtimeMinutes` had 26.7% missing values imputed via `IterativeImputer`, preserving its unique hexamodal distribution.

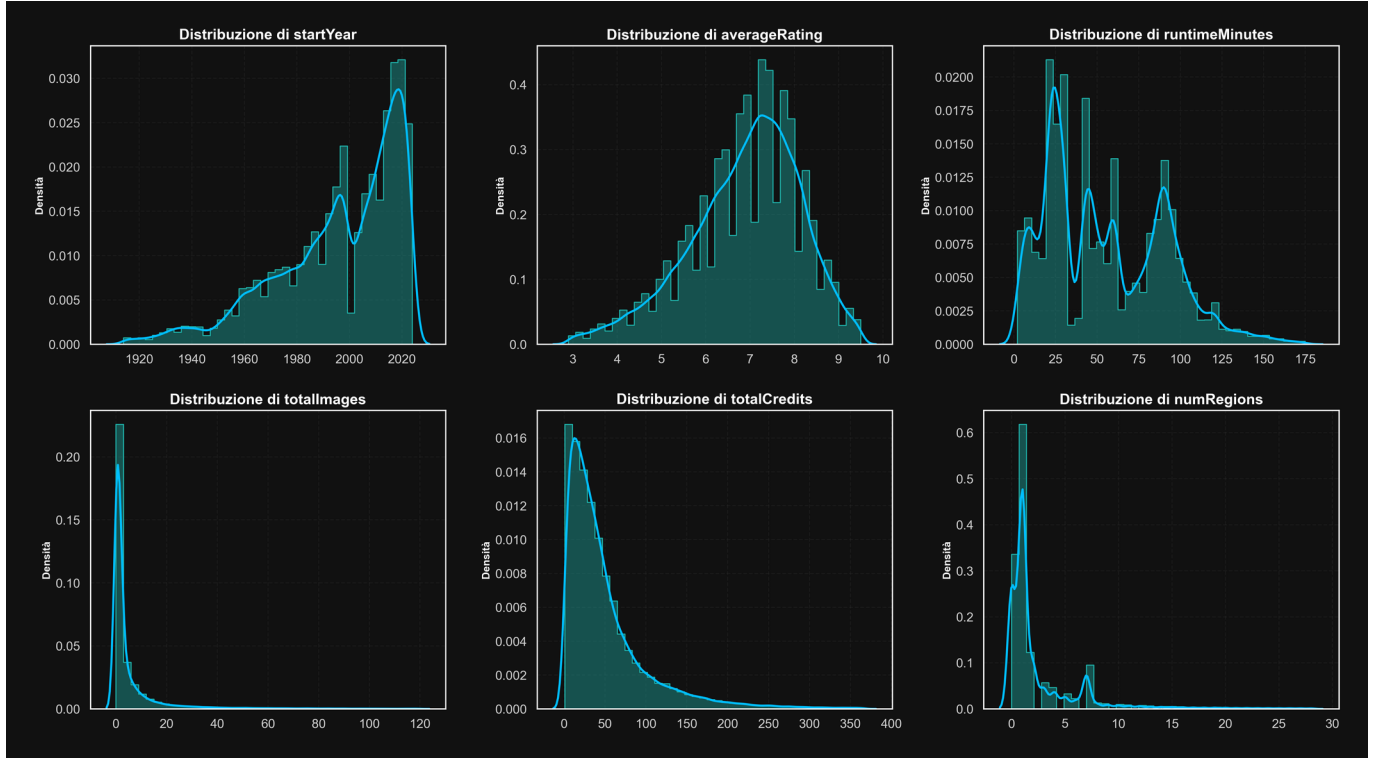


Figure 1: Six numeric variables with varied distributions, mostly right-skewed, necessitating normalization.

1.2.2 Categorical Feature Distributions

We first analyzed single-choice categorical variables, then complex multi-label attributes (Figure 2).

Among singles, `titleType` is dominated by `tvEpisode` and `movie`, reflecting many serialized entries. This feature was one-hot encoded. The `rating` variable is ordinal, with mode in `(7, 8]`, preserved by ordinal encoding.

Multi-label features (`genres`, `countryOfOrigin`, `regions`, `soundMixes`) were initially raw strings (e.g., `'Comedy,Drama'`). We parsed them into lists.

Analysis revealed strong class imbalance: *Drama* and *Comedy* dominate `genres`; *US* dominates `countryOfOrigin`. This warns of modeling bias towards majority classes. Hence, we applied **multi-label binarization**, creating binary indicators for each possible label, properly handling multiple categories per record.

1.3 Correlation and Redundancy Analysis

The EDA revealed potential overlaps between variables. To build an efficient model, we quantified these relationships to eliminate redundant features, central to our data cleaning via **feature elimination**.

1.3.1 Categorical Feature Association (Cramér's V)

We used Cramér's V, derived from the chi-squared test, to measure association between categorical variables on a 0-1 scale:

$$V = \sqrt{\frac{\chi^2/n}{\min(k-1, r-1)}}$$

with n observations, k columns, and r rows.

Figure 3 shows strong associations guiding feature removal:

- `isRatable`, `worstRating`, `bestRating` removed for zero variance (single value).
- `canHaveEpisodes` removed for perfect correlation (1.000) with `titleType`, making it redundant.
- `isAdult` removed due to high correlation (0.816) with `genres`, indicating redundancy.

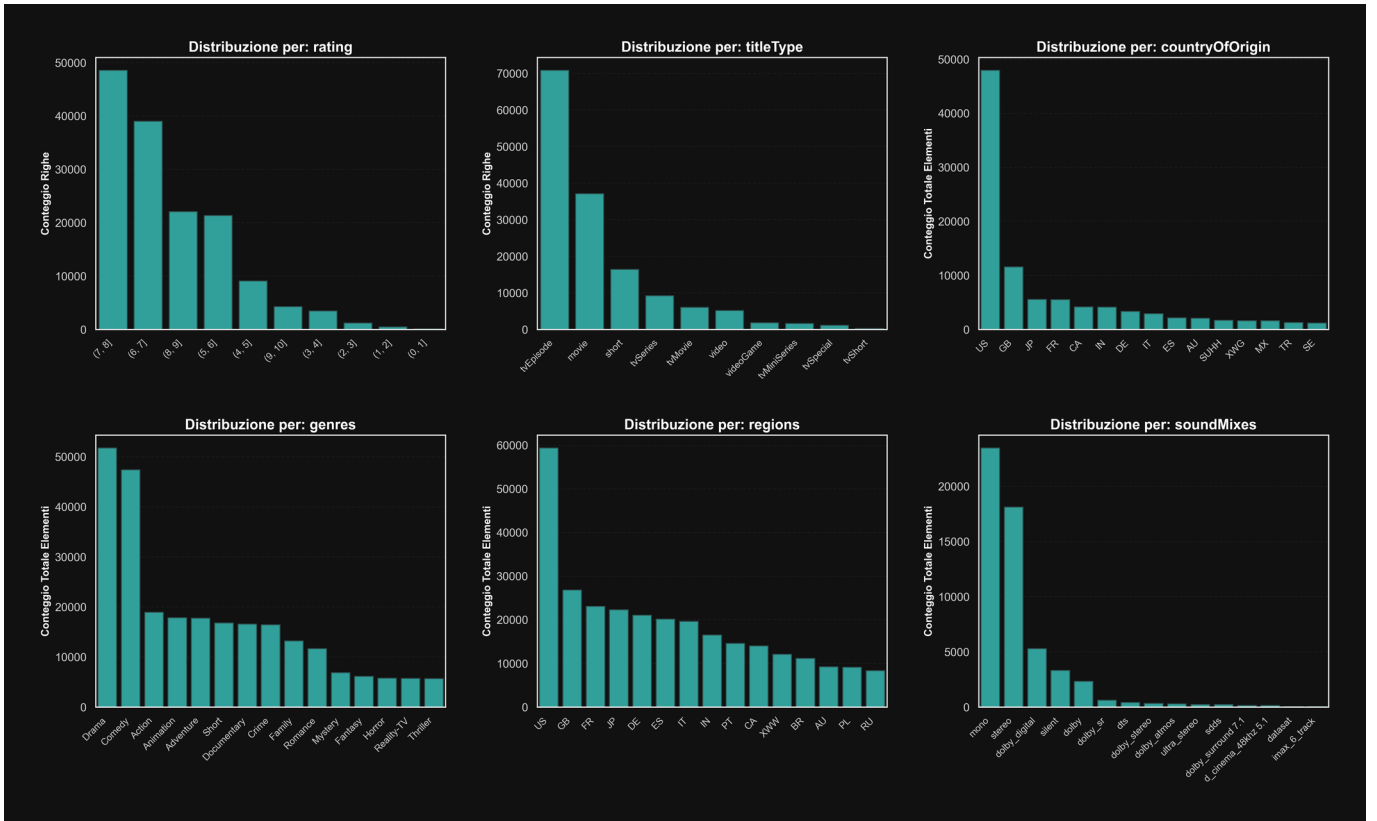


Figure 2: Distributions of categorical variables.

1.3.2 Numerical Feature Correlation (Pearson & Spearman)

Pearson (linear) and Spearman (monotonic) correlations identified redundant numerical features (Table 3). Key removals to reduce multicollinearity:

- `ratingCount` removed for perfect correlation (1.000) with `numVotes`.
- `externalLinks` removed for high correlation (0.932) with `criticReviewsTotal`; kept the latter for clarity.
- `castNumber` removed due to high correlation (0.832) with `totalCredits`; kept the latter for broader production scale.
- `rating_encoded` removed to avoid leakage, keeping continuous `averageRating` as primary rating.

Table 3: Top 12 Strongest Numerical Correlations (Pearson & Spearman)

(a) Pearson Correlations (Linear)

Variable Pair	Coeff. (r)
<code>numVotes</code> ↔ <code>ratingCount</code>	1.000
<code>rating_encoded</code> ↔ <code>averageRating</code>	0.976
<code>externalLinks</code> ↔ <code>criticReviewsTotal</code>	0.932
<code>totalCredits</code> ↔ <code>castNumber</code>	0.832
<code>ratingCount</code> ↔ <code>userReviewsTotal</code>	0.806
<code>numVotes</code> ↔ <code>userReviewsTotal</code>	0.806
<code>awardNominationsExcludeWins</code> ↔ <code>awardWins</code>	0.730
<code>criticReviewsTotal</code> ↔ <code>numRegions</code>	0.640
<code>numRegions</code> ↔ <code>externalLinks</code>	0.639
<code>userReviewsTotal</code> ↔ <code>criticReviewsTotal</code>	0.603
<code>ratingCount</code> ↔ <code>criticReviewsTotal</code>	0.592
<code>numVotes</code> ↔ <code>criticReviewsTotal</code>	0.592

(b) Spearman Correlations (Monotonic)

Variable Pair	Coeff. (ρ)
<code>numVotes</code> ↔ <code>ratingCount</code>	1.000
<code>rating_encoded</code> ↔ <code>averageRating</code>	0.969
<code>castNumber</code> ↔ <code>totalCredits</code>	0.738
<code>externalLinks</code> ↔ <code>criticReviewsTotal</code>	0.690
<code>userReviewsTotal</code> ↔ <code>numVotes</code>	0.660
<code>ratingCount</code> ↔ <code>userReviewsTotal</code>	0.660
<code>ratingCount</code> ↔ <code>criticReviewsTotal</code>	0.571
<code>criticReviewsTotal</code> ↔ <code>numVotes</code>	0.571
<code>awardWins</code> ↔ <code>awardNominationsExcludeWins</code>	0.567
<code>userReviewsTotal</code> ↔ <code>criticReviewsTotal</code>	0.544
<code>totalImages</code> ↔ <code>ratingCount</code>	0.535
<code>numVotes</code> ↔ <code>totalImages</code>	0.535

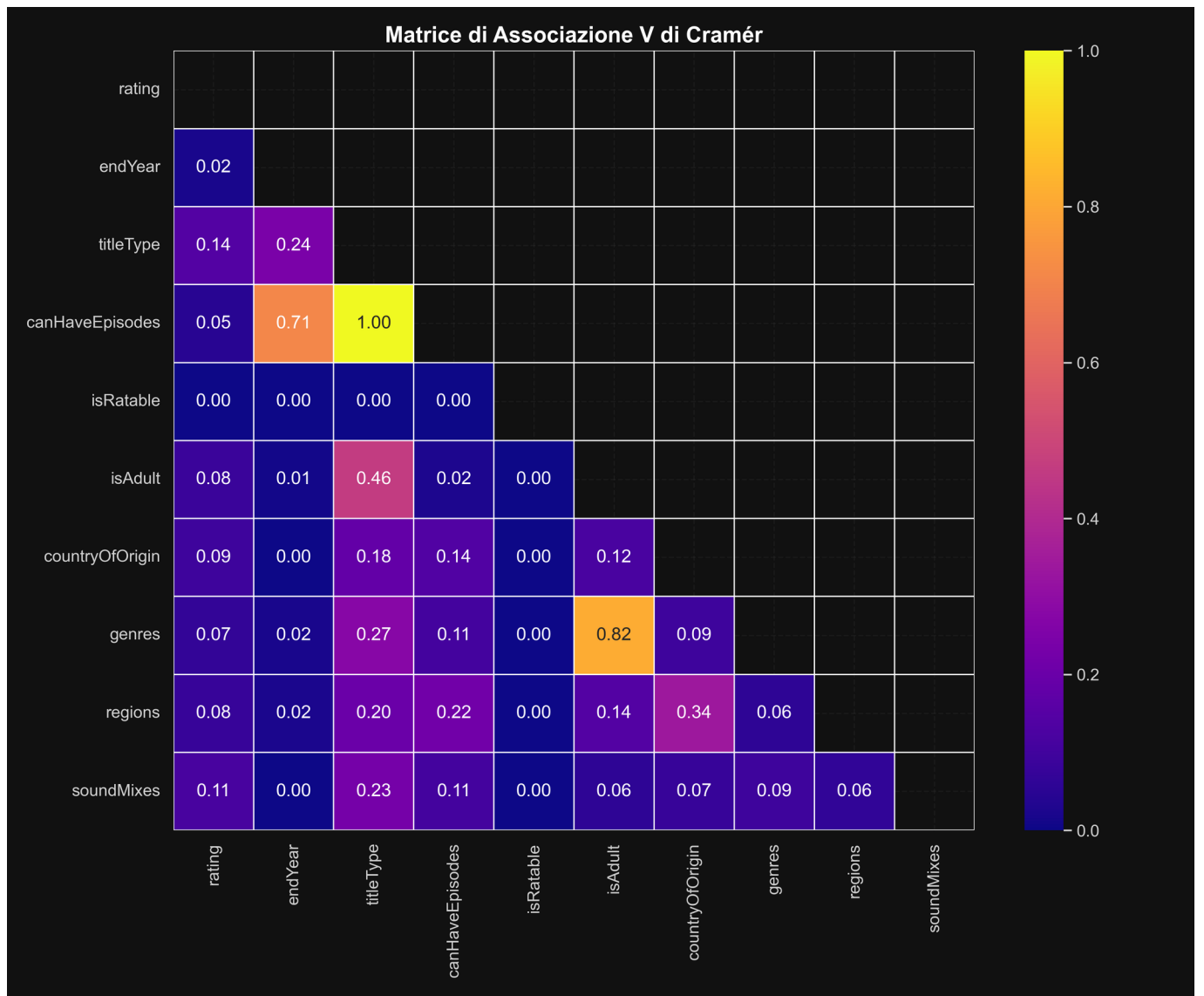


Figure 3: Cramér's V matrix for categorical features.

