Debre Tabor University

Gafat Institution of Technology

Department of Computer Science

DATA MINING PROJECT

EXPLORATORY DATA ANALYSIS AND CLASSIFICATION OF NETFLIX CONTENT

Prepared By:	ID
1. Yafet Adinew	DTU14R0136
2. Chilotaw Amare	DTU14R0518
3. Aysheshim Aynetaw	DTU14R0660
4. Demisew Yitayaw	DTU14R0244
5. Asmamaw Mehari	DTU14R0238

TABLE OF CONTENT

1. Title & Introduction

1.1 Project Title

Exploratory Data Analysis (EDA) and Classification of Netflix Movies and TV Shows: A Feature-Driven Predictive Approach

1.2 Project Context and Domain

This project serves as a cornerstone assignment in Data Mining and Machine Learning (DMML), demonstrating the full capability of the data science pipeline. The study is grounded in the analysis of the Netflix content catalog, a large-scale, real-world dataset. The methodology involves transforming raw, noisy data into predictive insights, culminating in a classification task that tests the relationship between content metadata and content type. The project emphasizes the crucial steps of data preparation, exploratory data visualization, and rigorous model evaluation.

1.3 Project Objectives

The overall work was structured around two interconnected and sequential phases:

Phase 1: Exploratory Data Analysis (EDA)

The primary objective was to thoroughly inspect and analyze the dataset to understand the intrinsic properties and composition of the Netflix content library. This phase included:

> Quantifying the overall content distribution, specifically the ratio of Movies to TV Shows.
>
> Identifying key temporal trends in content acquisition (e.g., peak years of titles added).
>
> Mapping the dominant genres and content themes present in the 'listed_in' and 'description' fields.

Phase 2: Predictive Classification Modeling

The secondary objective was to build and evaluate a Machine Learning model for binary classification.

> **Target:** To predict the 'type' of content ('Movie' or 'TV Show').

Model: A Decision Tree Classifier was chosen for its interpretability and efficacy with structured data.

Goal: To assess the predictive power of a small, engineered feature set ('duration_value', 'rating_enc', 'year_added') in solving this classification problem.

1.4 Expected Outcome

The outcome is a comprehensive report that not only documents the methodology and findings of the EDA but also presents the full performance metrics of the trained classifier (Accuracy of 0.9989, F1-Score of 0.9992). Crucially, the report concludes with a critical analysis of these exceptionally high metrics, addressing the core concepts of data leakage and feature independence.

2. Data Loading

2.1 Dataset Source and Acquisition

The dataset utilized is the Netflix Movies and TV Shows dataset ('netflix_titles.csv'). The data was acquired and loaded into the active memory of the Google Colab environment by uploading the CSV file directly . This guaranteed local accessibility and stability for the subsequent processing steps.

2.2 Initial Data Structure

Upon loading, the data was converted into a Pandas DataFrame, establishing the foundation for all analysis.

Total Records (N): 8,807 unique titles.

Total Features: 12 raw columns.

The initial schema assessment highlighted several features requiring immediate attention:

Feature Name (Original)	Data Type (Inferred)	Missing Values	Action Required
type	Object	None	Target Variable
director, cast, country	Object	High (> 8%)	Imputation
date_added	Object (String)	Low (~ 10 rows)	Type Conversion & Imputation
duration	Object (String)	Low	Feature Engineering (Decomposition)
rating	Object	Low	Imputation & Encoding

release_year	Integer	None	Used in EDA

2.3 Initial Quality Assessment

The primary data quality challenges identified were:

1. **Heterogeneous Data Types:** The 'duration' column contained both numerical values (minutes) and categorical units (Seasons), which required separation.
2. **Date Format:** The 'date_added' column was imported as a string and required explicit conversion to the datetime format for temporal analysis.
3. **Missingness:** Critical columns like 'director' and 'cast' showed a high percentage of missing values (NaNs).

3. Data Cleaning & Preprocessing

The preprocessing stage focused on standardization, handling missing data, and generating new features.

