30/03/2025

Project By: Erez Levy

ML Model Prediction - TMDB

Te TMDB Dataset The TMDB (The Movie Database) is a widely-used resource for movie and TV show data, providing valuable information such as ratings, plot summaries, and more.

This dataset contains a collection of 150,000 tv shows from the TMDB database, collected and cleaned.

Using the Google Colab platform I suggest a predictive model to determine the success of a TV show based on features like vote count, vote average, and popularity. We'll approach this as a regression problem, where we predict a continuous success metric, and we'll use multiple regression models

The Project Overview:

This dataset opens up a wide range of possibilities for data analysts and data scientists. Here are some ideas to get you started:

Explore trends in TV show popularity based on vote count and average. Analyze TV show genres to identify the most popular genres or combinations of genres. Investigate the relationship between TV show ratings and the number of seasons and episodes. Build a recommendation system that suggests TV shows based on a user's favorite genres or languages. Predict the success of a TV show based on features like vote count, average, and popularity. Identify the most prolific TV show creators or production companies based on the number of shows they have created. Explore the distribution of TV show run times and investigate whether episode duration affects the overall ratings. Investigate TV show production trends across different countries and networks. Analyze the relationship between TV show language and popularity, and investigate the popularity of non-English shows. Track the status of TV shows (in production or not) and analyze their popularity over time. Develop a language analysis model to identify sentiment or themes from TV show overviews.

The Approach:

1. Data Preparation :

- **Uniting Tables:** Combine different data sources into a unified dataset, ensuring consistency and alignment of data points**.**
- **Reduce Large Categories:** Simplify datasets by reducing the number of categories in categorical variables where necessary. This might involve grouping smaller categories into an "Other" category.
- **Clean Text:** Perform text cleaning by removing stop words, punctuation, special characters, and normalizing text (e.g., lowercasing). • Transform/Manipulate data

2. Exploratory Data Analysis (EDA):

- **Instant Reports:** Generate initial reports to get a quick overview of the dataset, including summary statistics and distributions**.**
- **Descriptive Analysis:** Use statistical methods and visualizations to understand the data, identify patterns, trends, and potential issues.
- **Correlation and other relationship Analysis:** between different features to identify multicollinearity and feature importance.

3. Data Cleansing :

- **Outliers:** Detect and handle outliers which might skew the data analysis and model performance. This might involve removing, transforming, or capping outlier values. •
- **Missing Values:** Address missing data through imputation methods (e.g., mean, median, mode) or by removing incomplete records, depending on the context and importance of

the missing information.

4. Feature Engineering & Feature Selection

- **Enriching:** Create new features from existing ones to better capture the underlying patterns in the data. This can involve mathematical transformations, aggregations, or domain-specific knowledge.
- **Normalization/Standardization:**Scale numerical features to ensure they have similar ranges, which helps certain algorithms perform better**.**
- **Feature Selection:** Choosing the most valuable features that predict the target value by running penalty models.

5. One-Hot Encoding

- Convert categorical variables into a format that can be provided to ML algorithms to do a better job in prediction. This typically involves creating binary columns for each category in a categorical feature.

6. Model Selection and Fine Tuning

- **Model Selection:** Choose appropriate machine learning models based on the problem at hand (e.g., regression, classification, clustering).
- **Cross-Validation:** Use cross-validation techniques to assess model performance and ensure robustness.
- **Fine Tuning:** Optimize model hyperparameters to improve performance, typically using methods like grid search, random search, or Bayesian optimization.

7. Model Evaluation

- **Performance Metrics:** Evaluate the model using appropriate metrics such as accuracy, precision, recall, F1-score, ROC-AUC for classification, or RMSE, MAE for regression.
- **Validation:** Validate the model on a separate validation set to ensure it generalizes well to unseen data.

Important Considerations:

Feature Engineering: Carefully select and engineer features from the TMDB data that you think will be most relevant to your chosen target variable.

Model Selection: Experiment with different machine learning models (regression for popularity or ratings, classification for status/renewal) to find the best performer.

Evaluation: Use appropriate metrics (like RMSE for regression or accuracy for classification) to assess the performance of your predictive model.

The Methods

1.Data Collection & Preperation:

The first step in dataset preparation is data collection. This involves gathering all relevant data that will be used to train and test the machine learning model. Sources might include:

- **Databases:** Structured data from existing databases, such as customer profiles and transaction records.
- **APIs:** Data from public or private APIs, like social media or customer feedback platforms.
- **CSV/Excel Files:** Data stored in spreadsheets, possibly containing historical customer interaction data.
- **Web Scraping:** Extracting data from websites if necessary, to gather competitive market analysis.

The Steps :

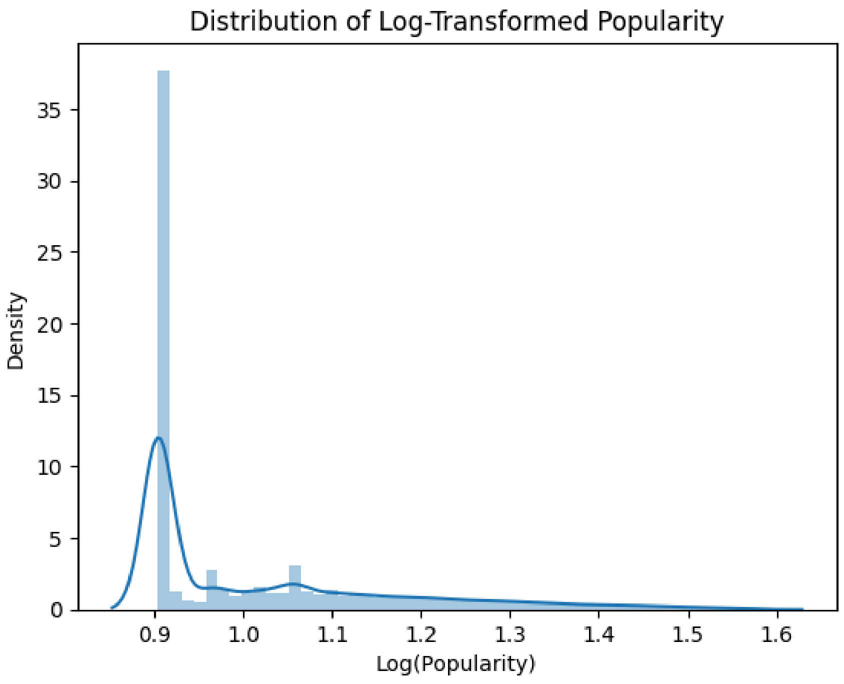
- Uploading the TMDB Original DataSet
- Reduce Large Categories:Dropping unnecesery Dataset columns
- Cleaning the Dataset tmdb as follow:
  - **Stop Words:**
  - These are common words (like "the," "a," "is," "and") that often don't add much meaning to text analysis. Removing them can reduce noise and improve processing efficiency.
  - **Punctuation:**
  - Punctuation marks (like commas, periods, question marks) can interfere with text analysis. Removing or standardizing them helps in focusing on the actual words.
  - **Special Characters:**
  - These include symbols, numbers, and other non-alphanumeric characters .
- Transform/Manipulate data-Consolidating rows with the same 'name' column
- Removes numbers from the 'name' using regular expressions
- Split tmdb_eda into 2 sub_Datasets:
  - tmdb_IP for 'in_production' dataset
  - tmdb_NP for ‘not in production’ Dataset
- Identifying & Replacing rare 'languages' into 'other'
- Droping rows where 'status' is 'Canceled' from the Dataset tmdb_eda
- Dropping the 'status' column from tmdb.eda Dataset
- Identifying & Replacing rare 'networks' into 'other
- Identifying & Replacing rare 'geners' into 'other'
- Identifying & Replacing rare 'production_companies' into 'other'
- Drop 'in_production' column from tmdb_eda dataset
- Data Preparation - Download the tmdb.NP as pickle file to my Google Drive

Target Value prediction

Before diving into data preparation, it's essential to have a clear understanding of the problem the model aims to solve. This helps in identifying the relevant data features that will be crucial for the model's predictive capabilities.My project focuses on predicting customer churn in a telecommunications company, making it vital to select features such as customer usage patterns and contract details.

Based on the potential insights and business value, I would suggest focusing on predicting the popularity value.

Target Value-Popularity histogram:



It is a complex and dynamic metric that reflects overall success. Vote Average (Rating): It captures audience satisfaction and critical acclaim. Both of these targets have valuable real-world implications and can be approached with a variety of machine learning models.

2. Exploratory Data Analysis (EDA):

Descriptive statistics are used to summarize and describe the main features of a dataset. They provide a concise overview of the data, helping you understand its central tendency, dispersion, and shape.

1. Measures of Central Tendency:

- **Mean:** The average of all values in a dataset.

- **Median:** The middle value when the data is sorted.
- **Mode:** The most frequent value in a dataset.

These measures tell you where the "center" of your data lies.

2. Measures of Dispersion (or Variability):

- **Range:** The difference between the maximum and minimum values.
- **Variance:** The average of the squared differences from the mean.
- **Standard Deviation:** The square root of the variance.

These measures describe how spread out the data is. A higher standard deviation indicates more variability.

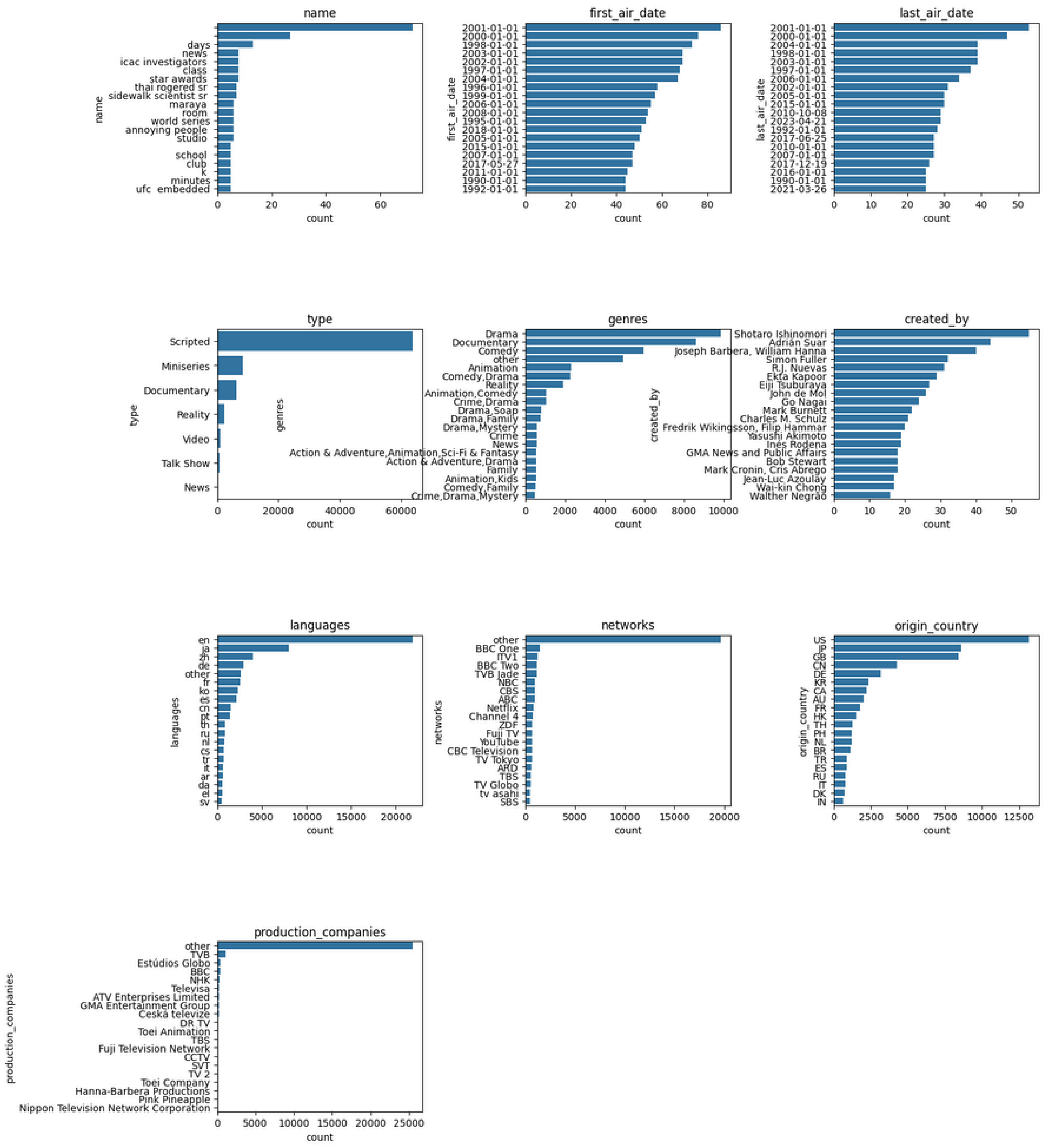
3. Shape of Distribution:

- **Skewness:** Measures the asymmetry of the data distribution.
- **Kurtosis:** Measures the "peakedness" of the data distribution.