1.4 Data Imputation

After removing redundant features, we focused on imputing missing data. The `runtimeMinutes` variable had about 26.7% missing values, initially marked by `\N`. These were converted to `np.nan` and imputed using `IterativeImputer`, which estimates values via multivariate relationships.

Figure 4 shows the imputed `runtimeMinutes` retains its distinctive **hexamodal** distribution, reflecting common content lengths: short episodes (10 min), sitcoms (24 min), TV dramas (40 min), specials (60 min), standard films (90 min), and long films (120 min), confirming accurate, distribution-preserving imputation.

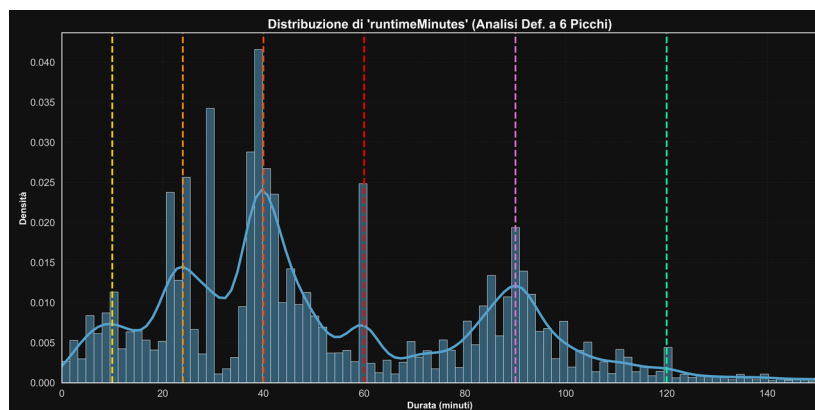


Figure 4: Hexamodal distribution of imputed `runtimeMinutes`.

For multi-label categorical features (`countryOfOrigin`, `regions`), missing values (`np.nan` and `\N`) were replaced by empty lists (`[]`) instead of statistical imputation to avoid bias.

During **multi-label binarization** (one-hot encoding for multi-labels), records with empty lists received zeros across all indicators, effectively representing "no known category" without adding artificial labels.

1.5 Feature Engineering

We created five new features to boost predictive power based on domain knowledge and EDA insights:

- **winRate**: Measures nomination-to-win efficiency, normalized [0,1], with safeguards for zero nominations:

$$\text{winRate} = \frac{\text{awardWins}}{\text{awardWins} + \text{awardNominationsExcludeWins}}$$

- **contentRichness**: Indicates entry completeness via the sum of metadata counts:

$$\text{contentRichness} = \text{quotesTotal} + \text{totalVideos} + \text{totalImages} + \text{totalCredits}$$

- **logPopularityScore**: Captures overall "buzz" using a log transform to stabilize variance. With:

$$P = \text{numVotes} \times (\text{userReviewsTotal} + \text{criticReviewsTotal})$$

the feature is:

$$\text{logPopularityScore} = \ln(1 + P)$$

- **criticUserRatio**: Highlights whether critics or users discuss a title more, using log ratios with zero-handling:

$$\text{criticUserRatio} = \frac{\ln(1 + \text{criticReviewsTotal})}{\ln(1 + \text{userReviewsTotal})}$$

- **hasWriterAndDirector**: Binary flag indicating if both writer and director credits exist:

$$\text{hasWriterAndDirector} = \begin{cases} 1 & \text{if } \text{writerCredits} > 0 \text{ and } \text{directorsCredits} > 0 \\ 0 & \text{otherwise} \end{cases}$$

1.6 Final Data Transformation

Initial attempts triggered **overflow** warnings due to extreme scale and skewness in features like `numVotes`. A robust three-stage pipeline was implemented to fix this:

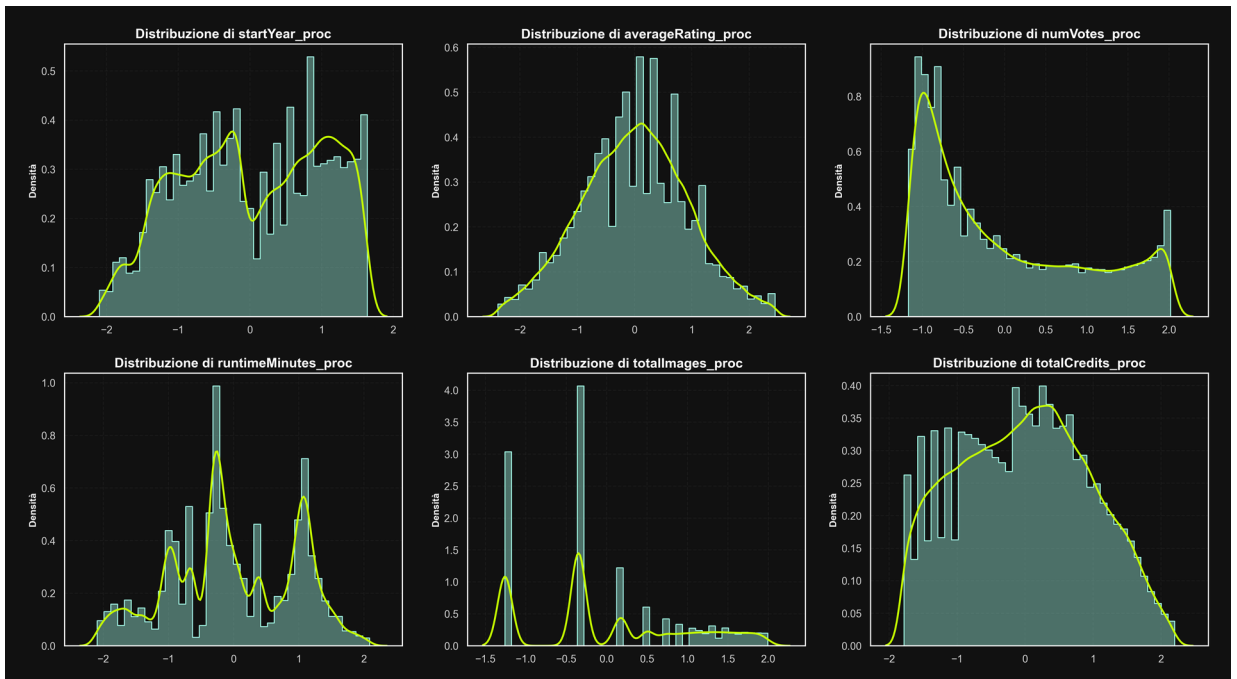


Figure 5: Significant numerical features after the 3-stage transformation, now symmetric and centered at zero.

1. **Robust Scaling:** Applied `RobustScaler` using median and IQR to handle outliers and compress data into a stable numeric range, preventing overflow.
2. **Yeo-Johnson Transformation:** Used `PowerTransformer` with Yeo-Johnson to optimally reshape distributions to Gaussian-like forms, correcting skewness and handling zeros or negatives.
3. **Standardization:** Final `StandardScaler` step centered data to mean 0 and std 1, ensuring uniform scale for ML algorithms.

New processed columns with suffix `_proc` were created, preserving originals. Figure 5 demonstrates the pipeline's success in producing symmetric, normalized features ready for modeling.

2 Outlier Detection

This module aimed to detect and handle anomalous observations in the dataset. Outliers: data points significantly different from others that can skew analyses like clustering and classification, so robust detection is essential for model reliability.

We used a multi-perspective approach with three algorithms from different families. A **consensus strategy** flagged a point as an outlier only if at least two methods agreed, reducing bias and increasing confidence. The goal was to find the top 1% of outliers based on scoring.

2.1 Methodology

We analyzed 21 processed numerical features (`'_proc'` columns), scaled and transformed during preparation, using three diverse algorithms.

2.1.1 Isolation Forest

An ensemble method based on the idea that anomalies are rare and distinct, thus easier to isolate. It builds random trees splitting on random features and values. Anomaly score depends on average path length, the shorter, the more anomalous.

2.1.2 Local Outlier Factor (LOF)

A density-based method evaluating anomaly by comparing a point's local density to neighbors'. It excels in datasets with varying densities, like IMDb with blockbusters and niche films. Points with much lower density than neighbors score high.

2.1.3 Cluster-Based Local Outlier Factor (CBLOF)

Identifies outliers based on cluster structure. Data is clustered into large and small groups. A point's anomaly score depends on its cluster size and distance to the nearest large cluster centroid. Points in small, distant clusters are strong outliers.

2.2 Comparative Analysis of Results

All models were set with `contamination = 0.01`, flagging roughly 1496 outliers each.

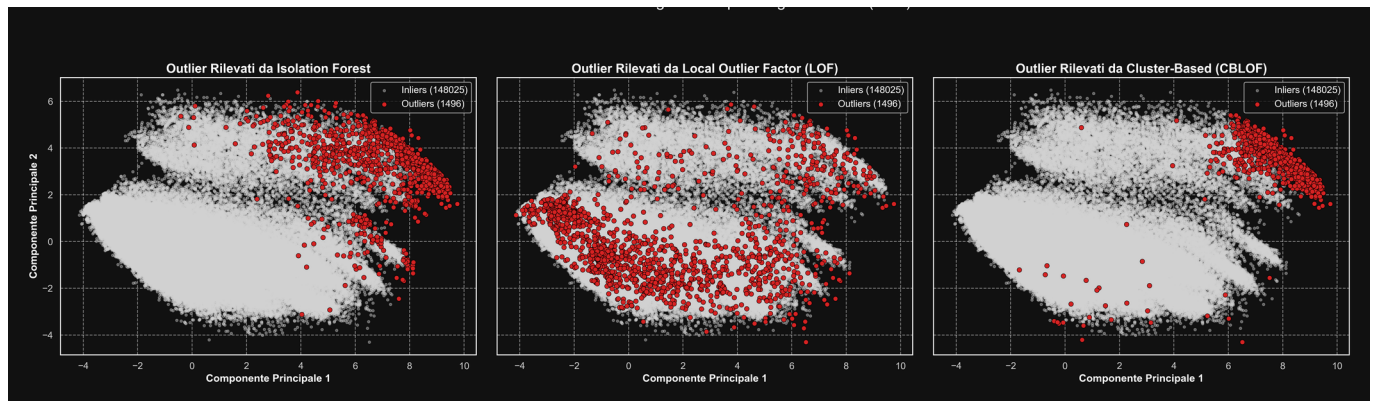


Figure 6: Dashboard comparing outliers detected by each method in 2D PCA space.

Outliers mostly cluster in the **upper-right quadrant** of PCA space, indicating a shared driving factor.

Consensus results: 2227 points flagged by one method, 1006 by two, 83 by all three. Using ≥ 2 agreements, **1089 outliers** were selected.

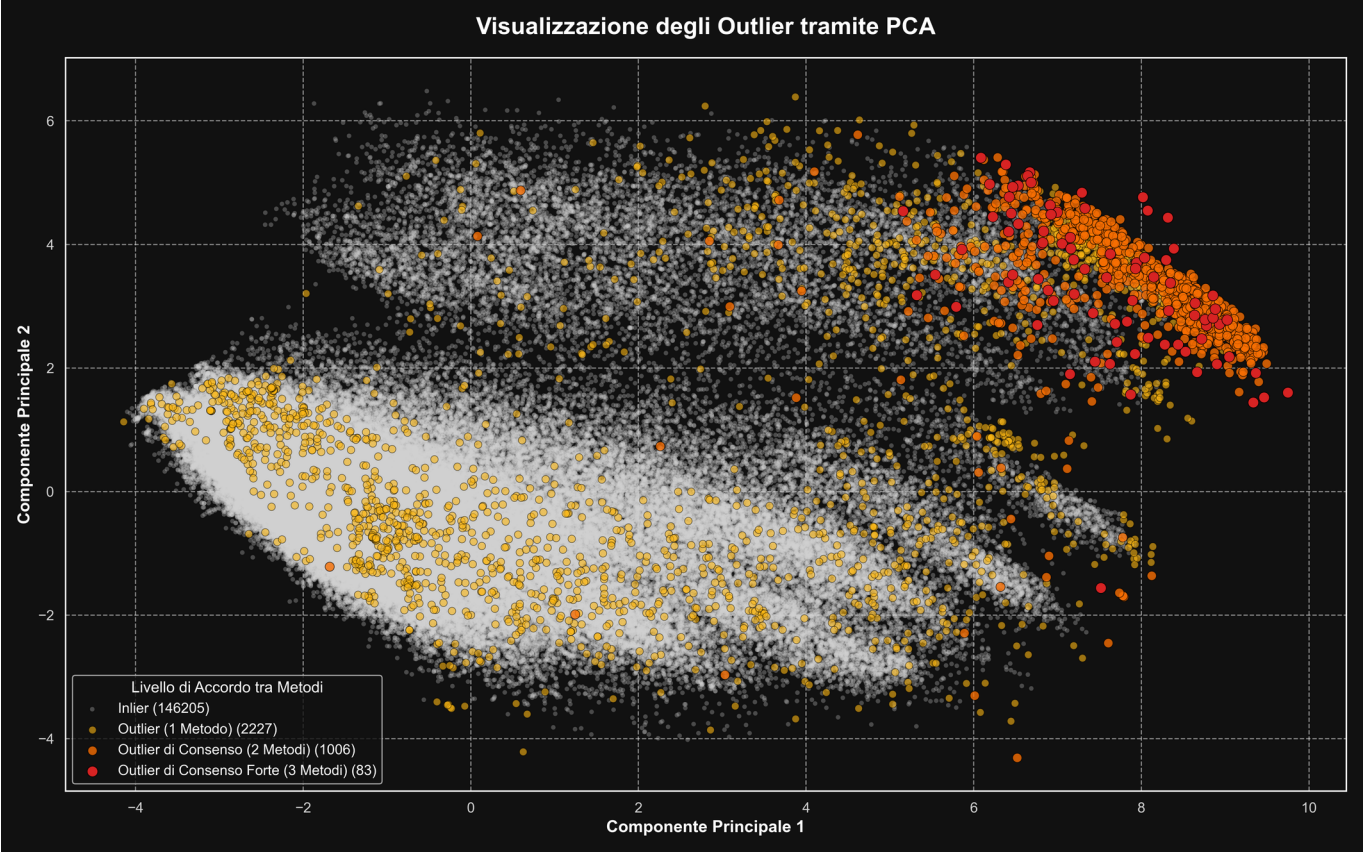


Figure 7: Outliers colored by agreement level among methods.

High-consensus outliers cluster in the upper-right “arm,” distinct from main data, suggesting shared anomalous traits.

2.3 Outlier Treatment and Qualitative Validation

The 1089 consensus outliers constitute only **0.73%** of the data; thus, removing them was chosen to prevent skewing models while preserving most information.

Qualitative checks on sample outliers reveal a consistent pattern: these instances belong to popular, long-running TV series characterized by extensive production crews and exceptionally high user engagement. As illustrated in Table 4, the three most extreme outliers by anomaly score include an episode from *Sons of Anarchy*, one from *Family Guy*, and another from the sitcom *Scrubs*. This alignment reinforces the notion that the outlier detection algorithm is capturing episodes that **deviate significantly** in structure or popularity due to their franchise-driven dynamics, likely tied to exceptional viewership spikes, cult followings, or unique narrative arcs within their respective series.

Table 4: Sample analysis of consensus outliers.