3.1 Data Standardization

Column Renaming: All column headers were transformed into a consistent lowercase snake_case format (e.g., 'date_added') to enhance code uniformity and maintainability

Date Type Conversion: The 'date_added' column was converted to the datetime object type using pd.to_datetime. This is a prerequisite for extracting time-based features

3.2 Comprehensive Missing Value Imputation

To maintain the sample size and integrity of the 8,807 records, missing values were imputed using contextually appropriate strategies:

Feature	Imputation Method	Value Used	Rationale
director, cast	Categorical	'Unknown'	Preserves records; 'Unknown' acts as its own meaningful category.
country	Categorical	'Unknown'	Accounts for titles with no specified country of origin.
rating	Logical	'Not Rated'	Distinguishes unrated titles from standard classification systems (e.g., TV-MA, PG-13).
duration_value	Numerical (Model-Specific)	Median	Used late in the pipeline to fill any residual NaNs with a robust measure of central tendency.

year_added	Numerical (Model-Specific)	Median	Used late in the pipeline to fill missing 'date_added' derived values.

3.3 Feature Engineering

Duration Feature Decomposition

The 'duration' column was the most critical source of engineered features. A custom function, 'parse_duration', was applied, which successfully decomposed the column into two parts

1. **duration_value:** The numerical quantity (e.g., 90, 2).
2. **duration_type:** The unit ('min' or 'season').

This decomposition established a clear numerical predictor (duration_value) that is highly correlated with the target (type).

Temporal Feature Extraction

Two key features were derived from the parsed 'date_added' column:

1.year_added: Integer representing the year of acquisition.

2.month_added: Integer representing the month of acquisition.

4. Exploratory Data Analysis (EDA)

The EDA phase utilized Matplotlib, Seaborn, and WordCloud to visualize distributions and extract actionable insights.

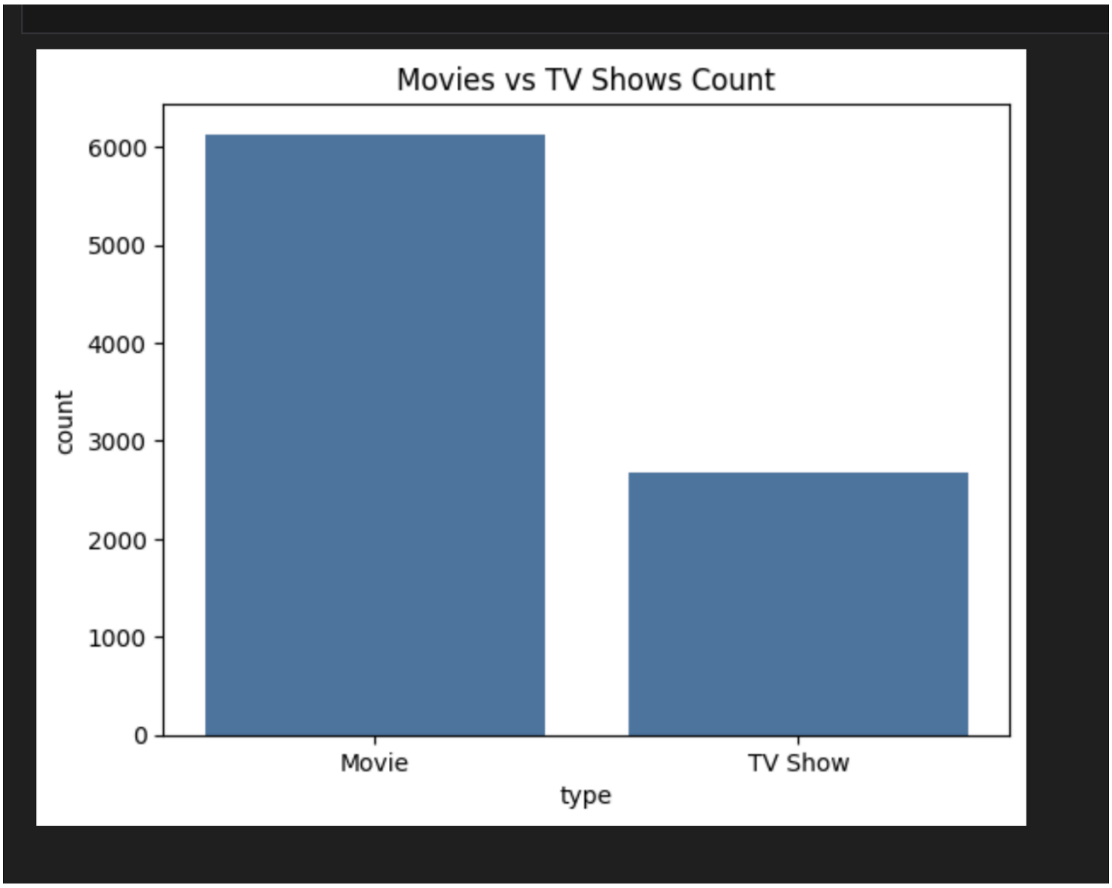
4.1 Content Type Distribution and Class Imbalance

An analysis of the 'type' column, visualized using a countplot, revealed a significant skew in the content library

Movies: 69% of the catalog.