These measures provide insights into the overall shape of the data distribution.

4.Instant Reports:

Generate initial reports to get a quick overview of the dataset, including summary statistics and distributions.



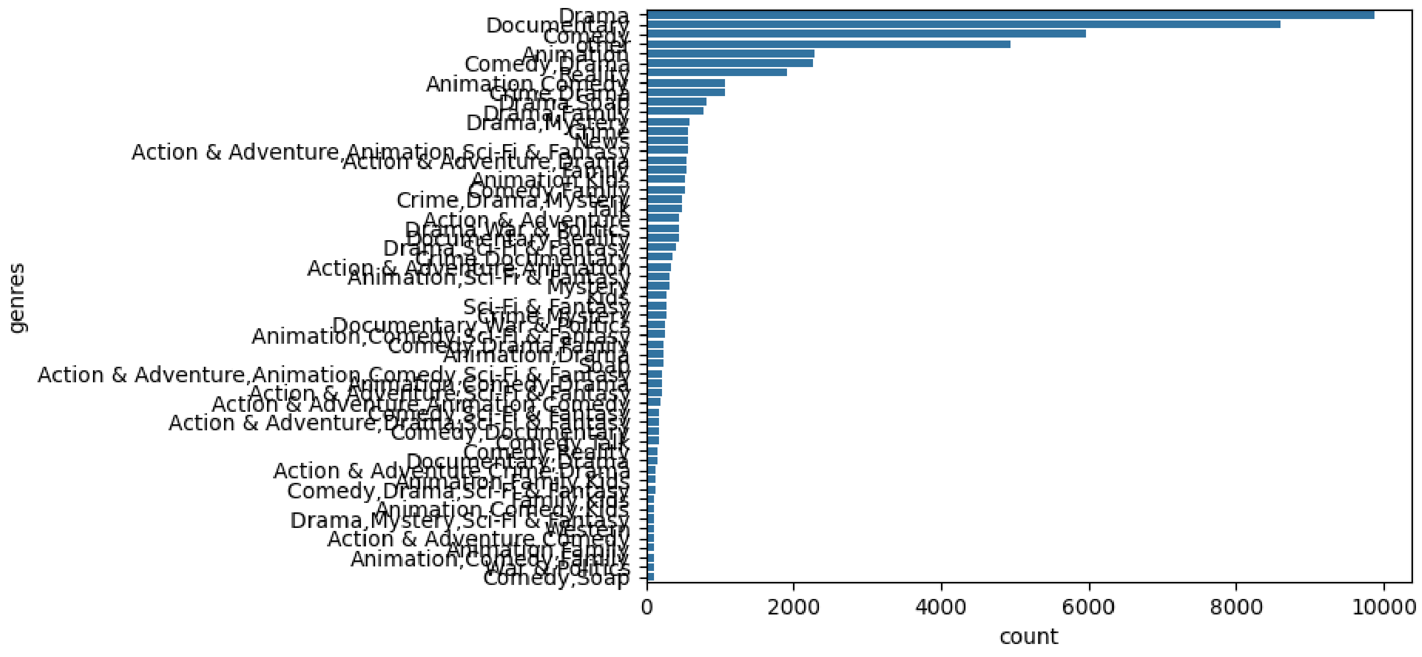
Box Plots and Outliers

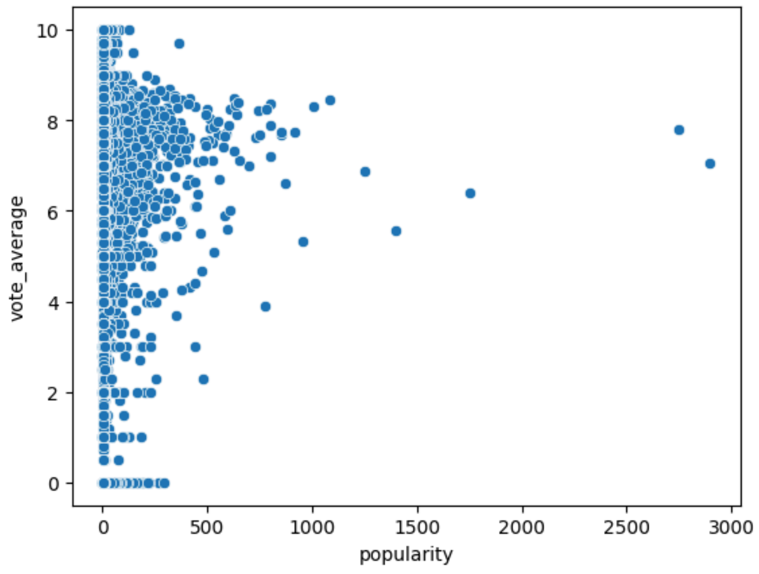
A box plot (also called a box-and-whisker plot) is a graphical representation of the distribution of a dataset. It displays the following key elements:

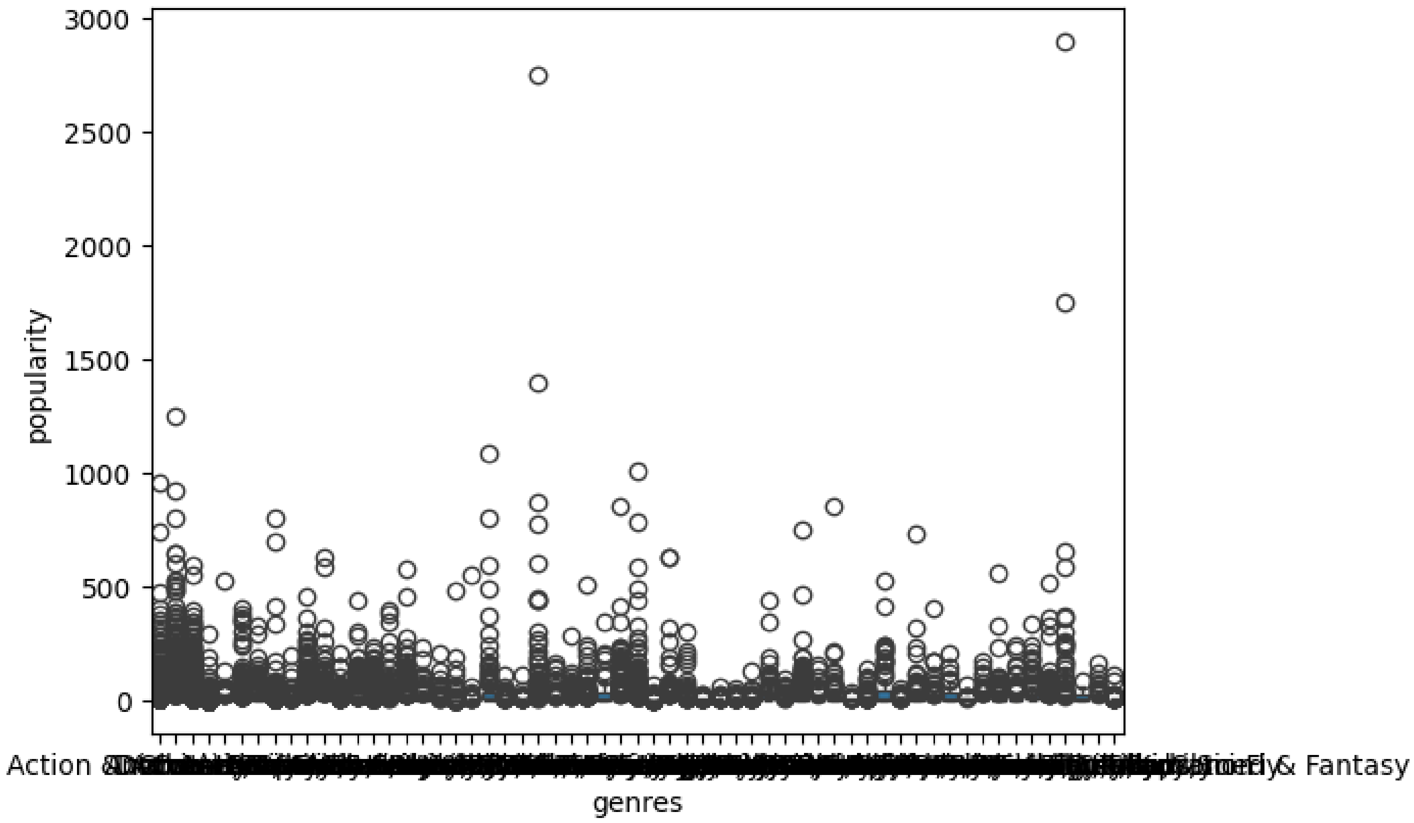
- **Box:** Represents the interquartile range (IQR), which contains the middle 50% of the data.
- **Line inside the box:** Represents the median (the middle value).
- **Whiskers:** Extend from the box to the minimum and maximum values within a certain range.
- **Outliers:** Data points that fall outside the whiskers are considered potential outliers and are plotted as individual points.

Descriptive Analysis:

Use statistical methods and visualizations to understand the data, identify patterns, trends, and potential issues.



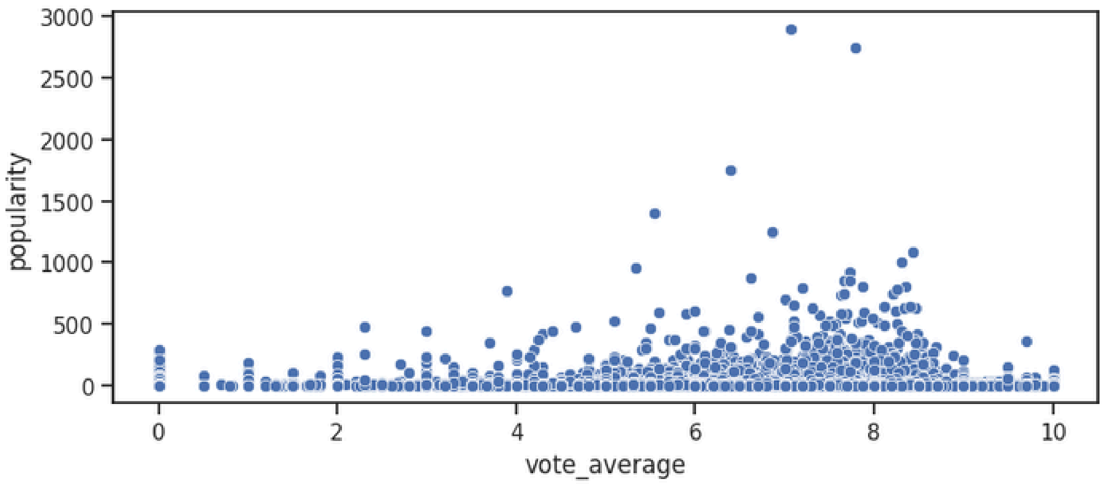




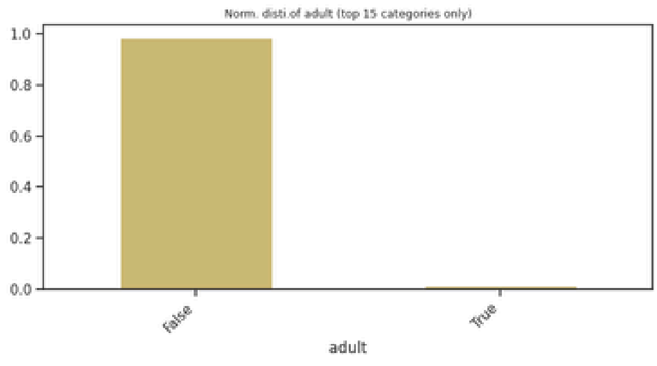
Descriptive Statistics

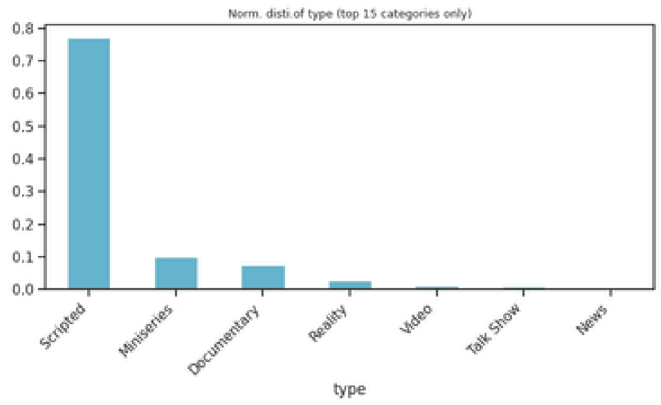
Descriptive statistics are used to summarize and describe the main features of a dataset. They provide a concise overview of the data, helping you understand its central tendency, dispersion, and shape.