Original Title	Anomalous Feature	Value	Dataset Mean
<i>Sons of Anarchy</i>	Number of Votes	330,347	~ 1,085
	Total Credits	1,734	~ 58
<i>Family Guy</i>	Number of Votes	369,380	~ 1,085
	Total Images	3,501	~ 10
<i>Scrubs</i>	Number of Votes	273,420	~ 1,085
	Total Credits	1,853	~ 58

These are genuine outliers, **not** data errors. Removing them ensures modeling focuses on typical titles, not skewed by the very extremes. The cleaned dataset now has **148,432** records.

3 Imbalanced Learning: A Methodological Investigation

This module tackles a common machine learning challenge: classification on **imbalanced datasets**. The goal was to build and evaluate a model to identify documentaries (`isDocumentary`), with a class ratio of **96% non-documentaries** to **4% documentaries**. This controlled setup enables a rigorous comparison of techniques addressing class imbalance.

We used a **Decision Tree** classifier with `max_depth=10` for consistent evaluation through three stages:

1. Baseline training on raw imbalanced data.
2. Oversampling minority class via **SMOTE**.
3. Undersampling majority class via **RandomUnderSampler**.

Evaluation focused on confusion matrices and especially the **Precision-Recall (P-R) curve**, which better reflects minority class performance under severe imbalance than ROC curves.

3.1 Baseline Model: Decision Tree on Imbalanced Data

Training the Decision Tree on the 96/4 dataset established a baseline, highlighting the problem's difficulty and serving as a benchmark for resampling methods.

3.1.1 Baseline Model Performance

Table 5 shows the baseline's **deceptive 96% accuracy**, a figure that superficially suggests strong performance but is in fact largely driven by the model's tendency to default to the dominant class. In highly imbalanced classification scenarios, such accuracy becomes practically meaningless: a classifier can achieve extremely high accuracy simply by predicting the majority class in all cases, without learning any discriminative structure. In this context, such a model would appear "correct" 96% of the time while completely ignoring minority categories, thus failing at the very task of classification.

This issue is particularly evident in the baseline's **18% recall for documentaries**, meaning it fails to identify over 80% of the instances belonging to the minority class. Such behavior renders it unusable in practical scenarios that require detecting rare but semantically crucial instances, such as niche content or underrepresented genres.

The P-R curve (third panel, Figure 8) further exposes the flaw: precision drops drastically with minimal increases in recall. The model maintains precision above 0.6 only when recall remains below 0.2—an untenable trade-off. The resulting low AUC-PR and a Macro F1-score of 0.27 confirm the model's inability to simultaneously ensure detection and confidence, especially for minority classes.

3.1.2 Baseline Feature Importance

The baseline model's feature importance reveals a simplistic logic, dominated by broad rules due to the overwhelming majority of non-documentaries. Key predictors included: `startYear_proc`, `genres_drama`, `writerCredits_proc`, `genres_comedy`, and `type_movie`.

Reliance on general genre categories and production year indicates the model learned crude heuristics (e.g., "dramas from certain years are unlikely documentaries") rather than subtle documentary-specific traits.

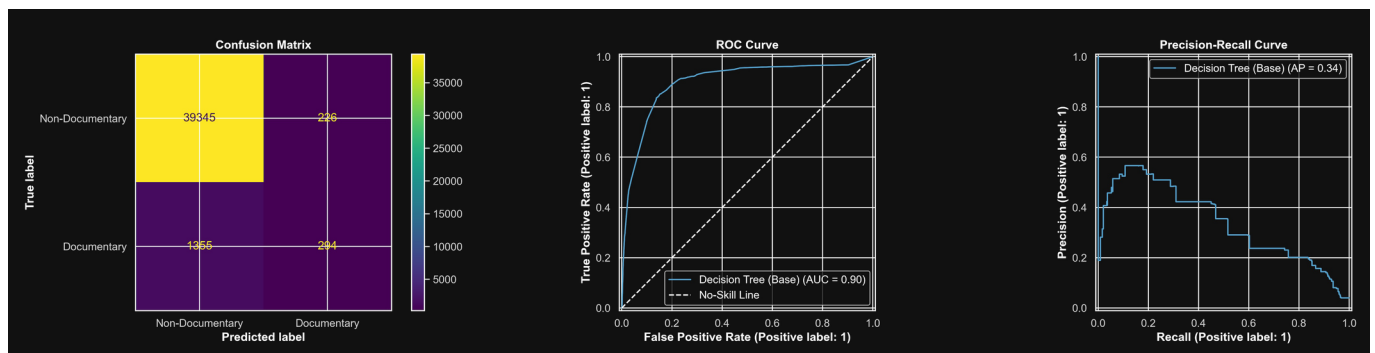


Figure 8: Baseline Model Performance: Confusion Matrix, ROC, and Precision-Recall Curves.

3.2 Oversampling Approach: The SMOTE-Enhanced Model

To address the severe imbalance, we applied SMOTE on training data, synthesizing new minority samples by interpolating between existing ones. This produced a balanced 50/50 training set, improving minority representation without simple duplication.

3.2.1 SMOTE Model Performance

SMOTE dramatically boosted minority class recall from 0.18 to 0.87 (Figure 9). This success comes with precision decreasing from 0.57 to 0.22, but the overall F1-score rose from 0.27 to 0.36, indicating a better balance.

The Precision-Recall curve (Figure 9) shows a marked improvement, maintaining higher precision across recall levels. The model learned a more refined decision boundary, no longer forced into a harsh trade-off, as reflected by the larger area under the curve (Average Precision 0.22), a substantial improvement over baseline.

3.2.2 SMOTE Feature Importance

Exposure to synthetic minority data shifted feature importance significantly. The model prioritized `writerCredits_proc`, `genres_comedy`, `genres_drama`, `genres_action`, and `genres_animation`.

Notably, `writerCredits_proc` emerged as the top feature, suggesting the model moved beyond simple genre or year cues, recognizing production metadata, likely reflecting distinct crediting or team structures, as a key documentary identifier. SMOTE facilitated uncovering this deeper pattern.

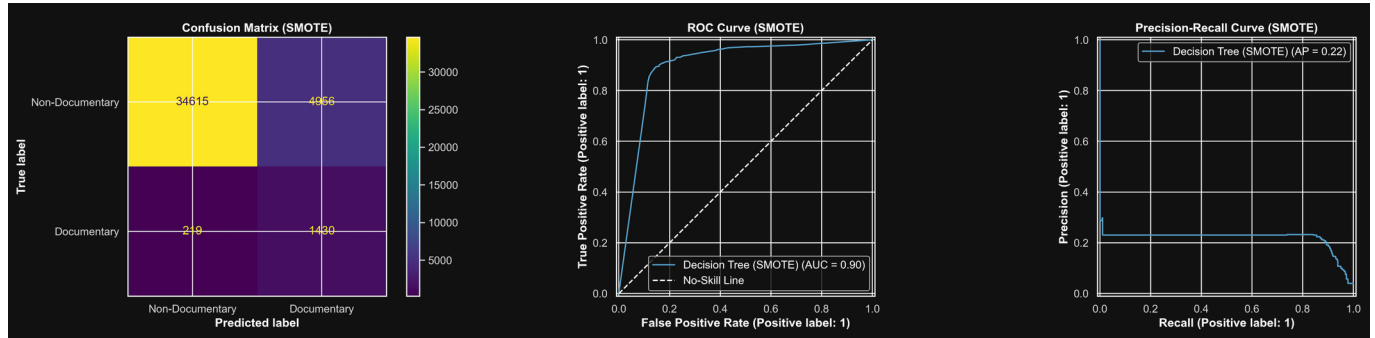


Figure 9: Performance evaluation of the SMOTE-enhanced model. Note the substantial lift in the Precision-Recall curve, indicating superior performance on the minority class.

3.3 Undersampling Approach: The RandomUnderSampler Model

As an alternative to SMOTE, we applied Random Undersampling, which randomly removes majority class samples until achieving a balanced 50/50 split. While efficient, this risks losing valuable information.

3.3.1 Undersampling Model Performance

Figure 10 shows this model attained the highest recall (**0.88**), excelling at detecting the minority class. However, precision dropped sharply to **0.18**, resulting in an F1-score of 0.30, better than baseline but notably worse than SMOTE.

The Precision-Recall curve reveals a hollow victory: despite high recall, precision is catastrophically low across thresholds, with Average Precision (0.19) even below SMOTE's. The flat, depressed curve indicates predictions are flooded with false positives, showing that information loss from undersampling cripples reliable detection.

3.3.2 Undersampling Feature Importance

Feature importance reflects this loss: the model resorts to crude, genre-based rules—`genres_comedy`, `genres_drama`, `genres_action`, `genres_animation`, and `startYear_proc`.

Discarding most data, the model loses nuance gained with SMOTE, relying on broad heuristics like “if not comedy, drama, action, or animation, then documentary,” resulting in a far less sophisticated model.

3.4 Final Comparative Analysis and Verdict

Table 5 consolidates results from all three models, enabling a clear comparison and final evaluation of each approach's strengths and weaknesses.

3.4.1 In-depth Discussion

These experiments clearly show the essential need for resampling techniques with imbalanced data:

1. **The Naive Baseline Fails:** Table 5 shows the baseline model is a "lazy classifier" with abysmal recall, making it unsuitable where false negatives are costly.
2. **Recall Dramatically Improved:** Both SMOTE and RandomUnderSampler boosted recall from 18% to nearly 90%, forcing the model to learn minority class traits.

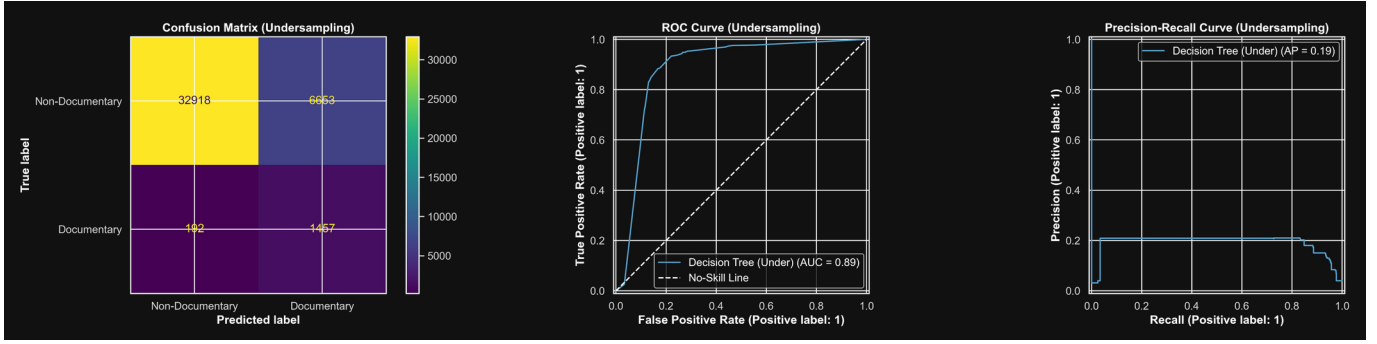


Figure 10: Random Undersampling Model Performance. High recall contrasts with low Precision-Recall area, indicating poor precision.

Table 5: Detailed Comparison of Classification Reports Across All Models. The metrics for the target class, **Documentary (1)**, are the primary focus of this analysis. The best F1-score for the minority class is highlighted.

Class / Metric	Baseline Model			SMOTE Model			Undersampling Model		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Non-Documentary (0)	0.97	0.99	0.98	0.99	0.87	0.93	0.99	0.83	0.91
Documentary (1)	0.57	0.18	0.27	0.22	0.87	0.36	0.18	0.88	0.30
Accuracy	0.96			0.87			0.83		
Macro Avg	0.77	0.59	0.63	0.61	0.87	0.64	0.59	0.86	0.60
Weighted Avg	0.95	0.96	0.95	0.96	0.87	0.91	0.96	0.83	0.88

- Feature Importance Reflects Model Sophistication:** Resampling shifts model focus. Baseline used simple genre/year rules; SMOTE unlocked a nuanced model emphasizing `writerCredits_proc`; undersampling caused regression to crude genre-only heuristics. Resampling choice directly affects model complexity.
- F1-Score as Final Arbiter:** Both methods sacrifice precision, with undersampling suffering most. The **SMOTE model's highest F1-score (0.36)** shows it best balances recall and precision.
- Synthesis Beats Destruction:** SMOTE's strategy of *synthesizing* new data uncovers deeper patterns, whereas undersampling's *information destruction* leads to a simplistic, less accurate model.

3.5 Final Verdict

In our scenario, characterized by a severe class imbalance (96% vs. 4%), **SMOTE outperformed random undersampling**. While undersampling may improve recall by aggressively reducing the dominant class, it does so at the expense of discarding valuable data, resulting in underfitting and a loss of generalization capability.

In contrast, SMOTE preserves the full information content while synthetically enriching the minority class, enabling the classifier to learn more robust and discriminative patterns. Therefore, although undersampling could be justified in contexts where maximizing recall for documentaries is the sole objective, SMOTE provides a smarter, more balanced solution, achieving better precision-recall trade-offs and a deeper, more reliable understanding of the data structure.

4 Advanced Classification

This section details the core predictive modeling phase focused on a granular, **10-class** classification task to predict a movie's rating from metadata. The choice to **avoid class aggregation** respects the original data structure, making the problem more realistic and challenging. We apply and evaluate sophisticated classifiers: Logistic Regression, Linear Support Vector Classifiers, tree-based ensembles (Random Forest and XGBoost), and a Deep Neural Network. The process emphasizes thorough hyperparameter tuning, comprehensive evaluation, and detailed comparative analysis to assess performance and limitations.

4.1 Target Variable Definition

We deliberately use the original **ordinal categorical rating** variable with ten intervals `'(0, 1]'` to `'(9, 10]'` as target classes. This grants:

- Maximum Granularity:** Models must distinguish nuanced rating levels, enabling precise predictions.
- Data Fidelity:** Avoids bias from grouping, allowing learning from raw data.

However, this increases task complexity and worsens class imbalance, with extreme ratings (`'1'` or `'10'`) much rarer than mid-range (`'7'` or `'8'`). This imbalance remains a key challenge.

4.2 Methodology and Evaluation Framework

Each classifier was trained and evaluated using a standardized pipeline. Hyperparameter tuning employed the use of `GridSearchCV`. The main metric optimized was the **macro-averaged F1-score**, chosen for robustness under imbalance, averaging per-class F1-scores without weighting. Final model performance was assessed on a stratified 30% test split with a broad evaluation suite.

4.3 Linear Models: Logistic Regression and LinearSVC

We start with two linear models as performance baselines. Logistic Regression used the `saga` solver with L2 regularization for stability. `LinearSVC`, efficient on high-dimensional data, was calibrated with `CalibratedClassifierCV` for probability outputs.

4.3.1 Hyperparameter Tuning

Both models tuned the regularization parameter `C`, with lower values controlling overfitting best:

- **Logistic Regression** — Best: `C=0.1`, L2 penalty, `saga` solver (CV macro F1: 0.1421)
- **LinearSVC** — Best: `C=0.01` (CV macro F1: 0.1591)

4.3.2 Performance Evaluation

The 10-class task proved difficult for linear models. Cross-validation favored `LinearSVC` (Macro F1-Score = 0.1591) over `Logistic Regression` (Macro F1-Score 0.1421). On the test set, `Logistic Regression` slightly outperformed (0.15 vs 0.14), indicating possible SVC overfitting or sensitivity to imbalance and calibration noise.