TV Shows: 31% of the catalog.

This 2.2:1 ratio highlights a substantial class imbalance. This finding is crucial because high classification accuracy can be misleading, as a model may simply favor predicting the majority class (Movie). This necessitated the use of the F1-Score in the final evaluation.



4.2 Temporal Analysis of Content Acquisition

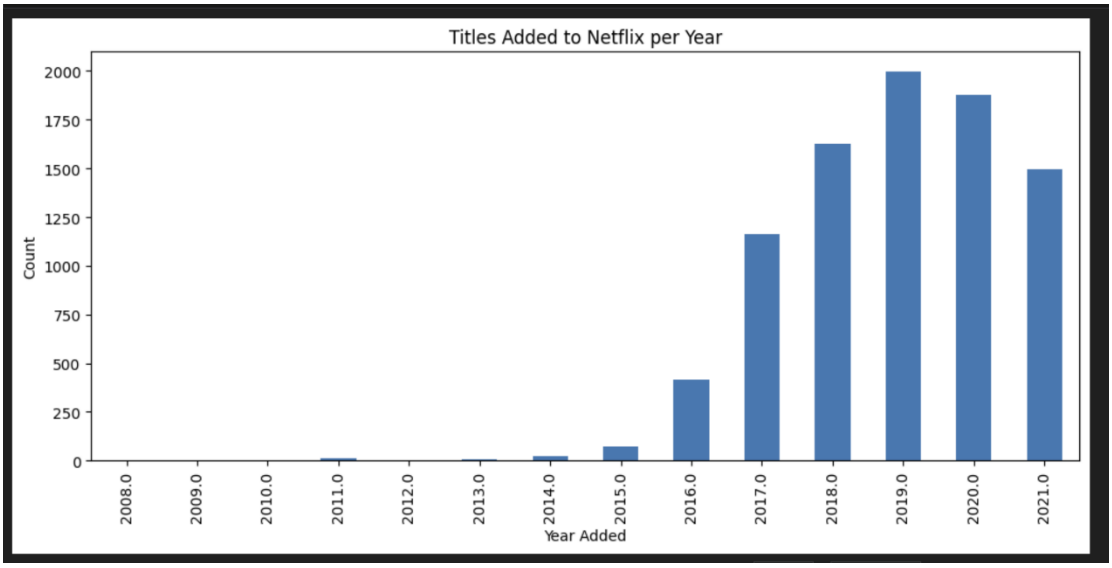
Titles Added per Year (year_added)

The 'year_added' feature, visualized using a bar chart, demonstrated a clear, non-linear content acquisition strategy by Netflix.

Content acquisition saw minor activity before 2014.