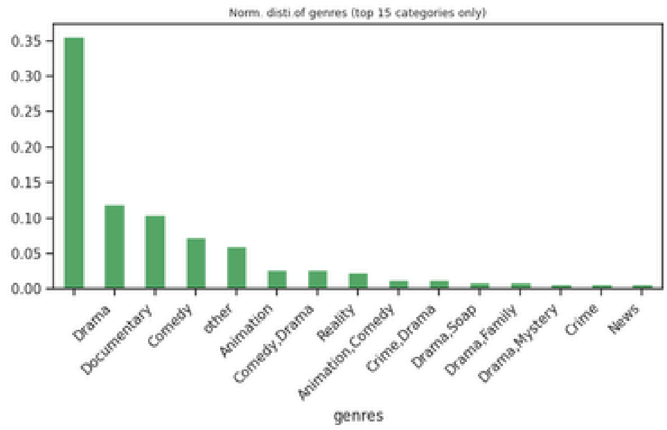
Pair-wise Scatter Plot of all Continuous Variables

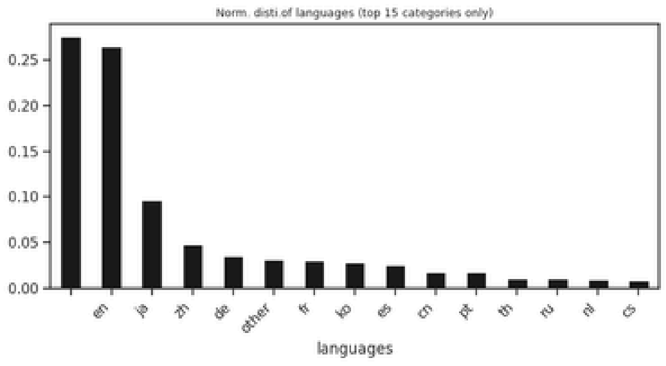


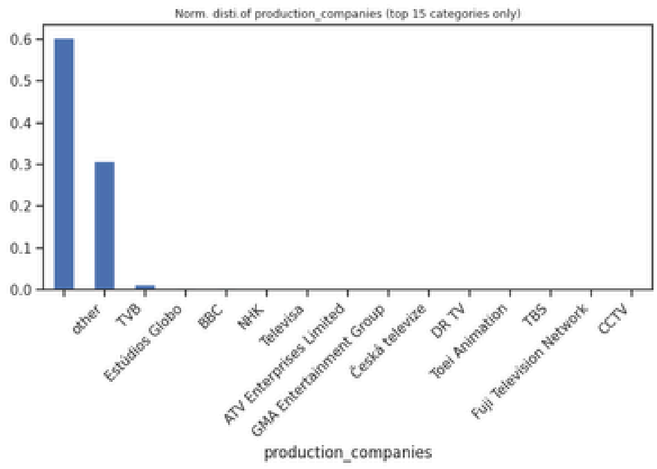
Histograms and Normalized distribitions of all variables

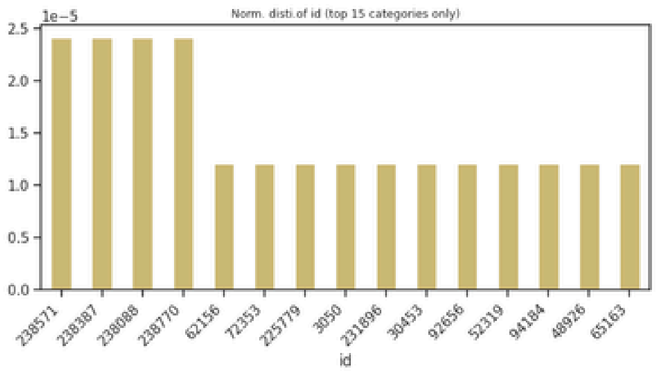


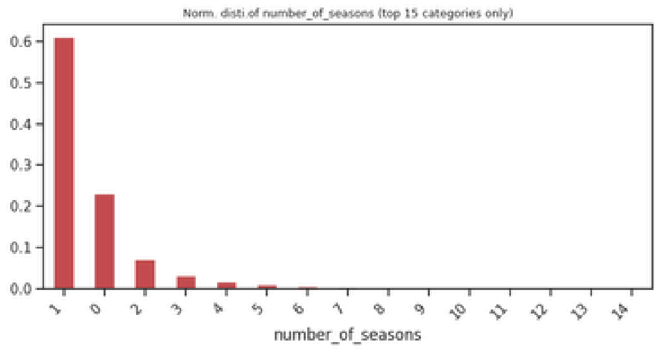


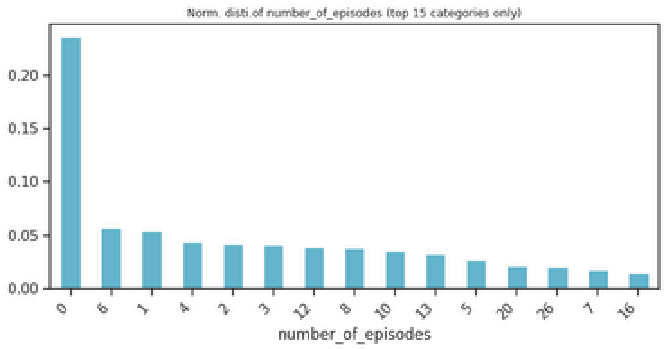


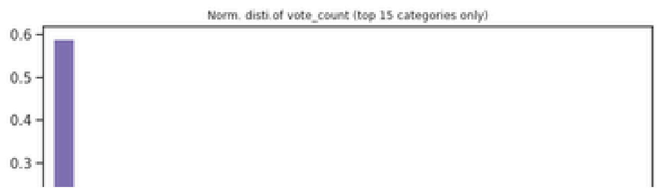


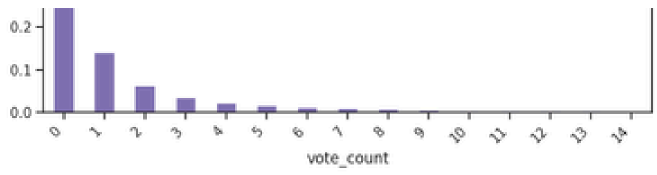


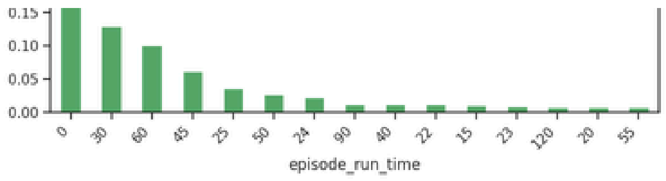




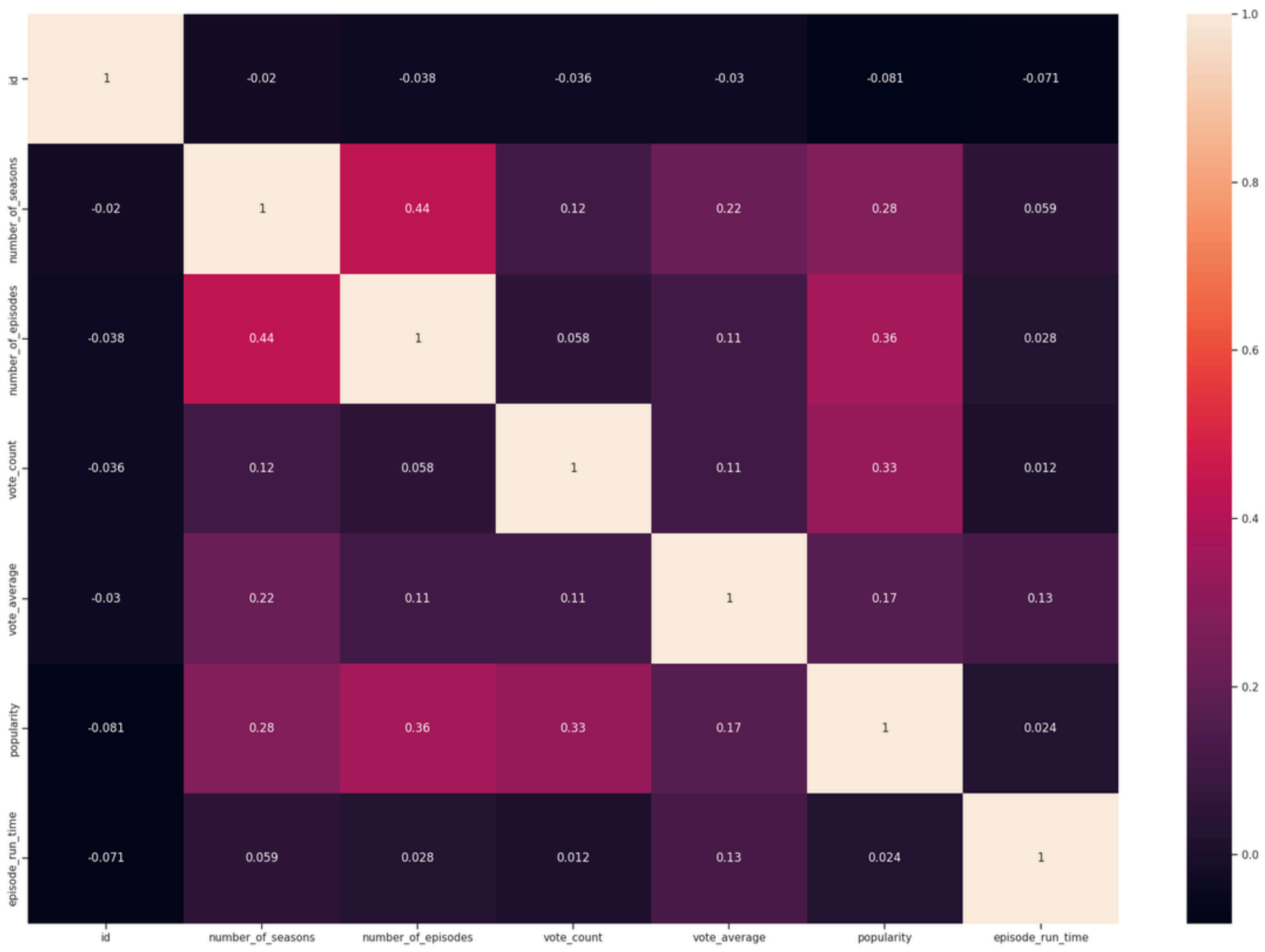






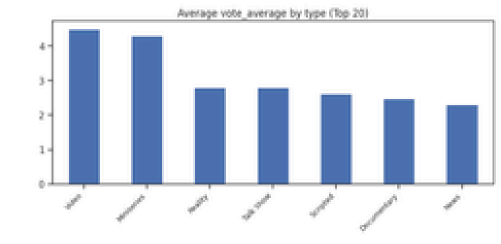


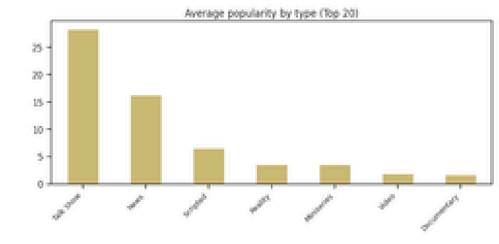
Heatmap of all Numeric Variables including target:

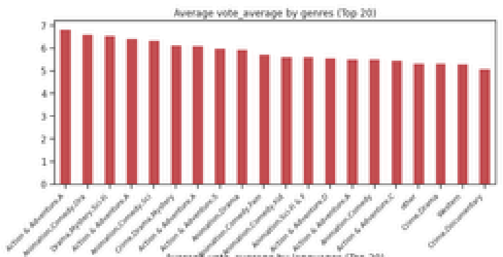


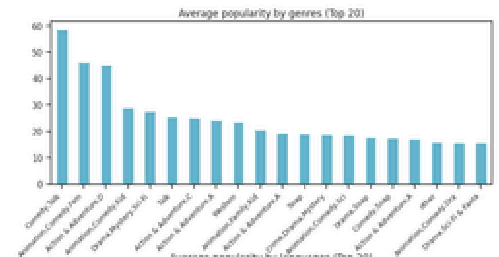
	id	number_of_seasons	number_of_episodes	vote_count	vote_average	popularity	episode_run_time
id	1	-0.02	-0.038	-0.036	-0.03	-0.081	-0.071
number_of_seasons	-0.02	1	0.44	0.12	0.22	0.28	0.059
number_of_episodes	-0.038	0.44	1	0.058	0.11	0.36	0.028
vote_count	-0.036	0.12	0.058	1	0.11	0.33	0.012
vote_average	-0.03	0.22	0.11	0.11	1	0.17	0.13
popularity	-0.081	0.28	0.36	0.33	0.17	1	0.024
episode_run_time	-0.071	0.059	0.028	0.012	0.13	0.024	1