Classification reports (Table 6) show `LinearSVC` collapsed on the three rarest classes (zero precision and recall), while `Logistic Regression` retained nonzero recall on all classes but with very low precision and many false positives. This confirms the data is not linearly separable and that linear models cannot model the complex, imbalanced class structure here.

Table 6: Comparative Classification Report for Linear Models

Logistic Regression					LinearSVC (Calibrated)				
Target	Precision	Recall	F1-Score	Support	Target	Precision	Recall	F1-Score	Support
(0, 1]	0.00	0.35	0.01	26	(0, 1]	0.00	0.00	0.00	26
(1, 2]	0.01	0.34	0.02	142	(1, 2]	0.00	0.00	0.00	142
(2, 3]	0.02	0.19	0.03	353	(2, 3]	0.00	0.00	0.00	353
(3, 4]	0.08	0.20	0.11	1029	(3, 4]	0.31	0.01	0.02	1029
(4, 5]	0.18	0.22	0.20	2718	(4, 5]	0.35	0.03	0.06	2718
(5, 6]	0.32	0.19	0.24	6364	(5, 6]	0.33	0.20	0.25	6364
(6, 7]	0.39	0.15	0.21	11618	(6, 7]	0.33	0.39	0.36	11618
(7, 8]	0.52	0.30	0.38	14440	(7, 8]	0.42	0.74	0.54	14440
(8, 9]	0.24	0.20	0.21	6559	(8, 9]	0.31	0.06	0.09	6559
(9, 10]	0.08	0.35	0.13	1281	(9, 10]	0.32	0.02	0.04	1281

4.4 Ensemble Models: Random Forest and XGBoost

Introduction: We next employ two powerful, non-linear ensemble methods. `Random Forest` operates by averaging decorrelated decision trees, while `XGBoost` builds trees sequentially, with each new tree correcting the errors of its predecessor.

Table 7: Comparative Classification Report for Ensemble Models

Random Forest					XGBoost				
Target	Precision	Recall	F1-Score	Support	Target	Precision	Recall	F1-Score	Support
(0, 1]	0.45	0.38	0.42	26	(0, 1]	0.30	0.27	0.29	26
(1, 2]	0.53	0.18	0.26	142	(1, 2]	0.36	0.21	0.27	142
(2, 3]	0.35	0.12	0.18	353	(2, 3]	0.24	0.15	0.19	353
(3, 4]	0.35	0.16	0.22	1029	(3, 4]	0.23	0.22	0.23	1029
(4, 5]	0.36	0.28	0.32	2718	(4, 5]	0.30	0.37	0.33	2718
(5, 6]	0.37	0.43	0.40	6364	(5, 6]	0.37	0.43	0.40	6364
(6, 7]	0.46	0.44	0.45	11618	(6, 7]	0.49	0.40	0.44	11618
(7, 8]	0.55	0.61	0.58	14440	(7, 8]	0.59	0.53	0.56	14440
(8, 9]	0.44	0.43	0.44	6559	(8, 9]	0.42	0.54	0.47	6559
(9, 10]	0.50	0.35	0.41	1281	(9, 10]	0.42	0.50	0.45	1281

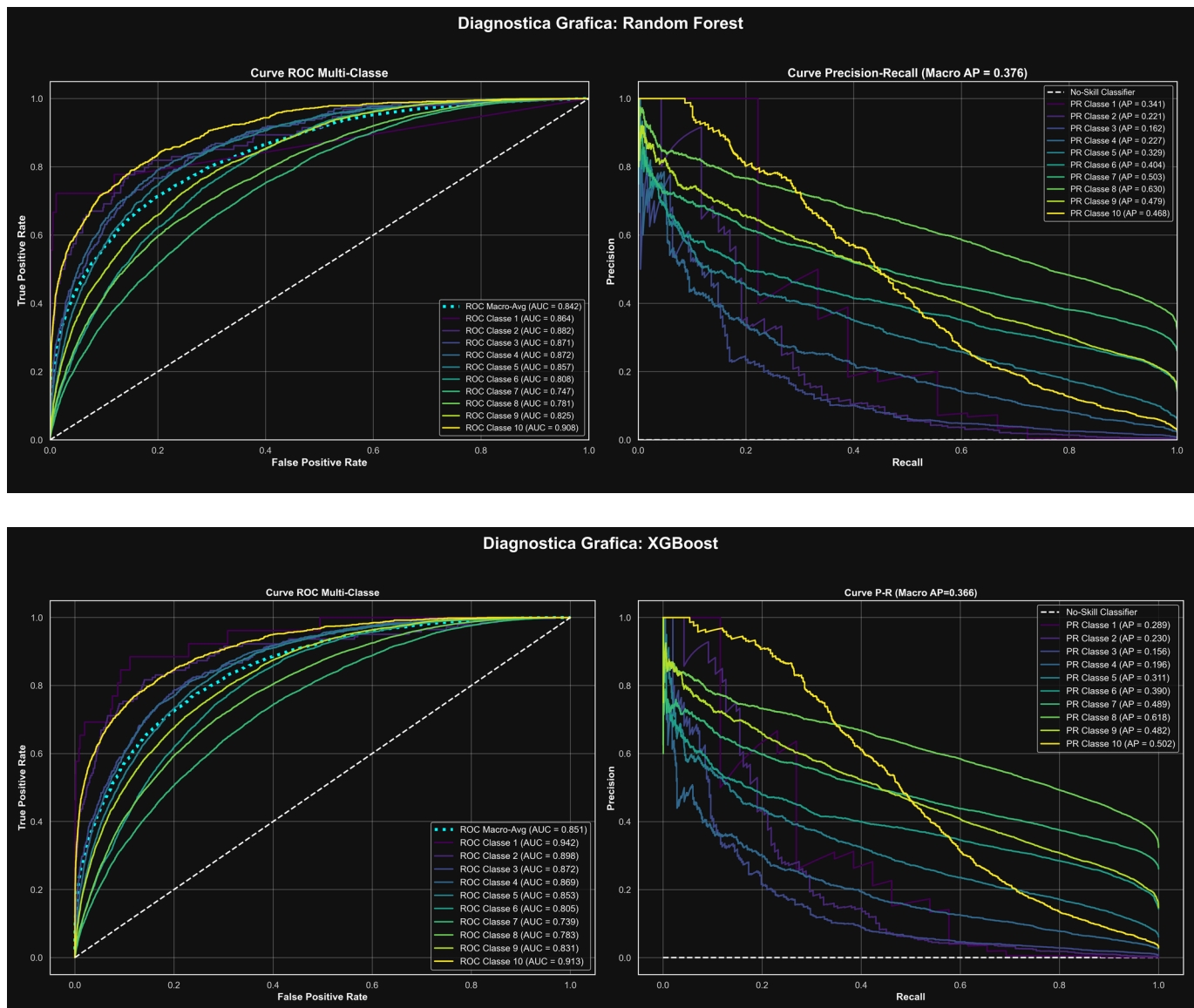


Figure 11: Diagnostic curves (ROC and Precision-Recall) for the tuned and calibrated models. **Top:** Random Forest. **Bottom:** XGBoost.

4.4.1 Hyperparameter Tuning.

A randomized search strategy was employed for both ensemble models to efficiently navigate their extensive hyperparameter spaces. RandomForestClassifier Cross-Validated F1-Macro = 0.3445, XGBoost CV F1-Macro = 0.3607

Random Forest — Optimal configuration:

- `n_estimators` = 352
- `max_depth` = 20
- `max_features` = 'sqrt'
- `min_samples_split` = 7
- `min_samples_leaf` = 1

XGBoost — Optimal configuration:

- `n_estimators` = 354
- `max_depth` = 9
- `learning_rate` = 0.154
- `gamma` = 0.424
- `subsample` = 0.702
- `colsample_bytree` = 0.988

Performance Evaluation: The ensemble models provided a dramatic performance improvement. The **Random Forest** achieved an **F1-Macro** of **0.37**, more than double that of the linear models. Crucially, it successfully predicts samples for all 10 classes. The **XGBoost** model achieved a comparable **F1-Macro** of **0.36**, and while slightly underperforming the Random Forest on the test set, it exhibited a more balanced profile, improving F1-scores on several minority classes. These results confirm that capturing non-linear feature interactions is essential for this task.

4.4.2 Feature Importance Analysis:

In the case of **Random Forest**, the most influential features were continuous variables, particularly `runtimeMinutes_proc`, `startYear_proc`, and `numVotes_proc`, which together highlight the importance of temporal and engagement-related metrics. These results suggest that the model strongly relies on production metadata and user interaction signals.

Conversely, **XGBoost** prioritized categorical features, especially those related to content type such as `type_tvEpisode`, `type_movie`, and `type_tvSeries`. These features dominated the gain-based importance metric, indicating that the model places greater emphasis on the type of production in determining the output.

This contrast reveals that while Random Forest exploits the numerical depth of metadata, XGBoost captures more hierarchical interactions based on content categories. The divergence in selected features reflects the different learning mechanisms and splitting strategies of the two ensemble methods.

4.5 Deep Neural Network (Weighted Softmax Classifier)

4.5.1 Architecture and Training Strategy

The model's architecture was designed for high capacity and robust regularization. A Deep Neural Network using a standard `softmax` output, but crucially trained with a weighted loss function. It consists of a stack of three hidden dense layers (256, 128, and 64 neurons respectively), each employing a modern block structure: **HeNormal** Initialization, **L2 Kernel Regularization**, **BatchNormalization**, **ReLU** Activation, and **Dropout** with a decreasing rate.

The input layer is preceded by a **GaussianNoise** layer for data augmentation. The most critical component of the training strategy, however, was the use of the `class_weight` parameter during the `.fit()` call. This technique re-scales the loss function, instructing it to penalize errors on minority classes far more heavily than errors on majority classes, thereby forcing the model to pay attention to them.

Training was governed by **EarlyStopping** and **ReduceLROnPlateau** to ensure optimal convergence.

4.5.2 Performance Evaluation

This weighted approach yielded significant behavioral changes. The model achieved an F1-Macro of **0.18**. While this score is low, the metrics tell a story of successful intervention: the model now achieves high **recall** on the extreme minority classes ('0.46' for '(0,1]' and '0.59' for '(9,10]'), completely solving the "prediction collapse" issue of previous unweighted neural network attempts. However, this comes at the cost of extremely low precision, indicating that while the model is no longer afraid to predict rare classes, it still struggles to do so accurately. This confirms that the predictive signal for these classes in the data is fundamentally weak.

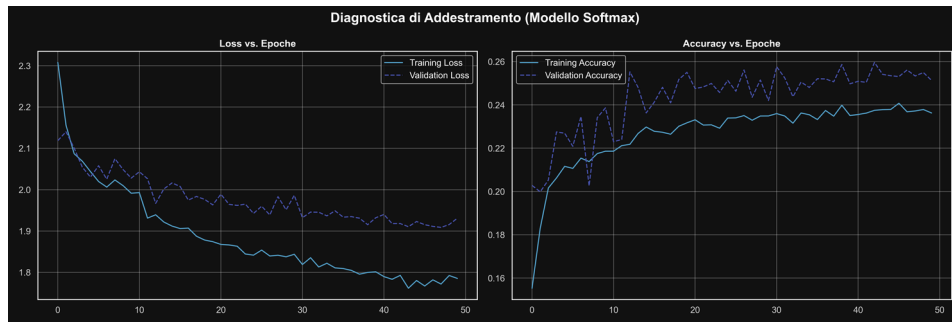


Figure 12: Training history for the Weighted Softmax DNN, showing the evolution of loss and accuracy on the training and validation sets over epochs. The stable convergence demonstrates the effectiveness of the regularization and callback strategies.

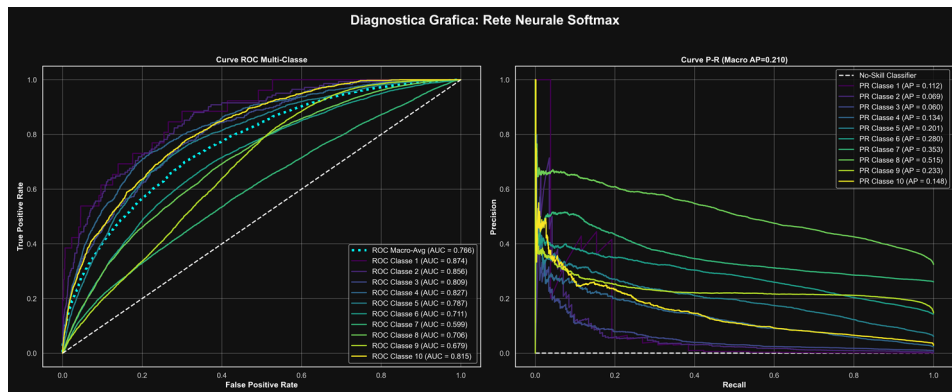


Figure 13: Diagnostic Curves (ROC and PR) for the Weighted Softmax DNN Model.

Table 8: Classification Report: Deep Neural Network (Weighted Softmax)

Target	Precision	Recall	F1-Score	Support
(0, 1]	0.01	0.46	0.01	26
(1, 2]	0.02	0.42	0.05	142
(2, 3]	0.04	0.25	0.07	353
(3, 4]	0.09	0.27	0.13	1029
(4, 5]	0.22	0.29	0.25	2718
(5, 6]	0.31	0.19	0.24	6364
(6, 7]	0.43	0.15	0.22	11618
(7, 8]	0.53	0.33	0.41	14440
(8, 9]	0.24	0.24	0.24	6559
(9, 10]	0.09	0.59	0.16	1281

4.6 Comparative Analysis and Final Conclusions

The decision to tackle the 10-class problem directly, while ambitious, led to a series of crucial insights into both the dataset’s nature and the models’ behaviors. The final performance metrics are summarized in Table 18.

Table 9: Final Comparative Performance of All Models on the 10-Class Task (Sorted by F1-Macro)

Model	F1-Score (Macro Avg)
Random Forest	0.37
XGBoost	0.36
Deep Neural Network (Weighted Softmax)	0.18
Logistic Regression	0.15
LinearSVC (Calibrated)	0.14

4.6.1 Conclusions

- **Linear Models Fall Short:** The 10-class task overwhelmed linear models. Logistic Regression and LinearSVC couldn’t manage severe class imbalance or nonlinear boundaries. LinearSVC completely failed on the rarest classes, proving simple models are inadequate for this granularity.
- **Ensemble Methods Excel:** Non-linear tree-based models delivered a major performance boost. Both Random Forest and XGBoost predicted all 10 classes well, with **Random Forest outperforming**. This shows decision trees’ ability to capture complex interactions and nonlinearities is key for this problem.
- **Neural Networks Highlight Data Limits:** The weighted deep neural network overcame prediction collapse on rare classes but sacrificed precision. This points to a fundamental issue: the predictive signal for a 10-level rating scale is weak and noisy. Adjacent classes are too subjective for reliable modeling by any method.

In sum, keeping all ten original classes, despite added difficulty, revealed important challenges otherwise hidden by aggregation.

Random Forest was the top performer with a macro F1-score of 0.37, handling imbalance well via non-linear feature interactions.

Linear models failed to capture rare classes and struggled with precision-recall balance. The deep network, despite sophistication and weighting, only reached 0.18 F1-score, indicating that *the main bottleneck lies in the inherent noise of human ratings*, not model complexity.

5 Explainability Random Forest Classifier

The Random Forest classifier (F1-Macro: 0.37) was the best performer on the 10-class movie rating task and deserve thorough explainability. We used **TreeExplainer** to compute exact SHAP values on 1,000 representative test instances.