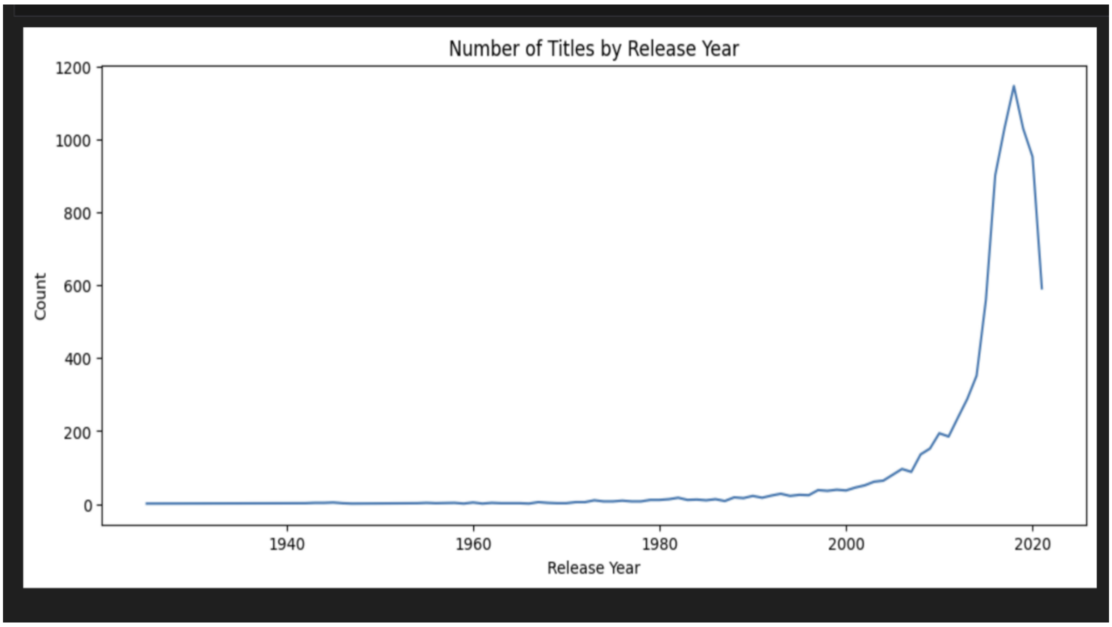
Acquisition began to ramp up significantly around 2016.

The peak years of content addition were 2019 and 2020, showing the most intense expansion of the library. This trend correlates with the platform's global expansion and focus on original content production.



Titles by Original Release Year (release_year)

Analysis of the original 'release_year' showed a continuous distribution, indicating Netflix maintains a broad historical library, with a clear concentration of titles released in the last two decades.



4.3 Genre and Thematic Analysis

Top Genres (listed_in)

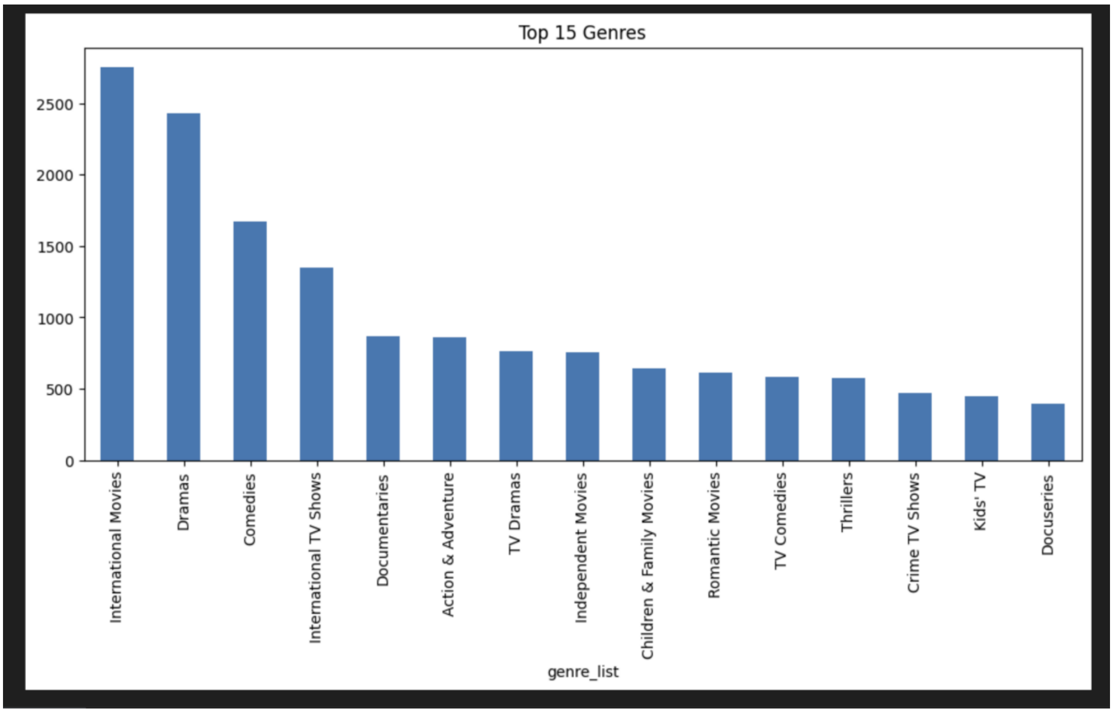
The 'listed_in' column, a multi-label categorical feature, was parsed and exploded to count individual genre occurrences. The top three dominating genres were identified