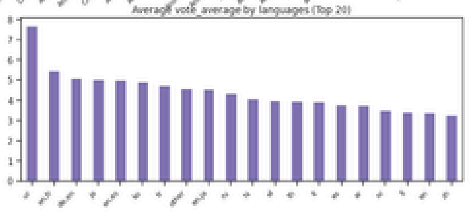
Bar plots for each Continuous by each Categorical variable

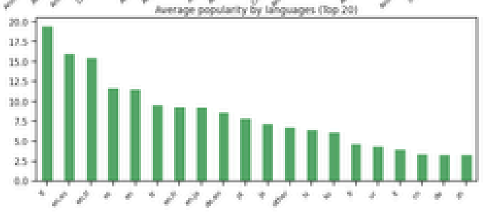


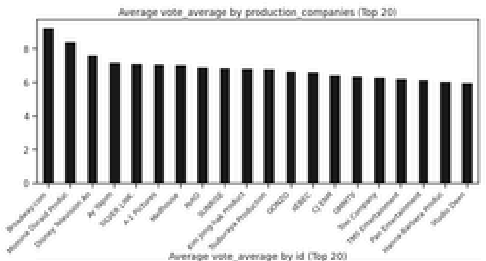


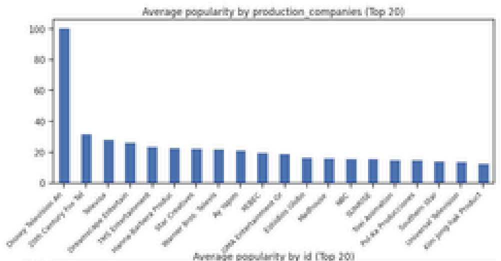


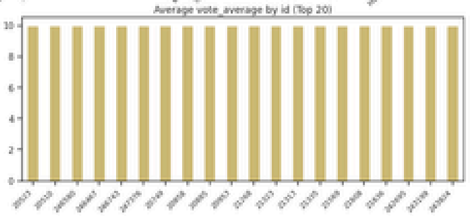


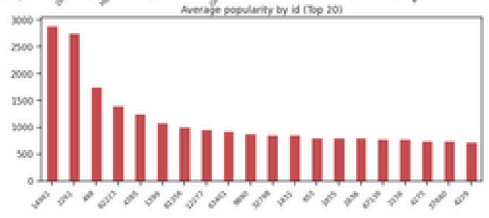


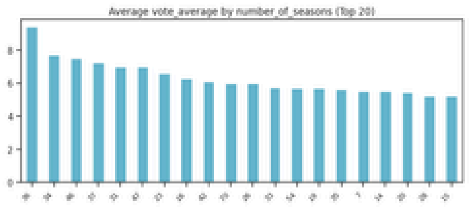


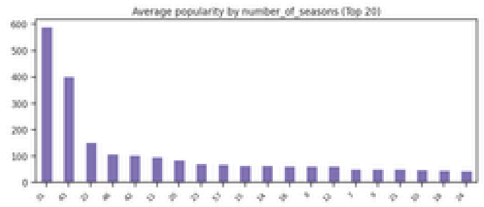


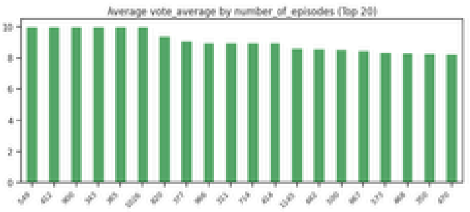


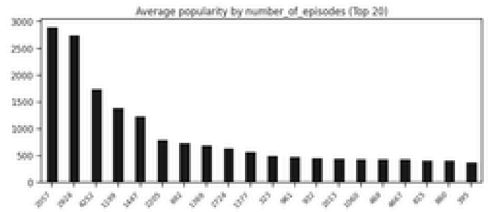


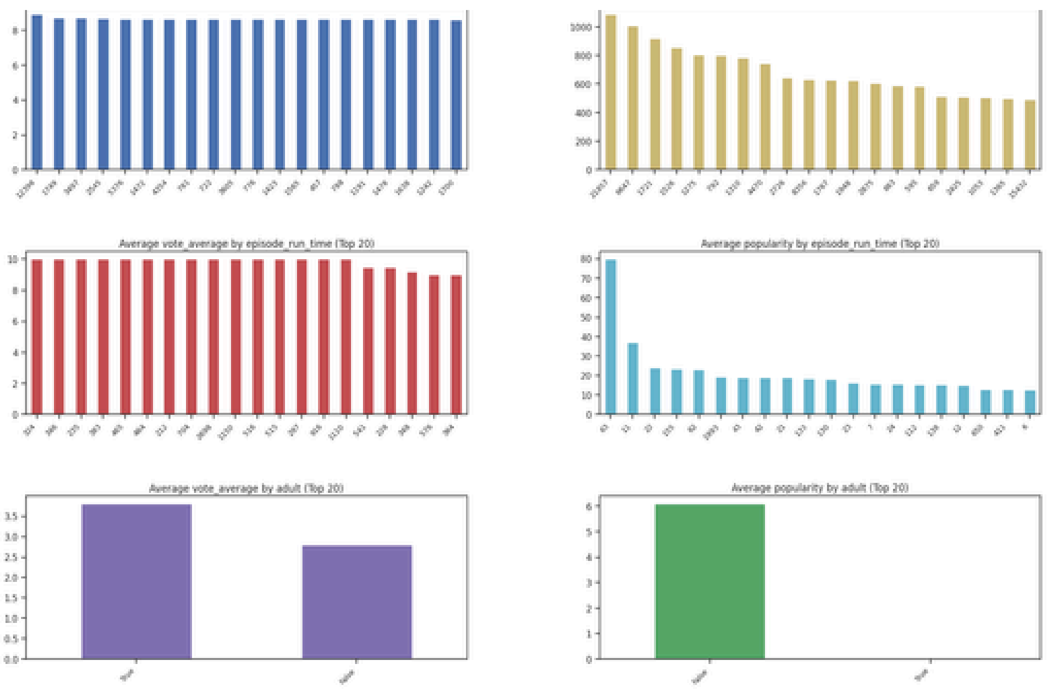








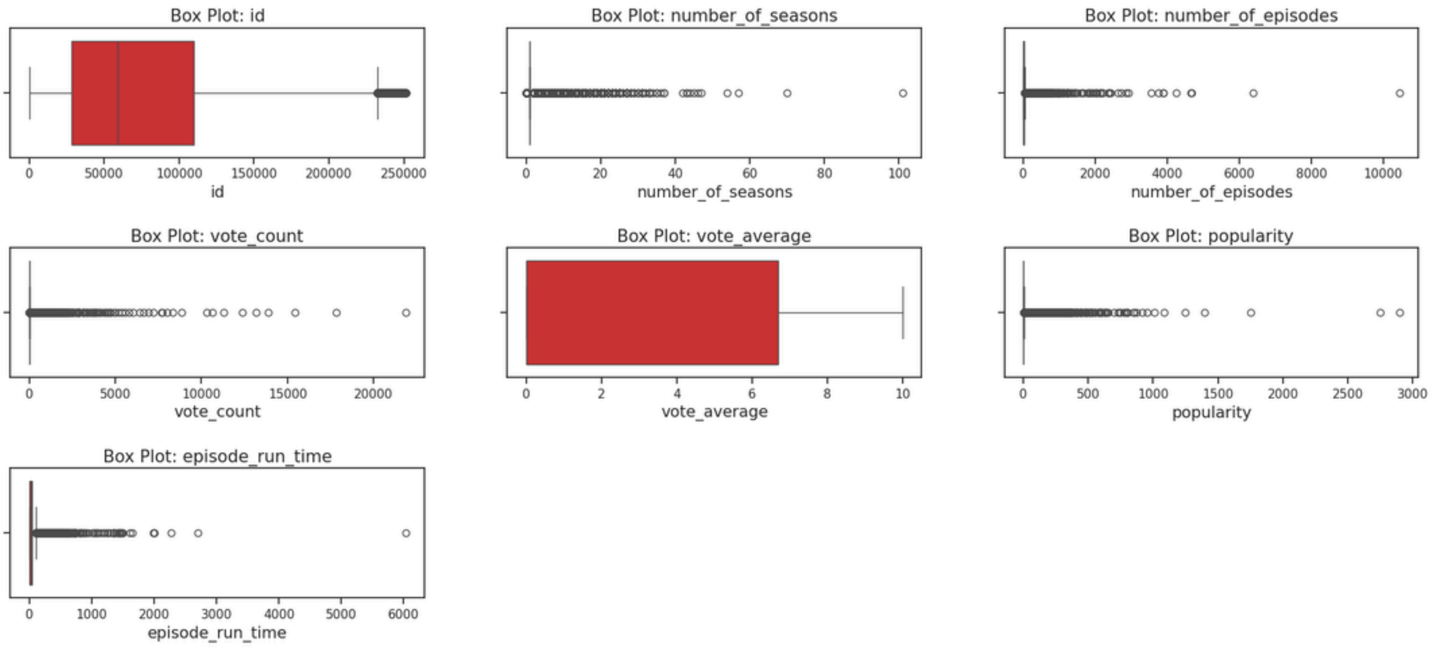




Identifying Outliers using Box Plots

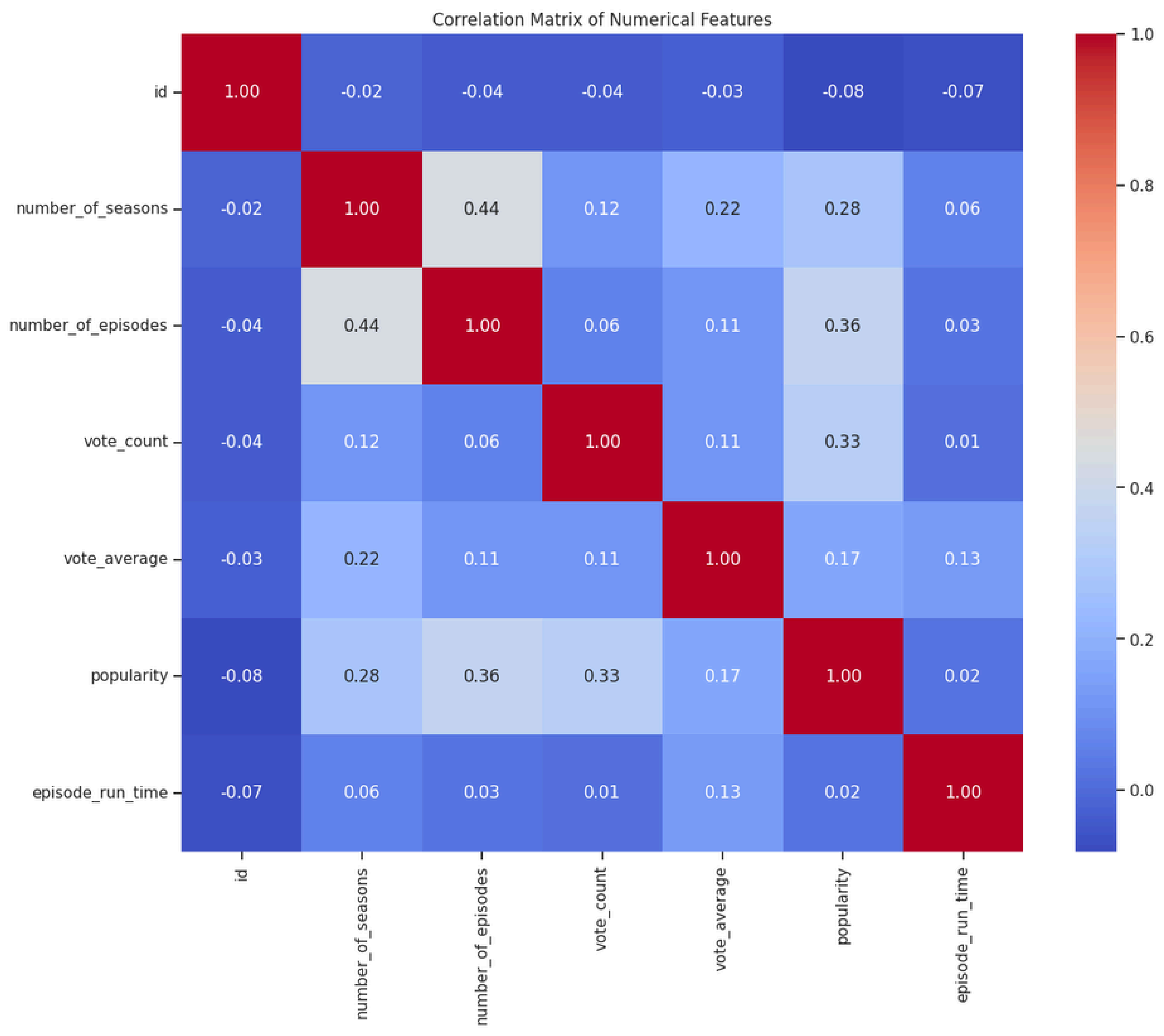
Outliers are typically identified using the following rule:

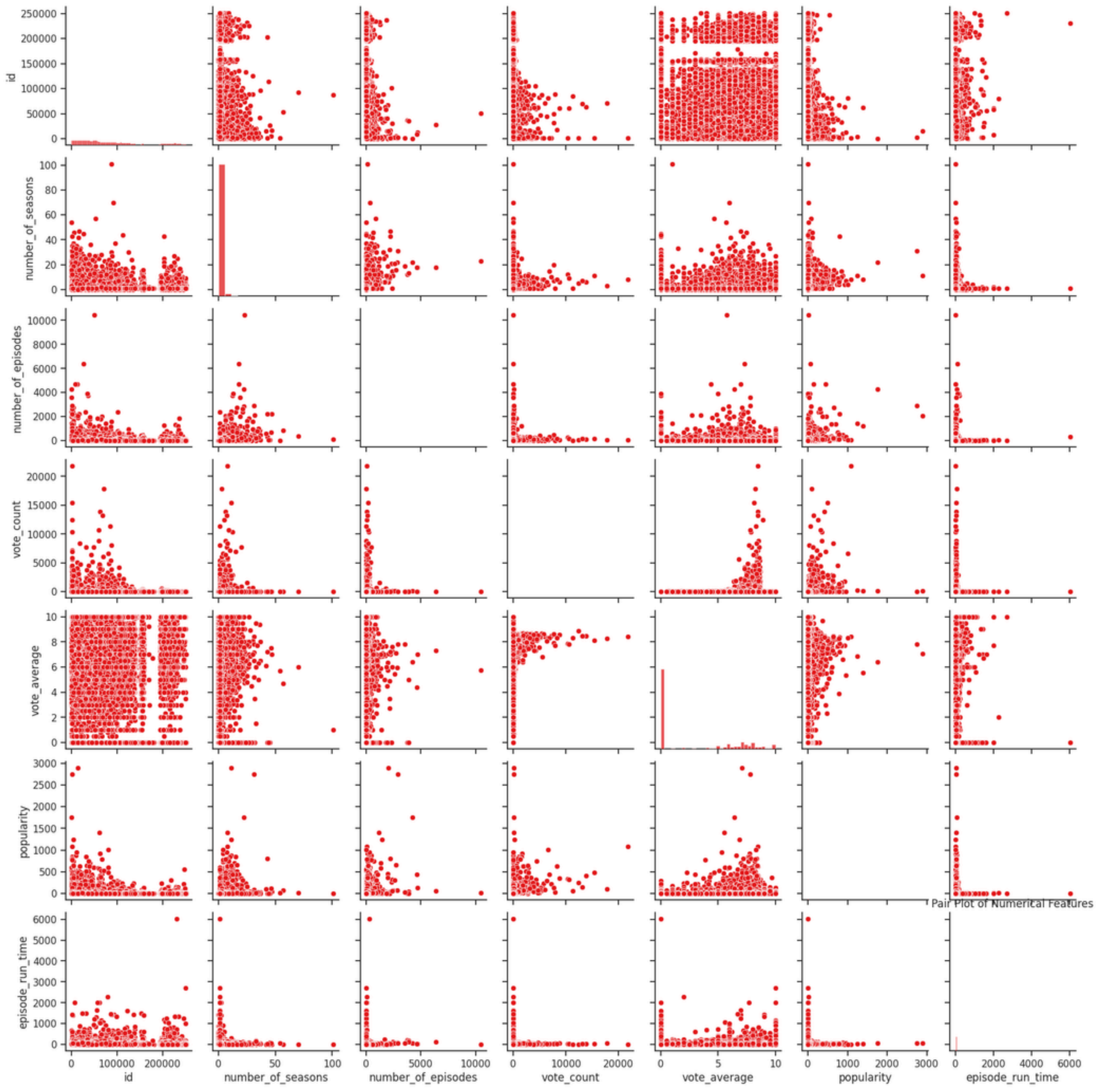
1. **Calculate the IQR:** The IQR is the difference between the third quartile (Q3) and the first quartile (Q1).
2. **Determine the upper and lower bounds:**
   - Upper bound: Q3 + 1.5 * IQR
   - Lower bound: Q1 - 1.5 * IQR
3. **Identify outliers:** Any data point that falls above the upper bound or below the lower bound is considered a potential outlier.



Correlation and other relationship Analysis:

Features to identify multicollinearity and feature importance.





Label Encodindg

Label Encoding is a technique used in machine learning to convert categorical data (data that represents categories or groups) into numerical data. Many machine learning algorithms work best with numerical data, so this conversion is often necessary.

Unique Categories:

It identifies all the unique categories within a categorical feature (column) of your dataset.

Assigning Numerical Labels:

It assigns a unique integer (numerical label) to each category. These labels usually start from 0 and increment for each new category.

Replacing Categories with Labels:

It replaces the original category values in the dataset with their corresponding numerical labels.

name	id	number_of_seasons	number_of_episodes	vote_count	vote_average	first_air_date	last_air_date	popularity type			genres	created_by	languages	networks	origin_country	production_companies	episode_run_time	popularity_log	adult_True	air_time	
0	minutes	39280.0	1	0.0	0	0.0	NaT	NaT	0.470004	4	19	20	6	18	19	18	30	0.385265	0	NaN	
1	man	222838.6	1	5.0	3	7.7	2023-05-28	2023-06-25	1.723837	4	12	20	10	18	12	18	50	1.002042	0	28.0	
2	erne tur retur	222838.6	1	10.0	0	0.0	2014-05-15	2014-07-17	0.470004	4	19	20	13	18	5	18	0	0.385265	0	63.0	

3. Data Cleansing

Data cleansing, also known as data cleaning, is the process of identifying and correcting or removing errors, inconsistencies, and inaccuracies in a dataset. This is a crucial step in data analysis and machine learning as it ensures the quality and reliability of your data, leading to more accurate and meaningful results.

Once the data is collected, it's crucial to clean it. Data cleaning involves:

- **Handling Missing Values:** Decide whether to impute missing values or remove records with missing data, based on their impact on model accuracy.
- **Removing Duplicates:** Ensure that duplicate entries are eliminated to avoid bias, particularly in customer records.
- **Correcting Errors:** Fix any errors or inconsistencies in the data, such as typos or incorrect values in billing information.
- **Dropping Incomplete Records (Optional):** I considered dropping rows with a significant number of missing values, but I ultimately decided against it.
- **Identifying Missing Values:** You used tmdb_eda.isnull().sum() to check for missing values in each column of your DataFrame.
- **Imputation:** You filled in missing numerical values using the mean or median, depending on the skewness of the data. Categorical missing values were imputed using the mode (most frequent value).
- **Detection:** You used Z-score and IQR methods to identify outliers in numerical features like 'popularity', 'number_of_episodes', 'number_of_seasons', 'vote_count', and 'episode_run_time'.
- **Treatment:** I employed techniques like capping outliers at certain percentiles and using transformations (e.g., log transformation for 'popularity') to reduce the impact of extreme values.

Removing Outliers using Z-score & IQR methods

Outliers are data points that are significantly different from other data points in a dataset. They can have a negative impact on the accuracy of machine learning models. There are several methods for removing outliers, but two of the most common are the Z-score method and the IQR method.

Z-score Method

The Z-score method involves calculating the Z-score for each data point in a dataset. The Z-score is a measure of how many standard deviations a data point is from the mean of the dataset. Data points with a Z-score greater than 3 or less than -3 are typically considered to be outliers.

The Z-score method is based on the assumption that the data is normally distributed. If the data is not normally distributed, the Z-score method may not be effective.

The Z-score method is sensitive to the size of the dataset. For small datasets, the Z-score method may identify too many data points as outliers.

IQR Method

The IQR method involves calculating the interquartile range (IQR) for a dataset. The IQR is the difference between the 75th percentile and the 25th percentile of the dataset. Data points that are more than 1.5 times the IQR below the 25th percentile or more than 1.5 times the IQR above the 75th percentile are typically considered to be outliers.

The IQR method is more robust to outliers than the Z-score method.The IQR method does not assume that the data is normally distributed. involves calculating the interquartile range (IQR) for a dataset. The IQR is the difference between the 75th percentile and the 25th percentile of the dataset. Data points that are more than 1.5 times the IQR below the 25th percentile or more than 1.5 times the IQR above the 75th percentile are typically considered to be outliers.

TMDB Target Values

project purpose:

building a predictive model to determine the success of a TV show. To do this, I need to define a target variable – what you're tryin to predict.

Pridicting the Popularity:

This is a numerical value provided by TMDB, reflecting the overall popularity of a TV show. It's a complex metric calculated by TMDB based on various factors like page views, user ratings, and interactions. Pros: It represents a holistic view of success, incorporating different aspects. Cons: It might be difficult to interpret directly and can be influenced by external factors.

Target Value prediction

Before diving into data preparation, it's essential to have a clear understanding of the problem the model aims to solve. This helps in identifying the relevant data features that will be crucial for the model's predictive capabilities.My project focuses on predicting customer churn in a telecommunications company, making it vital to select features such as customer usage patterns and contract details.Based on the potential insights and business value, I would suggest focusing on predicting the popularity value.

It is a complex and dynamic metric that reflects overall success. Vote Average (Rating): It captures audience satisfaction and critical acclaim. Both of these targets have valuable real-world implications and can be approached with a variety of machine learning models.

Data Type Conversion:

Boolean to Integer: You converted boolean columns to integers (0 and 1) for consistency.
Date/Time Conversion: You converted 'first_air_date' and 'last_air_date' to datetime objects, calculated 'air_time' (duration in days), and added it as a new column.

-
-

Handling Inconsistent Data:

- No specific code for handling inconsistent data, like duplicate records or data entry errors. However, this can also be part of data cleansing.

Creating Dummies variables:

Dummy variables are a way to represent categorical data (like TV show genres, languages, or status) as numerical values that machine learning models can understand. They are essentially binary (0 or 1) variables that indicate the presence or absence of a particular category.

Most machine learning algorithms work with numerical data. They can't directly handle categorical data in its original form (e.g., text labels). Dummy variables convert categorical data into a numerical format that these algorithms can process.

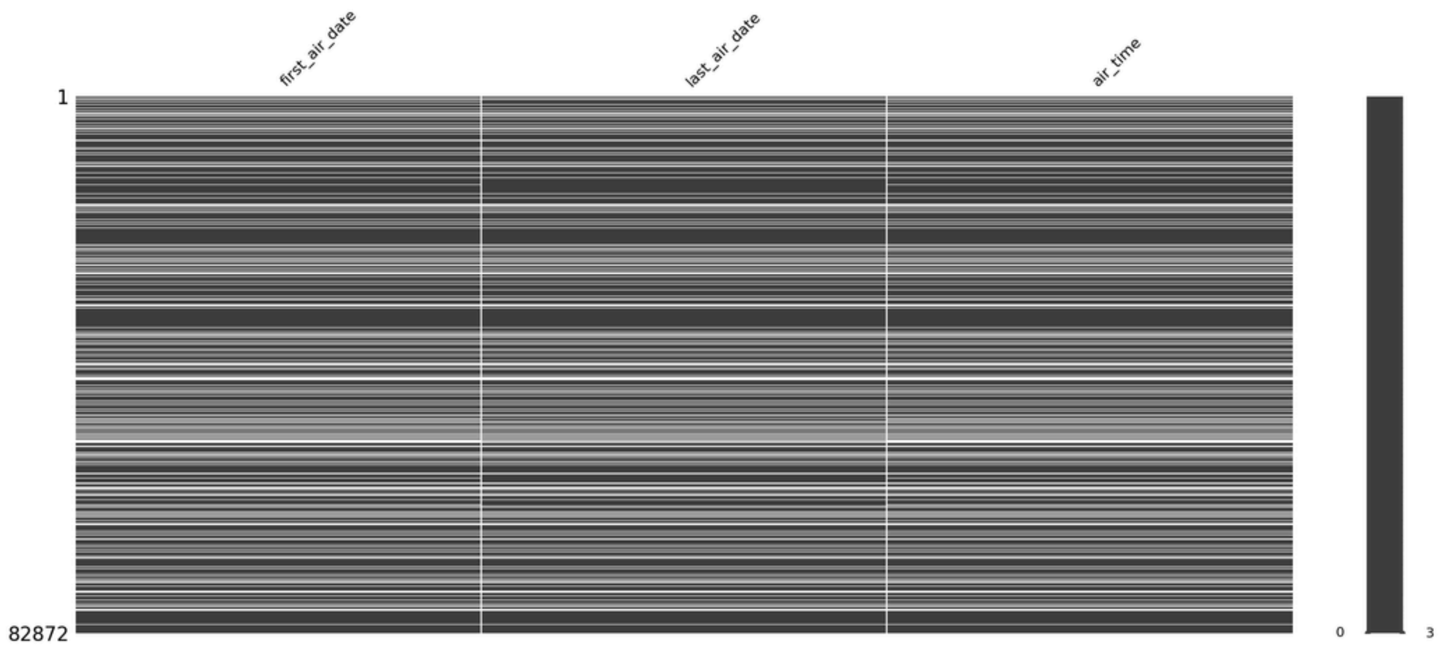
- Creating dummies for the ‘adult’ categorial column.
- Converting 'first_air_date' & 'last_air_date' to datetime objects & adding Column 'air_time'

Data Type: Converting to datetime objects allows you to perform date and time-based calculations and analysis (like finding the difference between dates).