Explainability is explored via three views: (1) **Global Analysis** with summary plots, (2) **Local Analysis** through waterfall plots, and (3) **Comparative Analysis** with decision plots.

5.1 Global Feature Importance Analysis

Figure 14 shows the global SHAP summary plot. The most influential feature is `type_tvEpisode`, consistent with domain knowledge that TV episodes have stable quality and rating patterns.

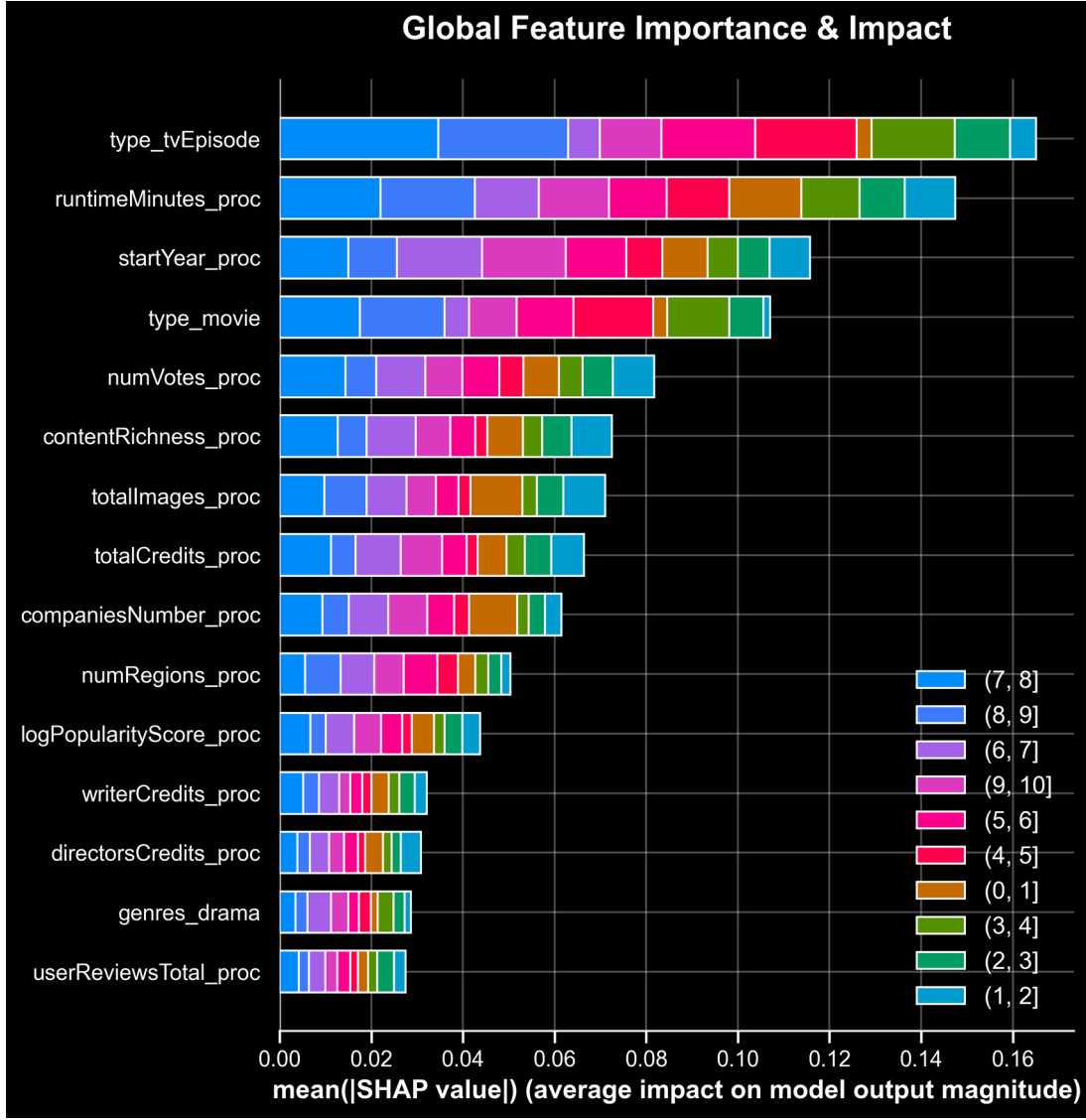


Figure 14: Global SHAP summary plot highlighting key features: `type_tvEpisode`, `runtimeMinutes_proc`, and `startYear_proc`.

Temporal features `runtimeMinutes_proc` and `startYear_proc` rank next. Runtime shows a nonlinear relation: both very short and very long runtimes correlate with lower ratings, moderate runtimes with higher ratings, indicating an optimal content length. More recent productions generally receive higher SHAP values, reflecting improvements in quality and storytelling.

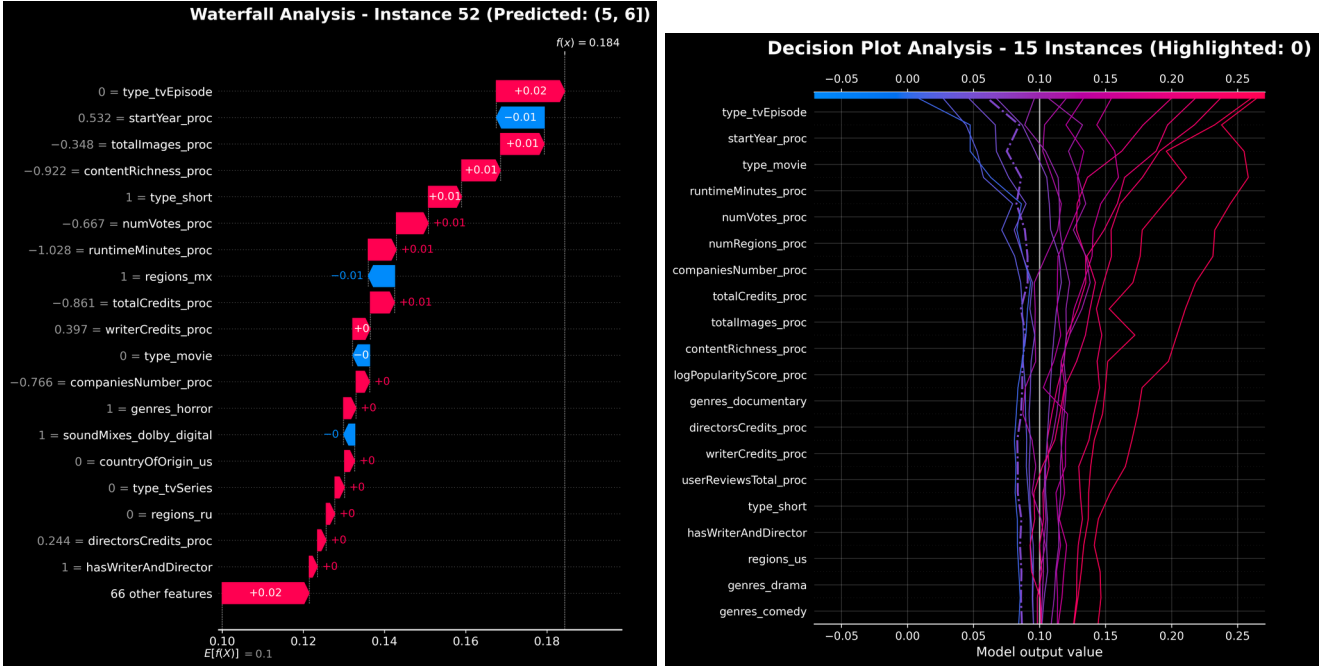
Content type features such as `type_movie` effectively distinguish rating patterns between movies and TV. Engagement metrics (`numVotes_proc`) and content richness (`contentRichness_proc`, `totalImages_proc`) are strong predictors, implying that audience interest and promotional effort proxy content quality.

5.2 Local Explainability: Case Study Analysis

Figure 15a exemplifies local explainability through a detailed waterfall plot for Instance 52 which has a true label of (6, 7] and received a predicted rating of (5, 6] with a model output value of 0.184, representing the highest decision score among all rating classes. This mid-range rating case demonstrates the nuanced decision-making process of the Random Forest model.

The waterfall analysis reveals a **balanced decision process** where positive and negative contributions compete. The strongest positive contributor is `type_tvEpisode` (+0.02), reinforcing the global finding that television episodes tend toward higher ratings. Conversely, `startYear_proc` provides a negative contribution (-0.01), indicating this particular instance represents older content that the model associates with lower ratings.

The analysis demonstrates several key insights: (1) **Content Type Dominance**: The binary nature of content type creates strong positive signals for television episodes, (2) **Temporal Effects**: Older content systematically receives lower predictions, reflecting industry evolution, (3) **Production Investment Indicators**: Features like `totalImages_proc` (+0.01) suggest that promotional investment correlates with quality expectations, and (4) **Geographic Factors**: The negative contribution of `regions_mx` (-0.01) indicates regional production differences in perceived quality.



(a) Waterfall plot for Instance 52 showing the step-by-step contribution of features to the final prediction. Starting from the base value $E[f(X)] = 0.1$, each feature contribution (positive in red, negative in blue) cumulatively leads to the final prediction $f(x) = 0.184$, representing the model's confidence score for the predicted rating class (5, 6]. (b) Decision plot analysis showing the decision pathways for 15 instances. Each line represents how feature contributions cumulatively influence the prediction from the base value toward the final output. The color gradient (blue to red) indicates different instances, with Instance 0 highlighted, revealing diverse decision patterns across the feature space.

Figure 15: SHAP analysis visualization: (a) Waterfall plot demonstrating individual feature contributions for a single instance, and (b) Decision plot showing prediction pathways across multiple instances

The mathematical decomposition shows: 0.1 (base) $+ 0.02 - 0.01 + 0.01 + \dots = 0.184$ (final), illustrating how individual feature contributions aggregate to produce the final rating prediction through the ensemble's decision-making process.

5.3 Decision Pathway Analysis

Figure 15b shows decision plots for 15 instances, illustrating diverse decision trajectories from the base value to final predictions. Instance 0 is highlighted to exemplify a specific path.

Key insights:

1. **Early Divergence:** Content type features (`type_tvEpisode`, `type_movie`) cause immediate separation, with some instances gaining positive momentum and others neutral.
2. **Runtime Correction:** `runtimeMinutes_proc` adjusts these initial signals, amplifying or dampening them.
3. **Convergent Features:** Lower-importance features provide fine adjustments refining the prediction.

This showcases Random Forest's strength in capturing complex, non-linear boundaries. Instances with similar colors follow comparable paths, reflecting similar content profiles, while divergent paths indicate distinct content types.

5.4 Implications and Model Insights

SHAP analysis confirms the model learns **meaningful content quality indicators**, not spurious correlations. Content type and temporal features dominate early decisions, with production investment and engagement metrics fine-tuning outcomes.

Explainability proves Random Forest's superiority over linear models by capturing **feature interactions** and **non-linear relationships**. The balanced local explanations show robust decisions that consider multiple features instead of relying on a single dominant one.

Practically, insights include: TV content receives rating premiums; optimal runtimes maximize ratings; newer productions benefit from advancements; and promotional investment (image counts) correlates with quality. These explainable decisions provide both model validation and strategic guidance for content producers.

6 Advanced Regression

This section details the methodology for constructing and evaluating advanced, non-linear regression models to predict the `averageRating` of audiovisual titles. The study compares two ensemble learning approaches: Random Forest and AdaBoost. The optimal configurations were found through intensive **hyperparameter optimization**. The framework ensured statistical rigor and transparency, covering data preparation, model selection, and hyperparameter tuning.

6.0.1 Target Variable Selection

The target variable is the continuous `averageRating`, an unscaled user rating from 1 to 10, chosen to preserve interpretability of error metrics like RMSE (e.g., 0.8825 directly relates to rating scale accuracy).

6.0.2 Feature Space Construction

The feature matrix **X** contains 85 columns from processed dataset variables, including:

- Standardized numerical features (e.g., `numVotes_proc`, `runtimeMinutes_proc`).
- One-hot encoded categorical features for nominal variables (e.g., `genres_drama`, `type_tvEpisode`).
- Engineered features capturing complex interactions (e.g., `contentRichness_proc`, `winRate_proc`).

Identifier variables, the target itself, and raw precursor columns were excluded to avoid leakage and multicollinearity.

6.0.3 Model Selection

Two distinct, powerful, non-linear regression algorithms were selected for comparative analysis:

- **Random Forest Regressor:** A state-of-the-art ensemble technique predicated on the principle of *bagging* (Bootstrap Aggregating). It cultivates a large "forest" of decorrelated decision trees, each trained on a random bootstrap sample of the data. The final prediction is derived by averaging the outputs of all constituent trees, a process that markedly reduces variance and enhances generalization.
- **AdaBoost Regressor:** A pioneering *boosting* algorithm that constructs a strong predictive model by iteratively fitting a sequence of weak learners. Each subsequent learner is trained to rectify the errors of its predecessors by assigning greater weight to misclassified instances. The final model constitutes a weighted sum of these weak learners.

6.0.4 Hyperparameter Optimization

To ascertain the optimal configuration for each model, we implemented a **Grid Search with 5-fold Cross-Validation** (`GridSearchCV`). This gold-standard technique exhaustively navigates a predefined hyperparameter space, robustly evaluating each parameter combination to prevent optimistic performance estimates and select a model with superior generalization capability. The objective function for the grid search was the **Negative Root Mean Squared Error** (`neg_root_mean_squared_error`).

6.1 Model Performance and Quantitative Evaluation

6.1.1 Evaluation Metrics

- **Mean Absolute Error (MAE):** Average absolute difference between true and predicted values,

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

- **Root Mean Squared Error (RMSE):** Square root of mean squared differences, penalizing large errors,

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

- **Coefficient of Determination (R^2):** Fraction of variance explained by the model,

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

where n is sample size, y_i true targets, $\hat{y}_i$ predictions, and $\bar{y}$ mean of true values.

6.1.2 Results

The exhaustive grid search identified optimal hyperparameters shown in Table 10.

Table 10: Optimal Hyperparameters for Random Forest and AdaBoost Regressors

Random Forest Regressor		AdaBoost Regressor	
n_estimators	500	n_estimators	300
max_depth	30	learning_rate	0.1
max_features	0.7	loss	linear
min_samples_split	5	estimator__max_depth	30
min_samples_leaf	3	estimator__min_samples_split	2
		estimator__min_samples_leaf	3

The out-of-sample performance of both fully-tuned models on the hold-out test set is summarized in Table 11.

Table 11: Final Performance Metrics Comparison on the Hold-out Test Set

Model	MAE	RMSE	R-squared (R^2)
AdaBoost (Optimized)	0.6115	0.8825	0.5682
Random Forest (Optimized)	0.6230	0.8921	0.5588

6.1.3 Comparative Analysis

The empirical results from the intensive tuning phase represent a fascinating convergence of two distinct ensemble philosophies. As detailed in Table 11, both the **AdaBoost** and **Random Forest** models, after exhaustive optimization, achieved remarkably similar and highly competitive performance levels. While AdaBoost technically registered the best scores across all metrics, the differences are marginal and likely fall within the statistical noise of the test set. The deltas in MAE, RMSE, and R^2 are 1.88%, 1.09%, and 1.68% respectively. This near-parity is a significant finding in itself, suggesting that both algorithms, when pushed to their optimal configurations, have reached a similar performance ceiling on this particular dataset.