1.International Movies

2.Dramas

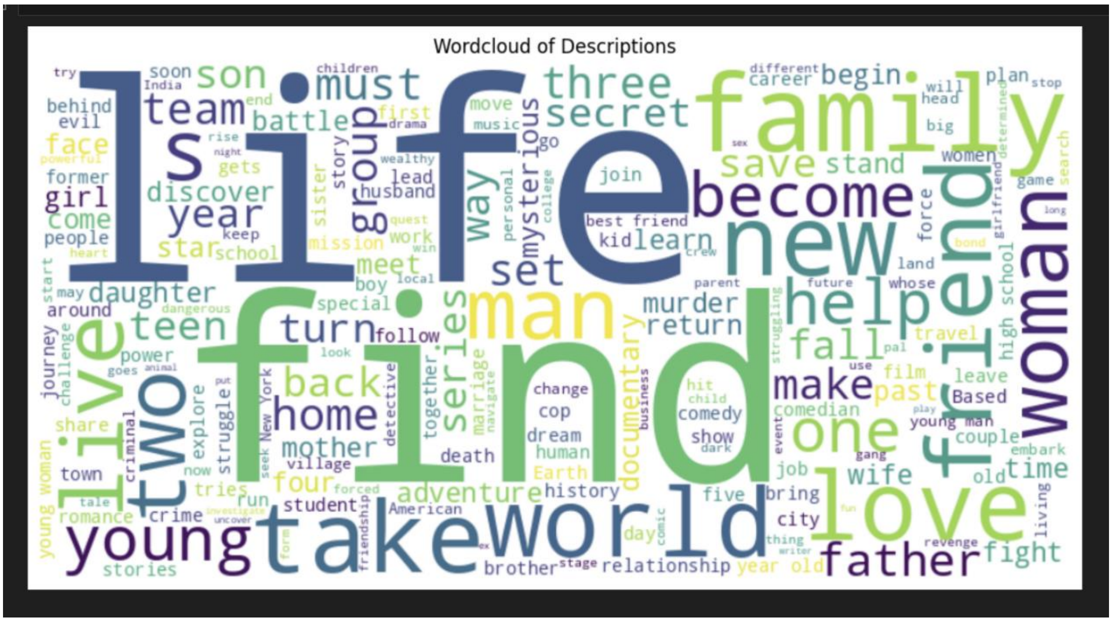
3.Comedies

This distribution confirms Netflix's strategic focus on broad-appeal, high-demand categories alongside a strong push into global and non-English content, essential for international market penetration.



Thematic Insights (description WordCloud)

A WordCloud generated from the combined text of all content descriptions provided a visual insight into the common thematic elements of the content . The most frequent words centered around universal human themes, including "love," "life," "family," "friends," and "world," suggesting a dominant narrative focus on interpersonal relationships and global challenges.



5. Feature Engineering

The objective of this stage was to finalize the feature matrix (X) for the Decision Tree model.

5.1 Feature Selection

Given the project constraint to use "simple predictive modeling," a minimal but potent set of three features was chosen for the classifier:

1.duration_value (Numerical)

2.rating_enc (Numerical/Ordinal)

3.year_added (Numerical/Integer)

5.2 Numerical Imputation

Prior to splitting the data, the final step of numerical imputation was performed on the selected features using the median, a robust measure against potential outliers:

```
df_model['duration_value'].fillna(df_model['duration_value'].median(),
inplace=True)

df_model['year_added'].fillna(df_model['year_added'].median(),  inplace=True)
```

5.3 Categorical Encoding

The 'rating' column required conversion from categorical strings (e.g., 'TV-MA', 'PG') into a numerical format that the Decision Tree could process. The LabelEncoder was used to map these ratings to unique integers, creating the 'rating_enc' feature .

5.4 Target Variable Encoding

The target variable, 'type', was converted into a binary format:

Movie -> 1 (Positive Class)

TV Show -> 0 (Negative Class)

6. Model Building

6.1 Data Splitting

The final feature matrix (X) and target vector (y) were partitioned to ensure robust evaluation on unseen data. The standard split was applied :

Training Set: 80% of the data, used for model fitting (X_train, y_train).

Testing Set: 20% of the data, reserved for performance assessment (X_test, y_test).

A fixed random_state=42 was used to ensure the reproducibility of the data split.