Feature Engineering: The 'air_time' column you create is a new feature that might be useful for your predictive model. It represents the total duration a TV show has been on air, which could be related to its popularity or success.

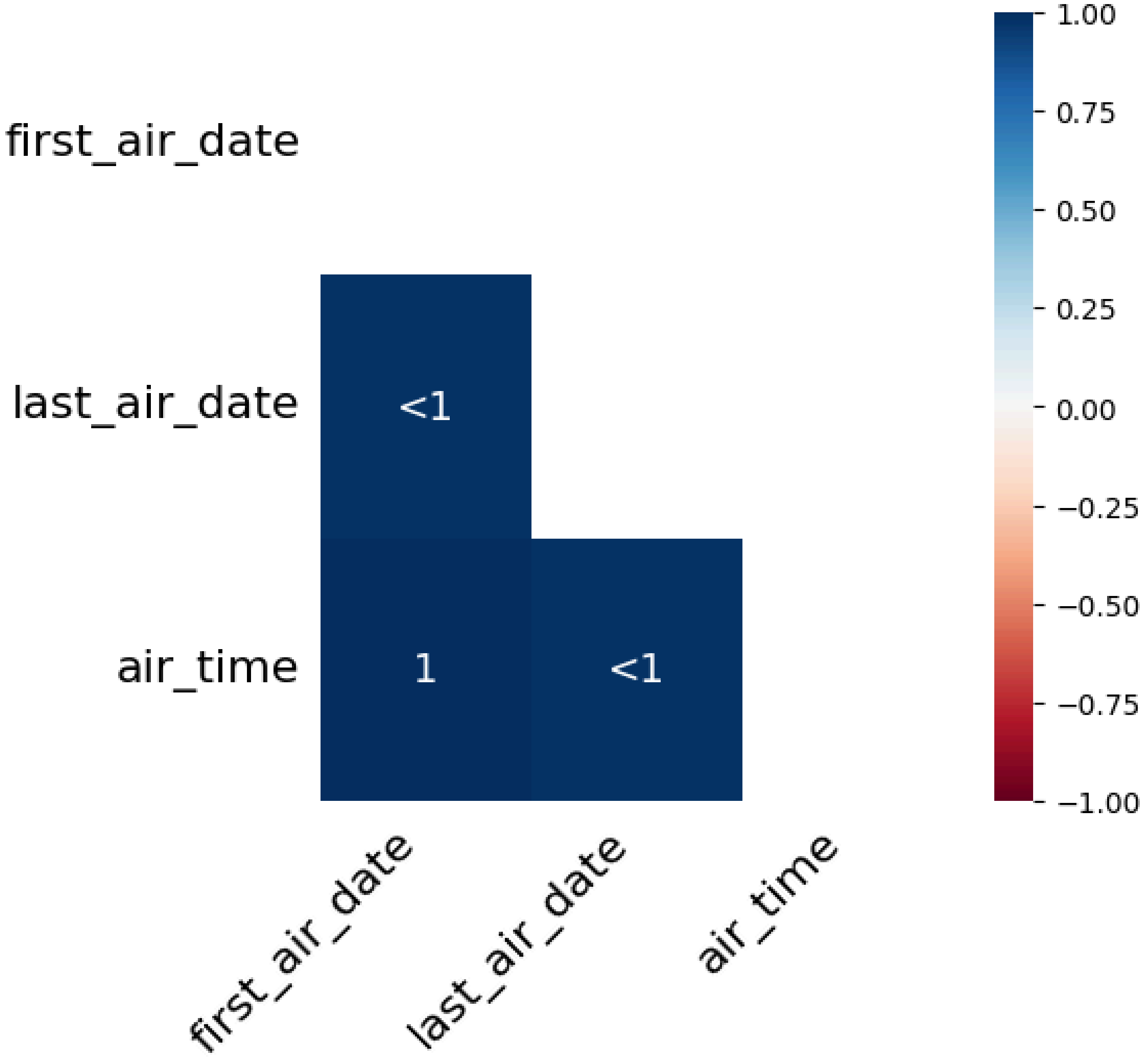
Plotting the missingness (nullity) matrix

A missingness matrix is a visual representation of missing values in a dataset. It helps you quickly identify patterns of missing data and understand the extent of missingness in your dataset.



Missingness Correlation HeatMap

A missingness correlation heatmap is a visualization that helps you understand the relationships between missing values in different columns of your dataset. It shows the correlation between the presence or absence of data in one column and the presence or absence of data in another column.



Missing Values Imputation:

Missing values imputation is the process of filling in missing data points in your dataset with estimated values. It's an important step in data preprocessing because many machine learning algorithms cannot handle missing values directly.

Missing Values & Imputation Methods using MICE and KNN

Missing values imputation is crucial for data analysis and machine learning. Two effective methods are Multiple Imputation by Chained Equations (MICE) and K-Nearest Neighbors (KNN).

MICE Imputation

MICE is a statistical method that fills in missing data by iteratively predicting missing values based on other variables in the dataset. It's particularly useful for handling missing values in complex datasets with multiple variables and complex relationships.

Handles various data types: MICE can impute missing values for both numerical and categorical variables.

Preserves data distribution: By iteratively predicting missing values, MICE aims to maintain the original data distribution.

Flexible and robust: It can handle different missing data patterns and complex relationships between variables.

Alternative Approach (KNN Imputation):

While MICE is the preferred approach, you could consider KNN imputation as an alternative, especially for numerical features with relatively few missing values. However, remember to scale your numerical features before applying KNN imputation

Data cleansing summery:

By performing these data cleansing steps, I improved the quality and consistency of the TV show data, which will be essential for building accurate and reliable predictive models in the subsequent stages of my project.

4.Feature Eng. & Feature Selection

Data transformation involves converting data into a suitable format or structure for analysis. This can include:

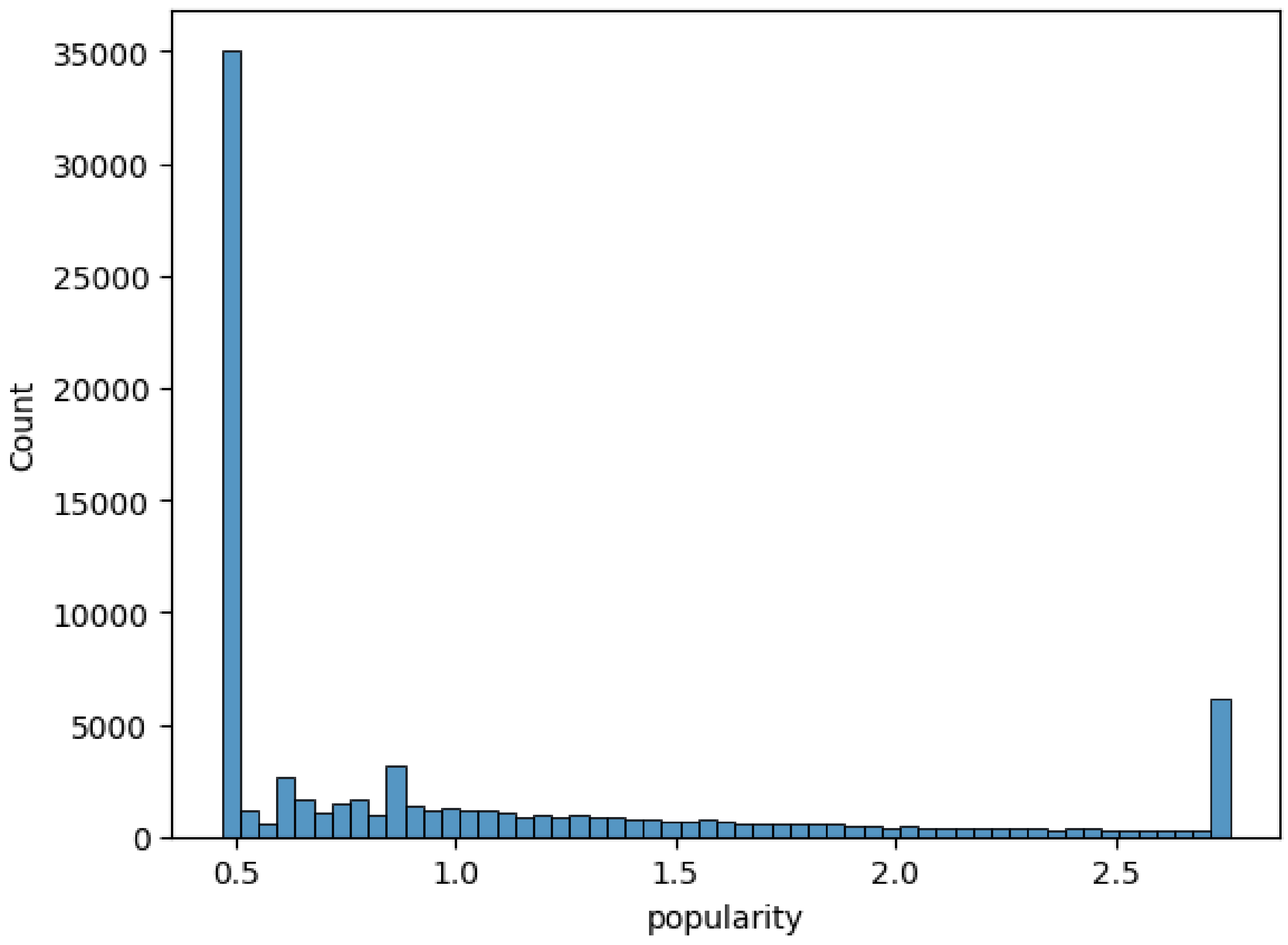
• **Enriching:** Create new features from existing ones to better capture the underlying patterns in the data. This can involve mathematical transformations, aggregations, or domain-specific knowledge.

• **Normalization/Standardization:** Scale numerical features to ensure they have similar ranges, which helps certain algorithms perform better.

• **Feature Selection** : Choosing the most valuable features that predict the target value by running penalty models