A crucial aspect of the Random Forest model’s evaluation is its **Out-of-Bag (OOB) Score**. This is an internal, unbiased estimate of generalization error. Our optimized Random Forest model yielded an OOB R^2 of **0.5632**. This value is exceptionally close to its test set R^2 of 0.5588 (a negligible difference of 0.0044). This confirms the model’s outstanding robustness and its excellent ability to generalize, solidifying its status as a reliable and practically equivalent alternative to the AdaBoost model.

6.1.4 Diagnostic and Residual Analysis

Figure 16 shows diagnostic plots comparing both models.

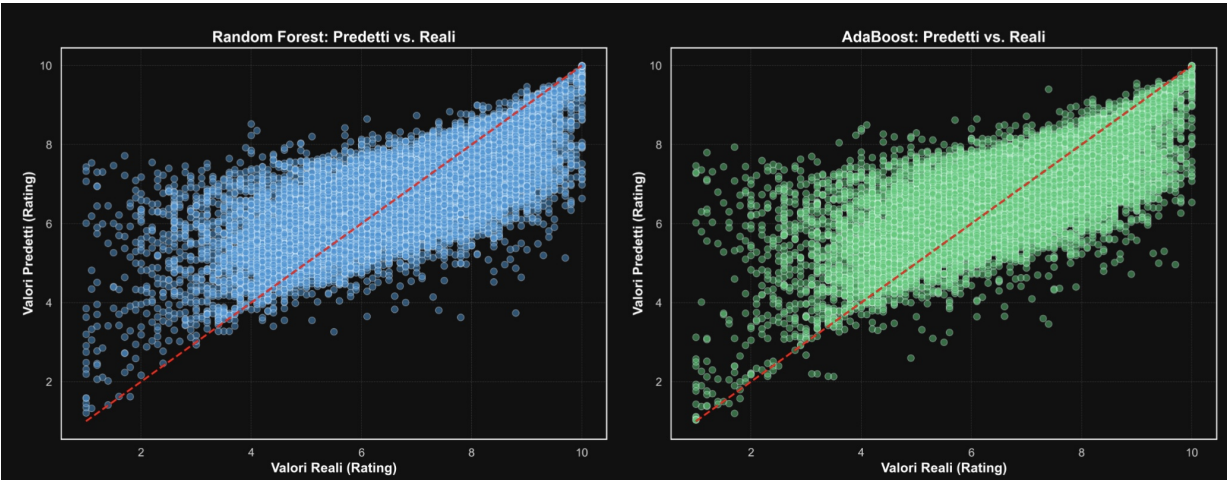


Figure 16: Diagnostic plots for optimized Random Forest and AdaBoost models: shows predicted vs. actual values

The diagnostic scatter plots reveal that AdaBoost predictions are more tightly aligned along the identity line, especially in the mid-to-high rating range. Random Forest, while competitive, shows a bit more dispersion and underestimation in the extremes. Both models struggle mildly in the [1–3] rating zone, but AdaBoost exhibits more consistent tracking overall.

6.2 Comparative Feature Importance Analysis

To interpret predictive logic, we compared feature importances of both models, exposing key drivers and differing algorithmic strategies. Table 12 lists the top seven influential features per model.

Table 12: Comparative Feature Importance: Top 7 Predictors

Random Forest Importance		AdaBoost Importance	
Feature	Importance	Feature	Importance
runtimeMinutes_proc	0.1700	genres_adventure	0.1385
startYear_proc	0.1206	startYear_proc	0.1329
type_tvEpisode	0.0991	runtimeMinutes_proc	0.1303
numVotes_proc	0.0759	numVotes_proc	0.0803
contentRichness_proc	0.0474	type_tvEpisode	0.0458
totalCredits_proc	0.0467	contentRichness_proc	0.0452
type_movie	0.0465	totalCredits_proc	0.0449

This comparison yields several critical insights:

- **Consensus on Core Drivers:** Both models concur on a core set of highly influential features: `runtimeMinutes_proc`, `startYear_proc`, `numVotes_proc`, and the format (`type_tvEpisode`). This cross-model agreement provides strong evidence that these are genuine, robust signals within the data.
- **Divergent Top Predictors:** A fascinating divergence is observed in the single most important feature. The Random Forest model identifies a continuous production metric, `runtimeMinutes_proc`, as the primary predictor. In contrast, AdaBoost elevates a categorical genre flag, `genres_adventure`, to the top position. This suggests AdaBoost’s boosting algorithm found that isolating the "adventure" genre provided the single largest initial reduction in error.
- **Validation of Feature Engineering:** The engineered feature `contentRichness_proc` appears prominently in the top features for *both* models. This powerfully validates our feature engineering hypothesis, confirming that a composite metric of metadata availability is a significant and reliable predictor.

6.3 Conclusion

This investigation developed and optimized two advanced regression models, reaching a clear conclusion. The key finding is the **performance convergence** of two distinct ensemble methods, bagging and boosting, after thorough hyperparameter tuning.

Both **AdaBoost** and **Random Forest** achieved decent metrics, with AdaBoost slightly ahead (RMSE 0.8825, R^2 0.5682), though differences likely fall within statistical noise. Despite different approaches, both identify and exploit similar predictive patterns, reaching comparable performance ceilings.

The Random Forest stood out for robustness, with its Out-of-Bag score closely matching test performance, confirming it as a co-equal solution.

Feature importance analysis shows agreement on key predictors but reveals different internal strategies, offering insight into data structure.

Ultimately, this confirms the practical performance limit on this dataset is around 56-57% explained variance, attainable by both ensemble methods with proper tuning.

7 Time Series Data Understanding and Preparation

This phase establishes a thorough understanding of the time-series dataset comprising daily revenues for **1,134** films. Each film’s revenue spans a fixed window of **100 days**, forming a data matrix of shape (1134, 100). Metadata such as `rating`, `genre`, and `rating_category` augment the data. Data quality checks confirm **no zero-revenue days**, ensuring dense, complete time series and removing the need for imputation, thus preventing artifacts and enabling reliable analysis.

7.1 Statistical Analysis of Revenue Distribution

The total 100-day revenues show extreme variance and strong positive skew. Table 13, first column "Total Revenue" summarizes key statistics.

A key insight is the large gap between mean (\$146.8M) and median (\$75.8M) revenues, indicating a strong right skew caused by a few blockbuster films. The standard deviation (\$209.8M) exceeds the mean, confirming vast dispersion. The revenue range from \$139k to over \$1.7B highlights the *winner-take-all* nature of the industry. This necessitates normalization before shape-based analyses.

7.2 Revenue Feature Engineering and Correlation Analysis

To analyze revenue dynamics, 14 engineered features covering revenue patterns, temporal trends, and volatility were created. Table 13 summarizes the most important ones.

Table 13: Descriptive Statistics for Key Engineered Revenue Features.

Statistic	Total Revenue	Log Total Revenue	Opening Day	Opening Weekend	First Week
Mean	\$146,807,424	17.94	\$9,075,217	\$25,551,050	\$46,343,460
Median	\$75,853,664	18.14	\$4,025,919	\$12,141,780	\$22,637,580
Std. Dev.	\$209,948,000	1.49	\$15,595,430	\$40,938,490	\$72,072,410
Min	\$139,777	11.85	\$1,266	\$7,198	\$18,660
Max	\$1,763,035,000	21.29	\$157,461,600	\$401,279,000	\$695,841,300

These features highlight the *front-loaded* revenue pattern. On average, the **opening weekend accounts for 15%** of a film’s total 100-day revenue, and the **first week captures 28%**. The **peak day averages at 8.95** with high variance (17.07 days), reflecting diverse trajectories. Weekend revenues consistently represent 50% of earnings, indicating stable audience behavior.

7.2.1 Correlation Analysis

Revenue-based features show strong multicollinearity ($r=0.93$ – 0.99 among `opening_weekend`, `first_week`, `total_revenue`), while **ratio-based** features are more independent (`opening_ratio` vs `first_week_ratio`: $r=0.96$). Notably, **peak_day negatively correlates with early revenue metrics** ($r=-0.72$ to -0.77), confirming distinct archetypes: front-loaded blockbusters vs. sustained performers. The **late_ratio** is largely independent ($r=-0.12$ to 0.16), valuable for identifying unconventional revenue patterns.

Importantly, **172 films (15.2%)** show *late surge* behavior with high `late_ratio` (mean= 0.17 , max= 0.96), deviating from simple decay and suitable for clustering or anomaly detection.

7.3 Global Pattern Discovery

Figure 17 supports these findings visually. The **log-revenue heatmap** of the top 200 films reveals unique signatures yet confirms a common pattern: a strong initial peak followed by gradual decay. Log scale avoids domination by blockbuster magnitudes.

Aggregated daily revenue shows mean and median revenue over 1,134 films with percentile bands, confirming a sharp initial spike and rapid decay, plus a persistent mean-median gap illustrating skew from blockbuster influence.

The **log-scaled histogram of total revenue** confirms a heavy-tailed market, with most films clustered low and few high-grossing outliers, reinforcing the *winner-take-all* nature.

Critically, averaging revenue curves by deciles shows the *shape* of revenue trajectories is stable across success levels, differing mainly in magnitude. This justifies shape-based distance measures on normalized data.

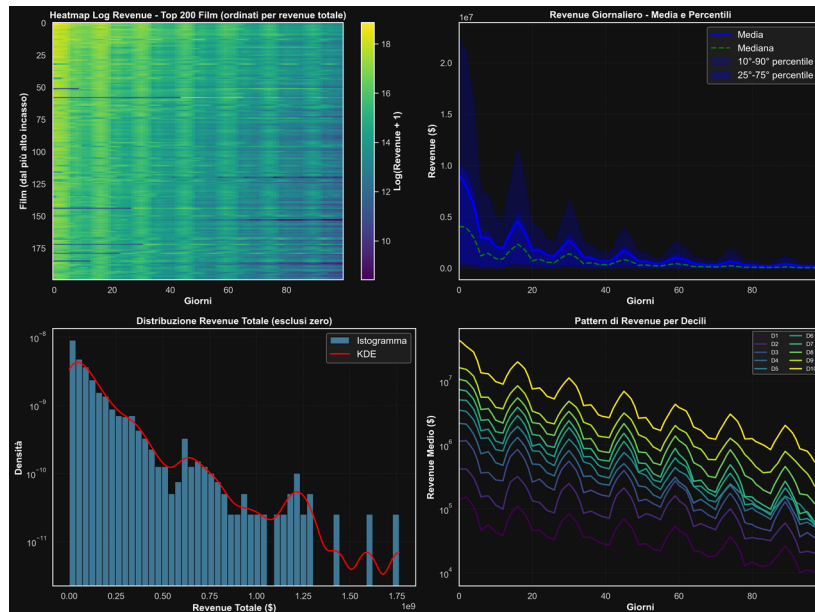


Figure 17: A composite visualization detailing the global patterns within the dataset.

7.3.1 Decay Patterns

A visual inspection of Z-normalized series reveals two main behavioral archetypes in film revenue dynamics (Figure 18):

- **Fast Decay:** Characterized by a sharp initial peak followed by rapid decline, typical of *front-loaded* films reliant on pre-release marketing rather than sustained engagement.
- **Slow Decay:** Exhibits gradual, sustained revenue decline, indicating longer market longevity, often linked to positive word-of-mouth or "sleeper hits."

These distinct patterns justify the use of clustering algorithms, confirming the dataset contains **meaningful subgroups**.

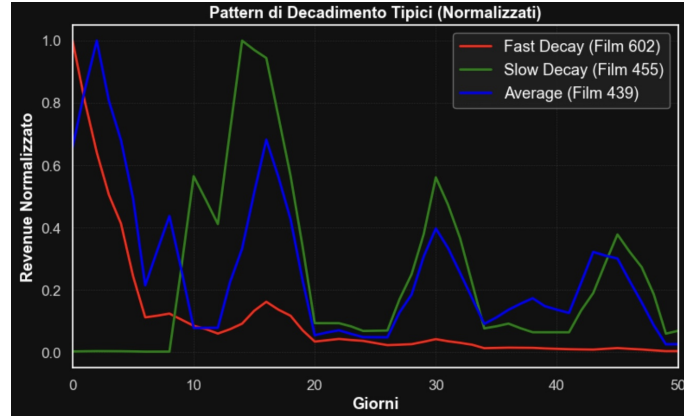


Figure 18: Mean Z-normalized trajectories of the two primary behavioral patterns observed during exploratory analysis: Fast Decay and Slow Decay. The overall dataset average is included as a baseline.

7.4 Data Pre-processing and Representation

Raw revenue series risk producing trivial groupings dominated by magnitude. To focus on behavior over scale, three key transformations were applied:

1. **ts_raw:** Original data, retaining absolute **magnitude features** (e.g., total revenue).
2. **ts_zscore_film:** Per-series Z-normalization standardizes each series to mean 0 and std 1, removing magnitude effects. This is essential for **shape-based clustering** and **motif/discord detection**.
3. **ts_percentage:** Converts each data point to a percentage of total revenue, useful for **classification** by capturing relative commercial strategy (e.g., first-week revenue share).

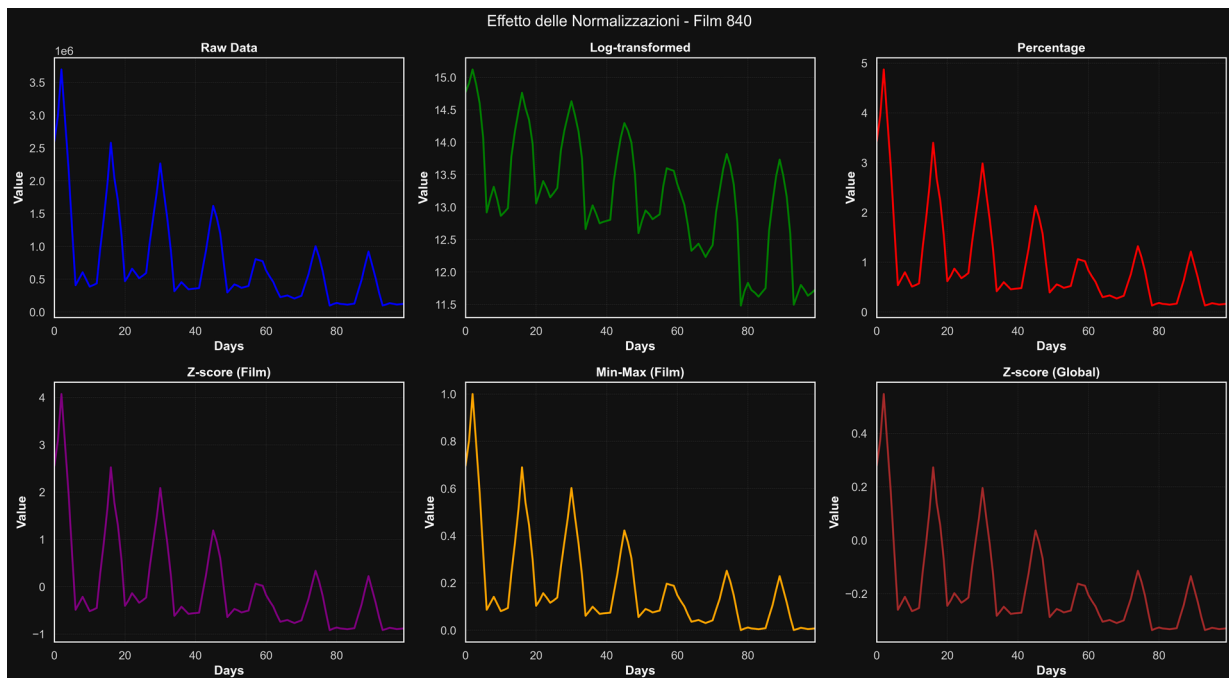


Figure 19: A direct comparison of all the six transformations performed.