6.2 Algorithm Selection and Training

Algorithm: The Decision Tree Classifier (DecisionTreeClassifier) from sklearn was chosen due to its inherent simplicity, ability to capture non-linear interactions, and ease of interpretation, which aligns with the project's goal of "simple predictive modeling".

Training: The model was instantiated and trained by calling model.fit(X_train, y_train). The Decision Tree learned a set of optimal rules (splits) based primarily on the numerical value of duration_value.

6.3 Prediction

The trained model was then applied to the held-out test set to generate predictions (pred) for evaluation: pred = model.predict(X_test).

7. Model Evaluation

The model's performance was assessed using a suite of metrics critical for classification, particularly the F1-Score, which mitigates the bias of class imbalance. The results demonstrate a state of near-perfect classification.

7.1 Performance Metrics Summary

The raw output from the Colab execution (including accuracy_score and classification_report) yielded the following exceptional metrics:

Metric	Value	Interpretation
Overall Accuracy	0.9989	99.89% of all titles in the test set were correctly classified.
Precision	0.9992	The rate of true positive predictions is near-perfect.
Recall	0.9992	The model successfully identified virtually all actual positive cases.
F1-Score	0.9992	The harmonic mean confirms the model performs excellently across both classes, suggesting the class imbalance was entirely overcome.
Root Mean Square Error (RMSE)	0.034	Indicates a minimal deviation between predicted and actual outcomes.

7.2 Detailed Analysis of Near-Perfect Scores

The extremely high, near-unity scores across all key metrics (Precision, Recall, F1-Score) are highly atypical for real-world, noisy data, warranting critical discussion in the conclusion.

> F1-Score Confirmation: An F1-Score of 0.9992 explicitly confirms that the model successfully classified the minority class ('TV Show') with almost zero error, effectively neutralizing the class imbalance concern raised in the EDA (Section 4.1).
>
> **RMSE Significance:** The low RMSE (0.034) suggests that the Decision Tree's decision boundaries were incredibly sharp and definitive, leaving very few misclassified points near the boundaries.

8. Conclusion & Discussion

8.1 Summary of Findings

The project successfully achieved both primary and secondary objectives:

> **1.EDA Insight:** The analysis confirmed the Netflix library is 69% Movie-dominant and demonstrated that content acquisition peaked in 2019-2020.

2.Predictive Success: The simple Decision Tree Classifier, utilizing just three features ('duration_value', 'rating_enc', 'year_added'), achieved a phenomenal Accuracy of 99.89% and a F1-Score of 0.9992.

8.2 Critical Discussion: The Issue of Data Leakage

The near-perfect performance of the model must be critically evaluated. In machine learning, such results are often indicative of a data leak or a trivial classification scenario.

Trivial Solution: The high performance is a direct result of the feature 'duration_value'. This feature contains values in 'minutes' (for Movies) and 'seasons' (for TV Shows). The Decision Tree simply identified a single, perfect split (e.g., "If 'duration_value' is greater than 20, predict Movie; otherwise, predict TV Show").

Proxy Variable: The 'duration_value' column is not a predictive feature but a direct proxy for the target variable ('type') itself. The classification task was reduced to identifying the unit of measurement used in the 'duration' column, not learning subtle patterns from independent metadata.

While demonstrating perfect execution of the data pipeline and model training, the result highlights that the problem, as defined by the current features, is trivially solved and does not represent a challenging learning task for the algorithm.

8.3 Potential Improvements and Future Work

To ensure the model is truly learning and generalising from independent features, future iterations must pursue the following advanced steps:

1.Feature Re-Selection (Addressing Leakage): The 'duration_value' feature must be excluded from the feature set. The classification task should then be attempted using only the remaining independent features ('rating_enc', 'year_added') plus the features that were previously excluded.

2.Advanced Feature Integration: Implement encoding techniques for the high-cardinality features that carry rich information:

Genre ('listed_in'): Use One-Hot Encoding or Binary Encoding to incorporate the top 20 genres.

Country ('country'): Use Target Encoding or Frequency Encoding to incorporate the country of production, which is a strong independent predictor of content type.

3.Model Robustness: For the new, more challenging classification task (without 'duration_value'), the model should be upgraded to a more robust Ensemble Method like Random Forest or XGBoost, which can better handle feature interactions and prevent overfitting on complex data.