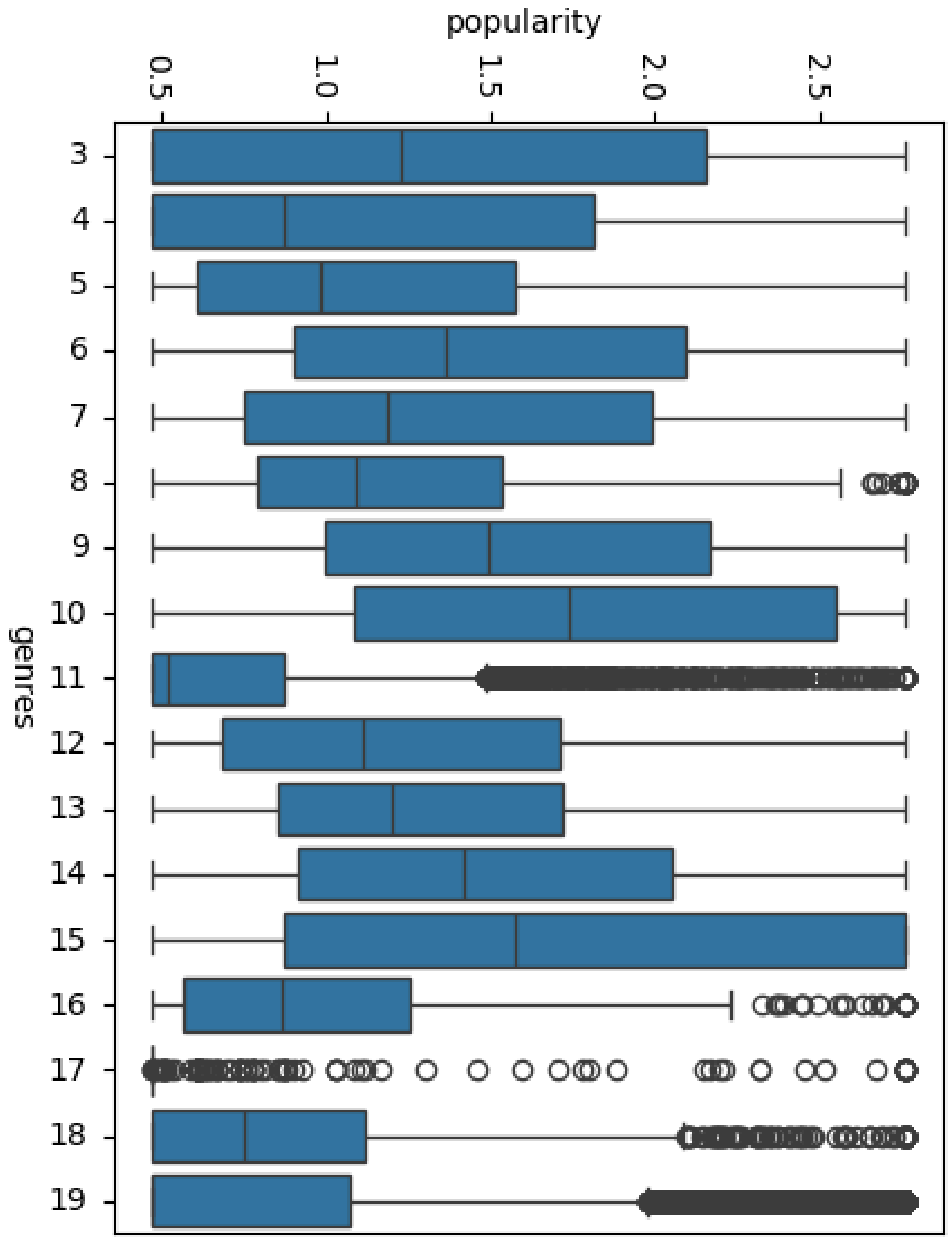
Univariate Analysis

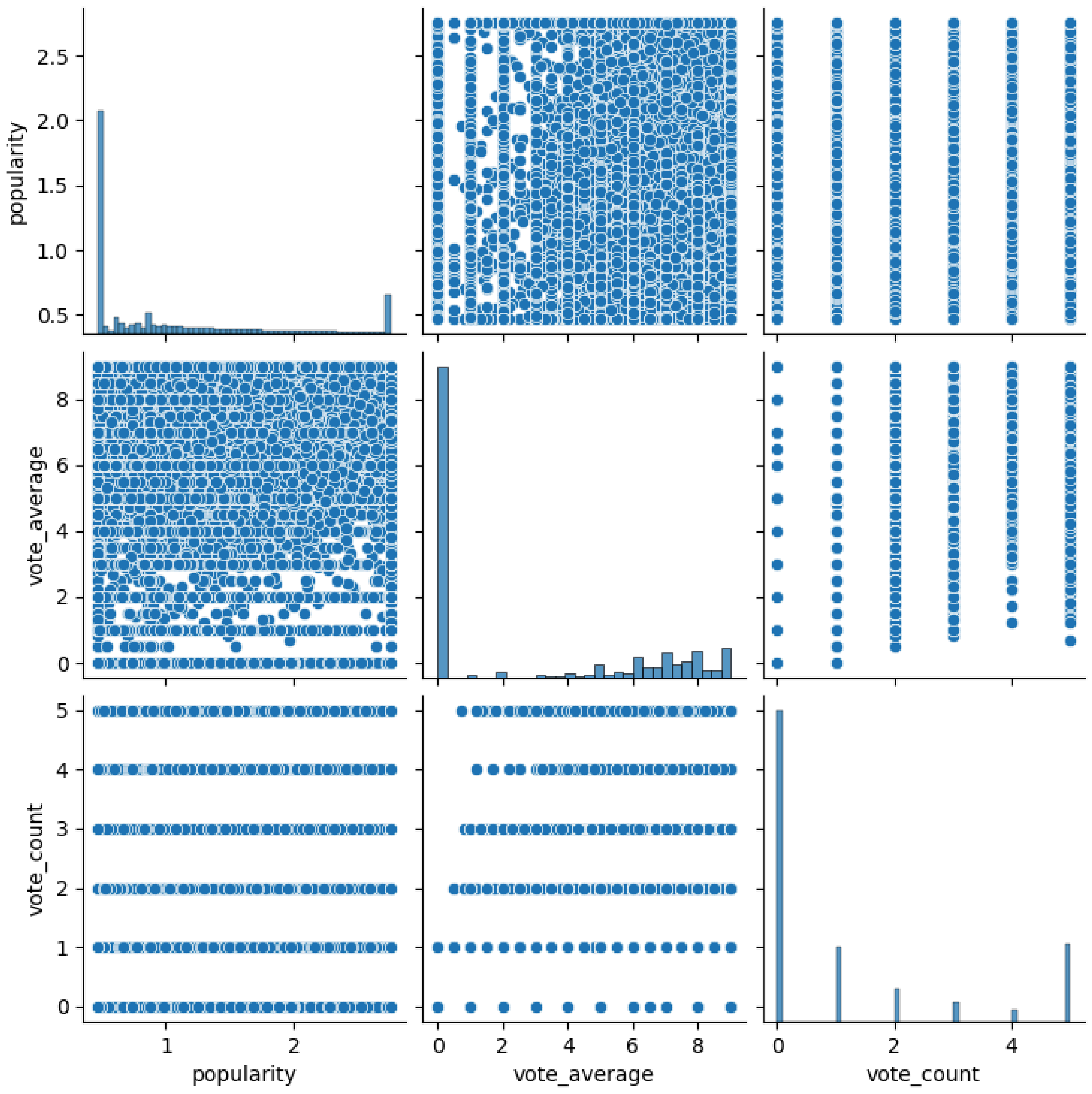
Univariate analysis is the simplest form of analyzing data where you focus on understanding the distribution and patterns of a single variable (feature) at a time. It's a fundamental step in exploratory data analysis (EDA) and helps you gain insights into the characteristics of your data before moving on to more complex analyses.



Categorical vs. Numerical:

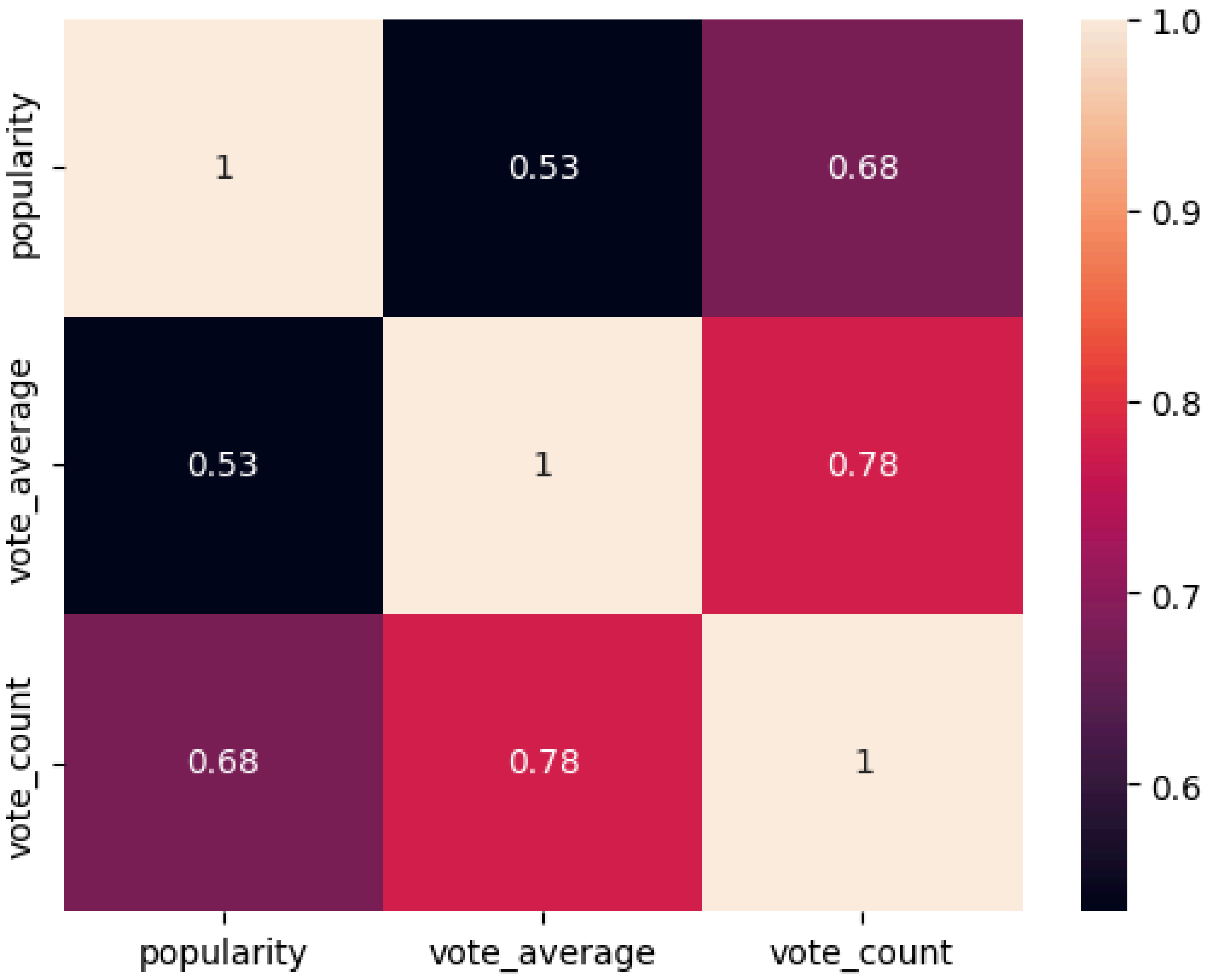
Analyze the relationship between categorical and numerical features using box plots or violin plots.


popularity
0.5
1.0
1.5
2.0
2.5
genres
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19



Heatmaps:

Visualize correlation matrices using heatmaps for a clearer view of feature relationships.



Multivariate Analysis - Pair Plots:

reate pair plots to visualize relationships between multiple numerical features simultaneously.

Enriching:

Creating new features from existing ones to better capture the underlying patterns in the data. This can involve mathematical transformations, aggregations, or domain-specific knowledge.

name	id	number_of_seasons	number_of_episodes	vote_count	vote_average	first_air_date	last_air_date	popularity	type	...	avg_episode_runtime_per_number_of_seasons	total_runtime	production_company_count	network_count	genre_count	origin_country_count	created_by_count	vote_count_to_episodes_ratio	weighted_vote_average	popularity_score	
0	minutes	39280.0	1	0.0	0	0.0	NaT	NaT	0.470004	4	...	30.0	0.0	1	1	1	1	1	NaN	0.0	0.235002
1	man	222838.6	1	5.0	3	7.7	2023-05-28	2023-06-25	1.723837	4	...	50.0	250.0	1	1	1	1	1	0.600000	23.1	3.771919
2	erne tur retur	222838.6	1	10.0	0	0.0	2014-05-15	2014-07-17	0.470004	4	...	0.0	0.0	1	1	1	1	1	0.000000	0.0	0.235002
3		34835.0	1	12.0	5	6.4	2006-10-06	2006-12-22	2.375928	4	...	30.0	360.0	1	1	1	1	1	0.416667	32.0	4.107964
4	bama	42598.0	1	0.0	0	0.0	NaT	NaT	0.470004	4	...	0.0	0.0	1	1	1	1	1	NaN	0.0	0.235002

5 rows × 31 columns

Normalization/Standardization:

Bringing Features to a Common Scale -In machine learning, many algorithms perform better when numerical features have similar ranges. This is because features with larger ranges can disproportionately influence the model, leading to biased results. Normalization and standardization are two common techniques used to address this issue.

Benefits:

1. **Algorithm Performance:** Many machine learning algorithms, such as distance-based methods (k-Nearest Neighbors, K-Means), gradient descent-based algorithms (Linear Regression, Logistic Regression), and Support Vector Machines, are sensitive to the scale of features. When features have vastly different ranges, those with larger values can dominate the calculations, leading to suboptimal performance.
2. **Equal Feature Contribution:** Scaling ensures that all features contribute equally to the model's learning process. Without scaling, features with larger ranges might be given more importance than those with smaller ranges, even if they are not inherently more informative.
3. **Faster Convergence:** In optimization algorithms like gradient descent, scaling can help the algorithm converge faster to the optimal solution. This is because the gradients are less likely to oscillate wildly when features are on a similar scale.

The Scale Numerical Features

Using StandardScaler - A Standardization Technique

StandardScaler is a preprocessing technique provided by the scikit-learn library in Python. It's used to standardize features by removing the mean and scaling to unit variance. In simpler terms, it transforms the data such that it has a mean of 0 and a standard deviation of 1.