8 Motifs, Discords, and Shapelets: A Unified Temporal Analysis

8.1 Introduction and Methodology

The primary objective was to identify recurring temporal patterns (motifs), significant anomalies (discords), and to understand their relationship with class-specific discriminative features (shapelets) derived from movie genres.

To ensure that the analysis focused on the underlying shape of the popularity curves rather than their absolute scale, each time series was individually Z-normalized. Motif and discord discovery was performed using the **Matrix Profile** algorithm, calculated via the **stumpy** library on a single, concatenated series of all samples. This approach enables a global search for patterns and anomalies across the entire dataset, identifying the most structurally relevant occurrences regardless of their source series.

8.2 Motif and Discord Discovery

The initial phase of the analysis focused on identifying the most frequent (motif) and the most structurally distinct (discord) subsequences within the dataset.

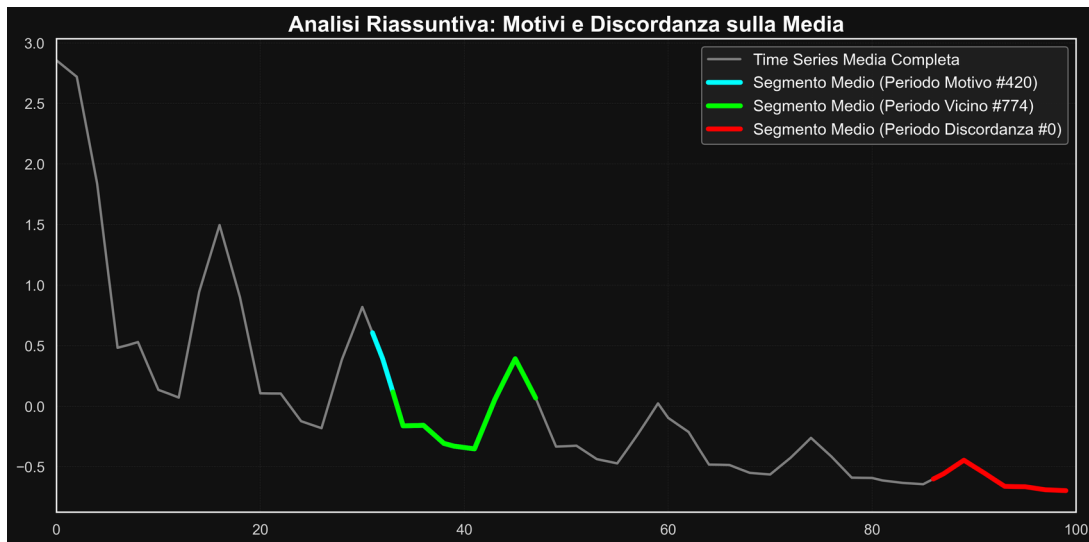


Figure 20: Summary of key temporal events overlaid on the dataset’s average popularity curve. The cyan and lime segments indicate the temporal locations of the motif and its neighbor, while the red segment highlights the discord. This visualization emphasizes the temporal coherence of motifs and the outlier status of the discord.

8.2.1 Global Motif Analysis

The Matrix Profile algorithm successfully identified a highly recurrent pattern. The two most similar instances of this motif were found in the time series with DataFrame indices 420 and 774. A lookup of the associated metadata revealed a compelling narrative:

- *Series 420*: Genres: ['Action', 'Biography', 'Drama'], Rating: 7.3
- *Series 774*: Genres: ['Action', 'Crime', 'Drama'], Rating: 7.3

The common thread is the shared genres *Action* and *Drama*, and an identical high rating. The motif, as visualized in Figure 20, exhibits a sharp initial peak in popularity followed by a gradual decline. The most frequent pattern across the dataset appears to be the signature popularity curve of successful, high-rated Action/Drama films, suggesting a highly homogeneous lifecycle for this sub-genre.

8.2.2 Global Discord Analysis

The algorithm identified the global discord in Series #0: the first time series in the dataset. This film is categorized under *Adventure* and *Sci-Fi*, with an IMDb rating of 8.3. The calculated Matrix Profile distance was undefined (effectively ∞), indicating the absence of any sufficiently similar subsequence across the dataset.

As shown in Figure 20 in red, the anomalous pattern is characterized by a sharp decline followed by a partial recovery, occurring in the final segment of the 100-day cycle. This high-volatility behavior deviates sharply from the smoother decay trends exhibited by the majority of other series.

The discord represents a *black swan* event, a atypical popularity trajectory for a very successful Sci-Fi film. Its structure is not replicated anywhere else in the dataset, underscoring its uniqueness.

8.3 Relationship with Shapelets

To formally investigate the connection between the discovered motif and genre, a shapelet analysis was conducted. The goal was to determine whether the motif’s shape is also a discriminative signature for its associated genres.

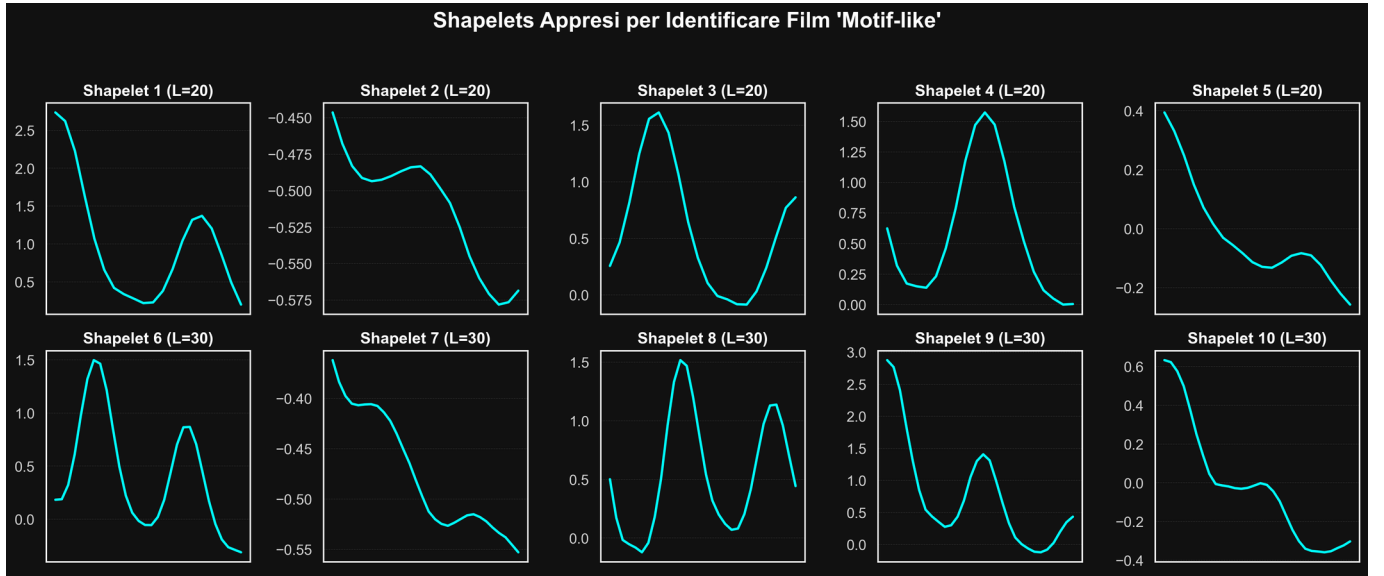


Figure 21: The top 10 shapelets learned by the model to distinguish 'Action/Drama' films from all others. Note the prevalence of initial-peak patterns.

8.3.1 Hypothesis-Driven Shapelet Discovery

Based on the motif analysis, we hypothesized that a discriminative temporal signature characterizes films that belong to both the *Action* and *Drama* genres. To test this hypothesis, a binary target variable was constructed to identify such films, and a `ShapeletModel` from the `tslearn` library was trained on the entire time series dataset. The model was configured to account for potential class distribution differences, ensuring that genre-specific patterns would not be overshadowed during the learning process.

8.3.2 Learned Shapelets and Interpretation

The `ShapeletModel` identified a coherent set of discriminative subsequences, shown in Figure 21. These shapelets represent meaningful recurring patterns tied to the *Action/Drama* genre. Visually, they cluster into three macro-categories:

1. **Initial Peak Patterns (e.g., Shapelets 1, 6, 9):** Sharp, high peaks in the first few steps followed by rapid decay. These resemble “blockbuster” trajectories driven by pre-release hype and marketing, with fast market saturation.
2. **Symmetric Bump Patterns (e.g., Shapelets 3, 4, 8):** Balanced bell-shaped curves with moderate rise and fall around the mid-time window, likely reflecting films with slower, steadier popularity growth via positive word-of-mouth or festivals.
3. **Monotonic Declines with Minor Fluctuations (e.g., Shapelets 2, 5, 7, 10):** Gradual decreases from moderate values with occasional small inflections, typical of mid-tier films fading slowly without major spikes.

These categories align with the genre lifecycle hypotheses: *Initial Peak* shapelets highlight early concentrated popularity as a key temporal signature for *Action/Drama*. The symmetric and declining shapes add interpretability, showing diverse success paths within this genre.

8.3.3 Final Synthesis: Connecting Motifs and Shapelets

Comparing the global motif and learned shapelets (Figure 21) reveals strong structural similarity. Shapelets 1 and 6 closely match the motif’s early sharp peak and gradual decline, supporting the hypothesis that this temporal structure is both statistically frequent and semantically discriminative, a “signature” of successful *Action/Drama* films.

Figure 20 situates motif, neighbor, and discord on the average popularity trajectory, showing motifs appear in the mid-early cycle (growth and decline phase), while the discord emerges late when most series have stabilized or decayed. The discord’s anomalous rise stands out in both shape and timing.

This analysis links three temporal concepts: **frequency** (motifs), **uniqueness** (discords), and **discriminative power** (shapelets). The shapelet model’s alignment with motifs confirms motif-driven shapelet discovery as an interpretable, explainable method for time series classification.

9 Clustering: K-Medoids and Hierarchical

The clustering phase aims to partition the dataset of $N = 1134$ time series into subsets with similar temporal dynamics. To focus on shape rather than scale, each series was Z-score normalized. Similarity was measured using a **Dynamic Time Warping** (DTW) distance matrix, suitable for temporal shifts and distortions. Clustering algorithms chosen operate directly on this distance matrix, excluding centroid-based methods like K-Means.

9.1 K-Medoids (Partitioning Around Medoids - PAM)

The K-Medoids algorithm was identified as the primary candidate for this analysis. In contrast to K-Means, which minimizes the sum of squared euclidean distances to a centroid, a mean-based, often synthetic data point that may not exist in the dataset, K-Medoids minimizes the sum of dissimilarities to a **medoid**. The medoid is an actual data point, specifically the most central observation within its cluster. This approach offers two decisive strategic advantages:

1. **Metric Agnosticism:** It is inherently compatible with any arbitrary distance metric, including our pre-computed DTW matrix, as it does not require algebraic operations on the data points themselves to define the cluster centers.
2. **Enhanced Interpretability:** The cluster centers (the medoids) are real time series from the dataset. This renders the interpretation of a cluster's prototypical behavior direct, concrete, and grounded in an actual example, obviating the need to analyze an artificial average curve.

9.2 Agglomerative Hierarchical Clustering

As a comparative method, an agglomerative hierarchical clustering algorithm was implemented. This bottom-up approach begins with N clusters (one for each data point) and iteratively merges the two closest partitions until the desired number of clusters is reached. An **average** linkage criterion was employed, which defines the inter-cluster distance as the average of all pairwise distances between points in the two clusters. However, the application of this method yielded a highly skewed and imbalanced partition. The result was a single macro-cluster containing over 80% of the observations, alongside numerous clusters composed of single elements (singletons). This phenomenon, not uncommon in high-dimensional spaces, indicates a failure of the algorithm to identify a meaningful grouping structure, rendering it unsuitable for the analytical goals of this study.

9.3 Optimal Number of Clusters (k)

Choosing k for K-Medoids relied on **Silhouette Score** analysis, which measures cluster cohesion vs separation (+1 is best). Scores were computed for $k \in [2, 20]$, peaking at $k = 2$. Although statistically optimal, two clusters were too coarse for meaningful segmentation. Thus, $k = 5$ was chosen as a pragmatic balance between quality and actionable granularity, a common approach in applied data science.

9.4 Analysis and Visualization of the Cluster Partition

The qualitative and quantitative analysis focused on the 5-cluster partition obtained from the K-Medoids algorithm, given its superior performance in terms of cluster balance and interpretive potential.

9.4.1 Dimensionality Reduction for Visualization

To visualize the cluster structure, the 100-dimensional time series space was projected onto a 2D Cartesian plane using two manifold learning techniques:

- **Multi-Dimensional Scaling (MDS):** This technique aims to preserve the global pairwise dissimilarities of the original DTW matrix. Its projection provides a map that best respects the large-scale distances between all data points.
- **t-Distributed Stochastic Neighbor Embedding (t-SNE):** A non-linear algorithm that excels at preserving local structure. It converts high-dimensional affinities into probabilities, making it exceptionally effective at revealing the intrinsic grouping structure of the data.

Visual inspection in Figure 22 revealed that the **t-SNE** projection produced a compelling map with visually dense and clearly separated clusters, whereas MDS showed considerable overlap. This result not only qualitatively validates the robustness of the K-Medoids partition but also suggests that the identified clusters represent natural, well-defined groupings within the DTW feature space.

Notably, hierarchical clustering exhibits severe limitations: even with $k = 5$, it produces a dominant macro-cluster and poorly defined groups, failing to capture the complex temporal morphology captured by K-Medoids. The contrast across projection methods underscores the need for both robust clustering algorithms and expressive visualization tools in time series segmentation.



Figure 22: Visualization of the clusters obtained via K-Medoids and Hierarchical Clustering using DTW distance, projected through dimensionality reduction techniques (MDS and t-SNE)

9.5 Characterization and Interpretation of the Discovered Clusters

The analysis of the 5 medoids (the prototypes) and the distribution of metadata (**genre**, **rating_category**) within each cluster allowed for the delineation of 5 distinct cinematic product life cycle archetypes.

9.5.1 Cluster 0 (312 films): The Front-Loaded Blockbuster

- **Temporal Dynamics:** A profile characterized by a very high initial velocity of revenue generation, with a peak reached in the first few days, followed by a rapid exponential decay. This is the signature of event-driven films supported by massive pre-launch marketing campaigns.
- **Composition:** Overwhelmingly dominated by high-impact genres such as **Action**, **Adventure**, and **Sci-Fi**. The rating category is almost exclusively **High** and **Medium High**, confirming the association with high-profile productions.

9.5.2 Cluster 1 (110 films): The High-Variance Performer

- **Temporal Dynamics:** A time series exhibiting markedly stochastic behavior, lacking a discernible trend and characterized by irregular peaks and troughs throughout the entire period. This suggests a volatile audience or highly polarized word-of-mouth dynamics.
- **Composition:** High incidence of genres such as **Horror**, **Thriller**, and **niche Sci-Fi**. These genres often cater to dedicated fanbases whose viewing behavior can be influenced by unconventional, external factors.

9.5.3 Cluster 2 (107 films): The Stable & Sustained Success

- **Temporal Dynamics:** A profile with an excellent initial box office performance followed by a very low decay rate, creating a "plateau" curve. It represents sustained success over time, benefiting from continuous positive

word-of-mouth.

- **Composition:** Primarily composed of genres like **Animation, Adventure, and Comedy**. This suggests a strong association with family films or broad-appeal comedies that enjoy a long theatrical run. The rating is almost exclusively **High**.

9.5.4 Cluster 3 (491 films): The Normative Long-Tail Performer

- **Temporal Dynamics:** As the most populous cluster, it represents the standard product lifecycle in this market: a modest initial peak followed by a gradual and steady decline. It embodies the classic "long-tail" concept.
- **Composition:** Extremely diversified, with a strong representation of genres like **Drama, Comedy, and Romance**. The rating distribution covers the entire spectrum, positioning this cluster as the market's baseline model.

9.5.5 Cluster 4 (114 films): The Slow-Burn Niche Success

- **Temporal Dynamics:** Characterized by an incubation phase with very low initial revenues that exhibit slow but progressive growth, sometimes culminating in a late-stage peak. This is the archetype of the *sleepers* hit.
- **Composition:** Dominated by industry-specific genres such as **Documentary, Music, and Biography**. The rating is predominantly **High** and **Medium High**, indicating critically acclaimed films whose commercial success is driven by prestige and qualified word-of-mouth rather than mass marketing.

10 Time Series Classification

This section details the methodology and results for the time series classification task. The objective is to classify movies into one of four rating categories (**High, Medium High, Medium, Medium Low**) based on the 100-day revenue trajectory.

The original dataset comprised five rating categories, including a sparsely populated **Low** category with only 10 instances. To mitigate statistical unreliability and improve model robustness, this class was removed from the analysis, resulting in a more balanced, yet still moderately imbalanced, four-class classification problem. The dataset was then partitioned into training (80%) and testing (20%) subsets. Stratified splitting ensured the preservation of the class distribution across these subsets, critical for maintaining representativeness and facilitating unbiased evaluation.

10.1 Evaluation Metrics

Given the moderate imbalance across categories, traditional accuracy was deemed inadequate due to its sensitivity to class distribution. Consequently, the models were rigorously evaluated using the **Macro F1-Score**, computed by independently calculating the harmonic mean of precision and recall for each class and averaging across all classes. Additionally, **Balanced Accuracy**, defined as the average recall across classes, was employed. These metrics collectively provided a robust and holistic assessment of model performance across all classes, reducing biases introduced by class imbalance.

10.2 Model 1: K-Nearest Neighbors (k-NN) with Euclidean or Manhattan Distances

The k-Nearest Neighbors classifier served as a foundational baseline model. To ensure methodological rigor, scaling was encapsulated within a `scikit-learn` pipeline employing `StandardScaler` prior to classification, thus preventing any potential data leakage. Hyperparameter optimization was conducted using a comprehensive `GridSearchCV` strategy with 5-fold stratified cross-validation. This identified an optimal configuration of 10 neighbors utilizing the Euclidean (Manhattan was discarded as the best metric) distance metric, which yielded a cross-validation Macro F1-Score of 0.3988 and a Macro F1-Score of 0.3702 on the unseen test data.

10.2.1 Performance Analysis

The best configuration selected via grid search was `n_neighbors=10` with **Euclidean** distance, yielding a cross-validated Macro F1-Score of 0.3988. On the test set, performance remained consistent with a Macro F1-Score of 0.3702 and Balanced Accuracy of 0.3786. Training and prediction times were fast (1.70s and 0.03s), confirming the method's efficiency.

However, the model struggled with class separability. While it performed reasonably well on **Medium** (recall 0.67) and **High** (0.55), it failed to distinguish **Medium High** (recall 0.11) and **Medium Low** (0.19). The confusion matrix reveals that most **Medium High** instances were misclassified as **Medium**, suggesting that adjacent classes with subtle temporal differences are poorly resolved in Euclidean space.

Table 1: Performance of k-NN (Pipeline) on the Test Set

Classification Report					Confusion Matrix				
Class	Prec.	Rec.	F1	Supp.	Predicted →				
					True ↓	ML	M	MH	H
Medium Low	0.29	0.19	0.23	26	Medium Low	5	18	0	3
Medium	0.43	0.67	0.52	78	Medium	8	52	4	14
Medium High	0.28	0.11	0.16	46	Medium High	2	28	5	11
High	0.59	0.55	0.57	75	High	2	23	9	41
<i>Accuracy</i>			<i>0.46</i>	<i>225</i>					
<i>Macro avg</i>	<i>0.40</i>	<i>0.38</i>	<i>0.37</i>	<i>225</i>					
<i>Weighted avg</i>	<i>0.44</i>	<i>0.46</i>	<i>0.43</i>	<i>225</i>					

10.3 Model 2: K-Nearest Neighbors (k-NN) with Dynamic Time Warping (DTW)

To investigate the potential of capturing temporal distortions within the data, a k-NN classifier leveraging Dynamic Time Warping (DTW) was explored. DTW provides an elastic, non-linear alignment of time series, theoretically advantageous for handling temporal shifts and distortions. GridSearch identified the optimal neighborhood size as 7.

10.3.1 Performance Analysis

Contrary to expectations, the DTW-based k-NN underperformed compared to Euclidean, with a test Macro F1-Score of 0.3466 and Balanced Accuracy of 0.3600. Computationally, it was costly: fitting took 182 seconds, prediction 60 seconds—over 2000 times slower than Euclidean—due to DTW’s $\mathcal{O}(n^2)$ complexity.

The confusion matrix reveals poor recognition of **Medium High** (recall 0.07), mostly misclassified as **Medium**, indicating overlap between these adjacent classes. Additionally, confusion between **Medium** and **High** was pronounced, reducing class separability in this range.

Although DTW aims to handle temporal distortions by aligning sequences flexibly, here it appears to have introduced noise. Revenue trajectories seem better distinguished by overall shape and amplitude rather than phase shifts, so DTW’s elasticity may have obscured meaningful class differences rather than clarified them.

Table 2: Performance of k-NN (DTW) on the Test Set

Classification Report					Confusion Matrix				
Class	Prec.	Rec.	F1	Supp.	Predicted →				
					True ↓	ML	M	MH	H
Medium Low	0.26	0.27	0.26	26	Medium Low	7	17	0	2
Medium	0.43	0.69	0.53	78	Medium	11	54	7	6
Medium High	0.12	0.07	0.08	46	Medium High	4	31	3	8
High	0.66	0.41	0.51	75	High	5	24	15	31
<i>Accuracy</i>			<i>0.42</i>	<i>225</i>					
<i>Macro avg</i>	<i>0.37</i>	<i>0.36</i>	<i>0.35</i>	<i>225</i>					
<i>Weighted avg</i>	<i>0.42</i>	<i>0.42</i>	<i>0.40</i>	<i>225</i>					

10.4 Model 3: Shapelet-based Model

A Shapelet-based classifier was employed with the primary goal of providing interpretability by explicitly extracting and identifying discriminative subsequences (shapelets):

10.4.1 Performance and Interpretability Analysis

The Shapelet-based model showed the lowest predictive performance, with a Macro F1-Score of 0.2626 and zero recall for both **Medium Low** and **Medium High** classes. The confusion matrix reveals a heavy bias toward the **Medium** class, effectively reducing the 4-class task to a skewed 2-class problem. This highlights a major limitation in the discriminative power of the extracted shapelets for this dataset.

Despite poor prediction results, the shapelet model remains conceptually valuable due to its inherently interpretable nature. Shapelets are designed as compact, representative subsequences capturing temporal patterns linked strongly to target classes, offering intuitive and actionable insights into class-specific time series behaviors.

Figure 23 illustrates four representative shapelets, each associated with a different target class and extracted with a fixed length of 15 timesteps. A brief analysis of these patterns reveals several interesting observations:

- **Medium Low (Shapelet #0):** This shapelet displays a relatively flat trajectory with a sudden, short-lived spike around timestep 12. This could hypothetically reflect a late and isolated increase in box office revenue, possibly tied to a niche event or marketing push, yet not sustained enough to elevate the movie’s overall performance.
- **Medium (Shapelet #1):** This shapelet shows a steep drop after an initially high value, followed by minor fluctuations and a gradual recovery. Such a pattern might be characteristic of a strong initial release dampened by word-of-mouth or negative reviews, stabilizing into a moderate revenue pattern.
- **Medium High (Shapelet #2):** The shapelet exhibits a slow but steady ascent peaking around timestep 6, then a noisy decline. This could reflect movies that build momentum through sustained viewer interest before tapering off, indicative of a more gradual popularity curve.
- **High (Shapelet #3):** This shapelet is marked by a general upward trend with intermediate oscillations, peaking around timestep 11. It may capture a profile of films that sustain and grow their audience over time, potentially reflecting word-of-mouth success or extended cinematic runs.

While these visual patterns are plausible and semantically interpretable, they **did not** translate into discriminative success for the classifier. This discrepancy underscores a key limitation: the challenge lies not only in extracting shapelets that are visually distinctive, but in ensuring that these patterns generalize across multiple instances within each class. Despite extensive parameter tuning (including shapelet count and length) and a dedicated preprocessing pipeline, the model failed to capture consistent motifs that could support reliable classification boundaries.

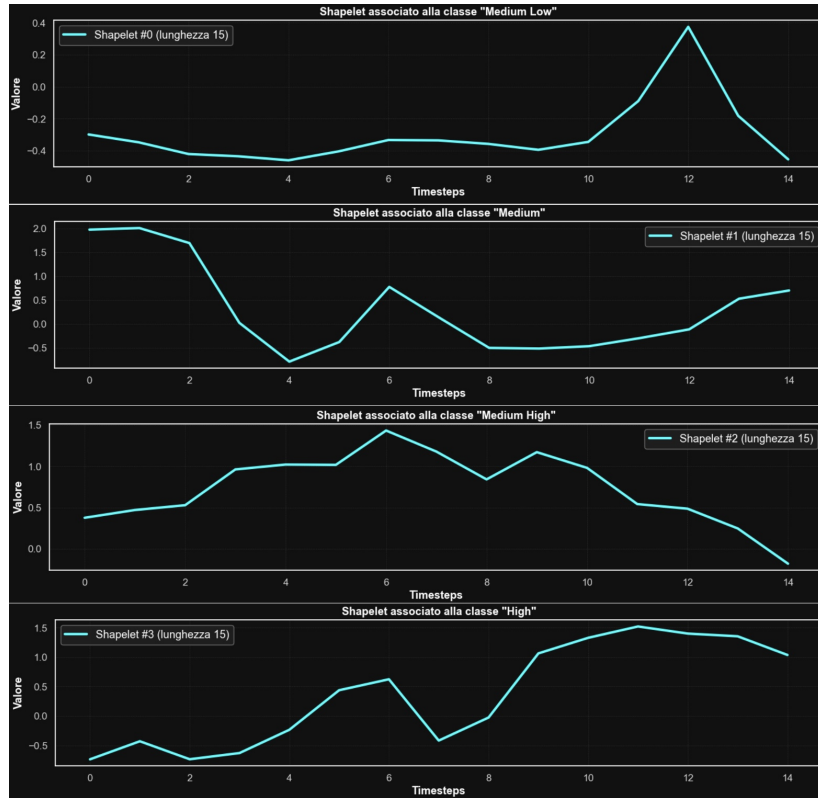


Figure 23: Visualization of the most representative shapelets associated with each of the four target classes.

Table 3: Performance of Shapelet Model on the Test Set

Classification Report					Confusion Matrix				
Class	Prec.	Rec.	F1	Supp.	Predicted →				
					True ↓	ML	M	MH	H
Medium Low	0.00	0.00	0.00	26	Medium Low	0	25	0	1
Medium	0.40	0.88	0.55	78	Medium	0	69	0	9
Medium High	0.00	0.00	0.00	46	Medium High	0	34	0	12
High	0.59	0.43	0.50	75	High	0	43	0	32
<i>Accuracy</i>			<i>0.45</i>	<i>225</i>					
<i>Macro avg</i>			<i>0.25</i>	<i>0.33</i>	<i>0.26</i>	<i>225</i>			
<i>Weighted avg</i>			<i>0.34</i>	<i>0.45</i>	<i>0.36</i>	<i>225</i>			

10.5 Model 4: MiniROCKET

MiniROCKET (RandOm Convolutional KErnel Transform), a recent state-of-the-art approach, was applied to leverage its high accuracy, efficiency, and scalability. The method involved transforming the time series through a vast number of randomized convolutional kernels followed by logistic regression classification.

10.5.1 Performance Analysis

MiniROCKET emerged as the superior model, achieving the highest Macro F1-Score (0.4006) and Balanced Accuracy (0.4384). The detailed classification results revealed a complex interplay of strengths and limitations. Notably, MiniROCKET displayed exceptional recall for **Medium Low** (0.62) but at the expense of low precision (0.28), indicating considerable misclassification from other classes. Consistent with prior models, MiniROCKET also exhibited difficulty in accurately classifying **Medium High** (recall 0.13), although it maintained strong performance in recognizing **High** instances. Such findings suggest a subtle yet crucial bias within MiniROCKET’s classification behavior, necessitating future model tuning and refinement.

Table 4: Performance of MiniROCKET on the Test Set (Corrected)

Classification Report					Confusion Matrix				
Class	Prec.	Rec.	F1	Supp.	Predicted →				
					True ↓	ML	M	MH	H
Medium Low	0.28	0.62	0.38	26	Medium Low	16	9	0	1
Medium	0.44	0.47	0.46	78	Medium	26	37	6	9
Medium High	0.27	0.13	0.18	46	Medium High	10	19	6	11
High	0.66	0.53	0.59	75	High	6	19	10	40
Accuracy			0.44	225					
Macro avg	0.41	0.44	0.40	225					
Weighted avg	0.46	0.44	0.43	225					

10.6 Comparative Analysis and Discussion

Table 18: Final Model Comparison

Model	Macro F1	Balanced Acc.	Fit Time (s)	Predict Time (s)
MiniROCKET	0.4006	0.4384	0.48	0.07
k-NN (Euclidean)	0.3702	0.3786	1.70	0.03
k-NN (DTW)	0.3466	0.3600	182.40	60.04
Shapelet Model	0.2626	0.3278	296.55	0.62

This comparison reveals several key insights:

- Performance versus Computational Complexity:** The more computationally demanding models (DTW, Shapelets) did not demonstrate improved performance over simpler alternatives. MiniROCKET, despite its feature-rich yet computationally efficient structure, provided optimal results, challenging the intuitive correlation between complexity and accuracy.
- Accuracy versus Interpretability:** A classical trade-off emerged distinctly. Shapelets, while potentially highly interpretable, failed to deliver sufficient predictive accuracy. Conversely, MiniROCKET, with its exemplary accuracy, offered minimal interpretative insights, complicating its adoption in interpretability-critical contexts.
- Limitations of DTW:** Despite DTW’s theoretical advantages for capturing temporal variability, it significantly underperformed. This highlights that, in this dataset, shape magnitude and overall trajectory were more crucial discriminative features than temporal alignment.

10.7 Conclusion and Recommendations

MiniROCKET is recommended unequivocally for production environments demanding high predictive accuracy and efficiency. Nevertheless, the model’s limitations, particularly concerning class-specific accuracy biases and interpretability, necessitate supplemental approaches. A hybrid deployment is advisable, utilizing MiniROCKET for general prediction complemented by a refined Shapelet-based model or similar interpretable methodologies for strategic decision-making and business insight generation. This balanced approach promises to deliver robust, accurate classifications while simultaneously supporting the interpretative clarity demanded by business stakeholders.