The Method for StandardScaler

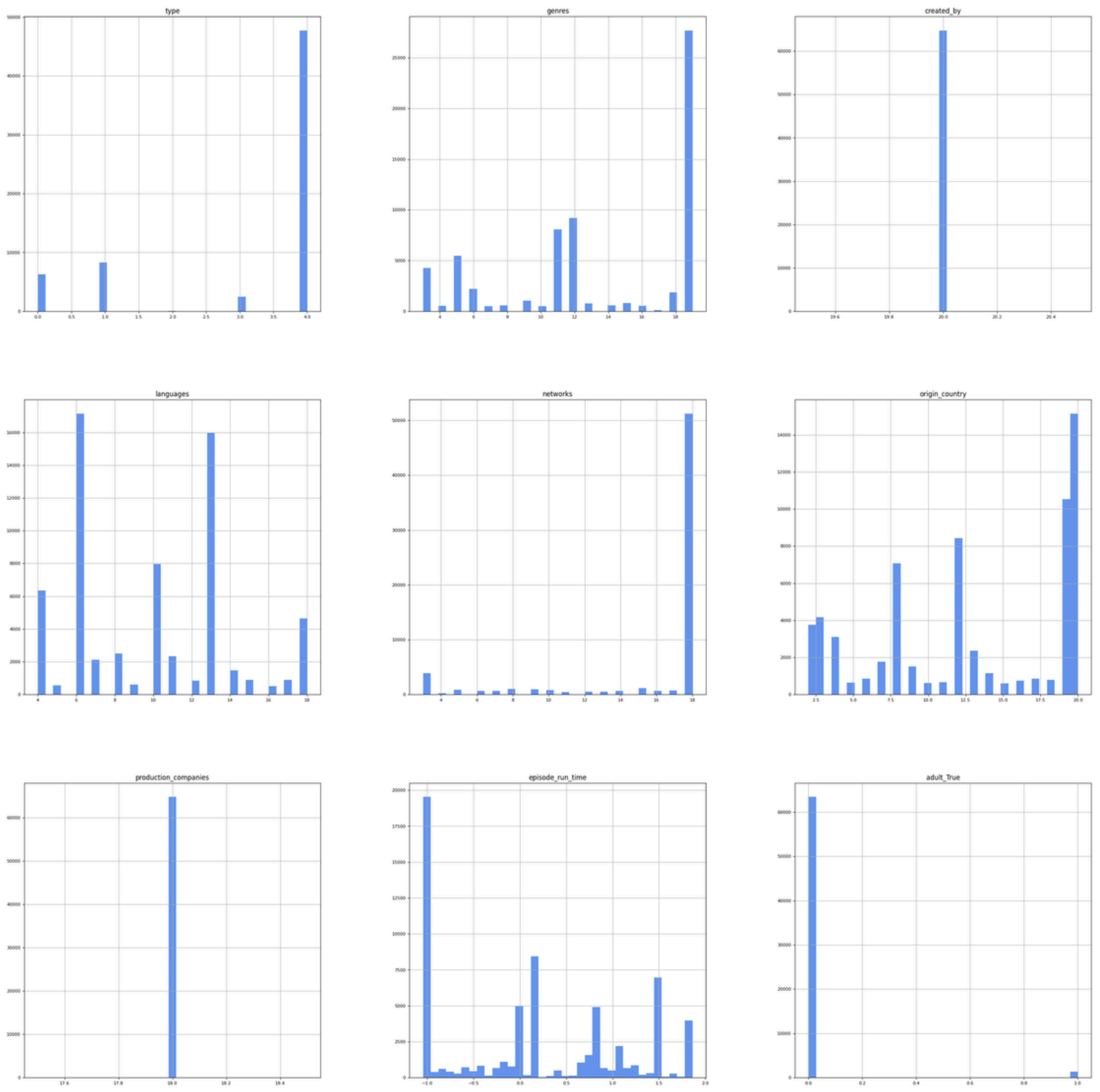
1. **Calculating Mean and Standard Deviation:** StandardScaler first calculates the mean (μ) and standard deviation (σ) of each feature in your dataset.
2. **Centering the Data:** It then centers the data by subtracting the mean from each value: X_centered = X - μ. This shifts the data so that the mean becomes 0.
3. **Scaling to Unit Variance:** Next, it scales the data by dividing each centered value by the standard deviation: X_scaled = X_centered / σ. This ensures that the standard deviation becomes 1.

Convert categorical variables:

Convert categorical variables into a format that can be provided to ML algorithms to do a better job in prediction. This typically involves creating binary columns for each category in a categorical feature.

Numeric (Continuous) Analysis:

Numeric (continuous) analysis involves examining numerical features in your dataset to understand their distributions, relationships, and potential impact on your machine learning models. It's a crucial step in the data exploration and preprocessing phase of a data science project.



Skeuness

Skewness is a statistical measure that describes the asymmetry of the probability distribution of a real-valued random variable about its mean. In simpler terms, it tells you how much a distribution leans to one side or the other.

	skewness
air_time	7.348937
adult_True	6.739496
total_runtime	1.877938
weighted_vote_average	1.067884
number_of_episodes	0.948948
vote_count	0.931158
popularity	0.890423
id	0.694727
log_popularity	0.657328
popularity_log	0.651853
languages	0.324002
popularity_score	0.269934
vote_average	0.190634
avg_episode_runtime_per_number_of_seasons	0.172641
episode_run_time	0.172641

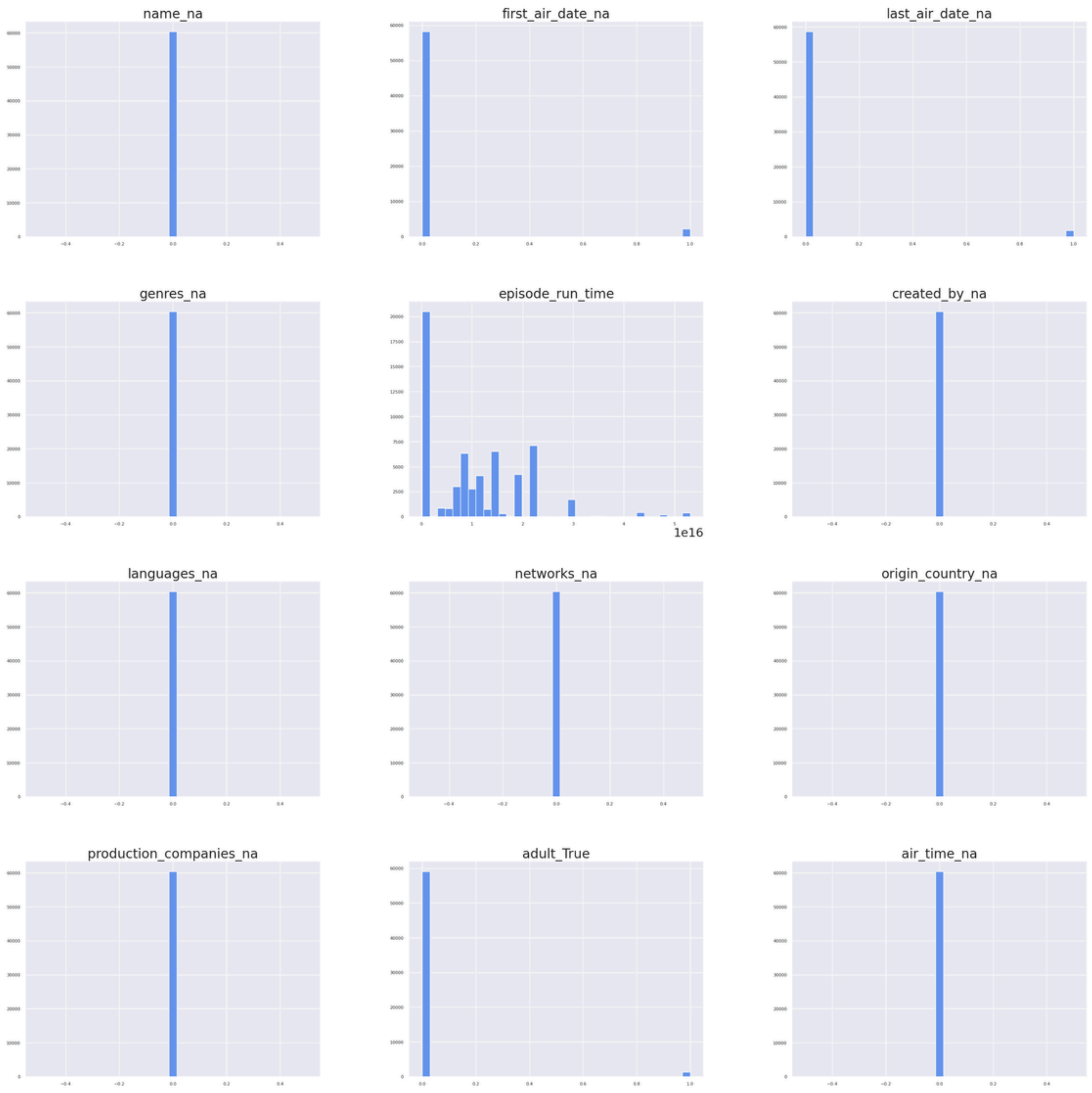
Convert specified columns to float:

In data analysis and machine learning, it's often necessary to ensure that numerical columns are represented as float data types. This is because many algorithms and operations expect or perform better with float values. Here are some reasons for conversion:

Mathematical Operations: Performing calculations like addition, subtraction, multiplication, and division on numerical data often requires float values for accuracy.

Algorithm Compatibility: Many machine learning algorithms expect features to be represented as floats for proper functionality and performance.

Data Consistency: Maintaining consistency in data types can simplify data manipulation and analysis.



Correlation: Understanding Relationships Between Variables

Correlation is a statistical measure that describes the strength and direction of the linear relationship between two or more variables. In simpler terms, it tells you how much two variables tend to change together.

Feature Selection: Choosing the Right Ingredients for the Model

Feature selection is the process of selecting a subset of relevant features (variables, predictors) to use in building a machine learning model. It's a crucial step in the model development process because it can:

Improve Model Performance: By removing irrelevant or redundant features, you can reduce noise and overfitting, leading to better generalization and accuracy.

Reduce Computational Cost: Fewer features mean faster training and prediction times, especially for complex models and large datasets.

Enhance Model Interpretability: Simpler models with fewer features are easier to understand and explain, making them more trustworthy and actionable.

Methods for Feature Selection:

There are three main categories of feature selection methods:

1. **Filter Methods:** These methods evaluate the relevance of features based on their statistical properties, independent of the chosen machine learning algorithm.
   - **Examples:**
     - **Correlation:** Selecting features highly correlated with the target variable.
     - **Chi-squared test:** Evaluating the dependence between categorical features and the target variable.
     - **ANOVA (Analysis of Variance):** Comparing the means of the target variable across different categories of a feature.
2. **Wrapper Methods:** These methods use a specific machine learning algorithm to evaluate the performance of different feature subsets. They are computationally more expensive but often lead to better results.
   - **Examples:**
     - **Recursive Feature Elimination (RFE):** Iteratively removing the least important features based on the model's coefficients or feature importance scores.
     - **Sequential Feature Selection (SFS):** Starting with an empty set and adding features one by one based on their contribution to model performance.
3. **Embedded Methods:** These methods incorporate feature selection as part of the model training process. They are often more efficient than wrapper methods and can identify interactions between features.
   - **Examples:**
     - **LASSO (Least Absolute Shrinkage and Selection Operator):** A linear regression model that uses regularization to shrink some coefficients to zero, effectively performing feature selection.

- **Decision Tree-based algorithms:** These algorithms inherently select features based on their importance in splitting the data.

Featuer Selection :Creating DataFrame with most valuable variables

Selected variables - recommended by 3 or more models :

Found 13 columns for the prediction 'popularity' column:

Index: 78501 entries, 0 to 82871

Data columns (total 13 columns):

```
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   id                          78501 non-null  float64
 1   number_of_episodes          78501 non-null  float64
 2   vote_average                78501 non-null  float64
 3   first_air_date              58633 non-null  datetime64[ns]
 4   last_air_date               59105 non-null  datetime64[ns]
 5   episode_run_time            78501 non-null  int64
 6   air_time                    78501 non-null  float64
 7   log_popularity              78501 non-null  float64
 8   total_runtime               78501 non-null  float64
 9   vote_count_to_episodes_ratio  60393 non-null  float64
 10  weighted_vote_average       78501 non-null  float64
 11  popularity_score            78501 non-null  float64
 12  popularity                  78501 non-null  float64
dtypes: datetime64[ns](2), float64(10), int64(1)
memory usage: 8.4 MB
```

Multivariable Analysis -Summarization and Selection of Variables

The Lasso penalty here is controlled by alpha = 0.01. This penalty forces some coefficients to shrink to zero, effectively performing feature selection. A higher penalty (larger alpha) would result in more coefficients being zeroed out, and a lower penalty (smaller alpha) would retain more features.

Corroletion TMDB Dataset

Correlation measures the strength and direction of a linear relationship between two variables.

In the TMDB dataset, this could be the relationship between features like:

vote_average and popularity:

Do movies with higher average ratings tend to be more popular?

number_of_episodes and total_runtime:

Does the number of episodes in a show correlate with its total runtime?

vote_count and popularity_score:

Is there a connection between the number of votes a show receives and its calculated popularity score?

Spearman's Rank Correlation

The method='spearman' argument specifies that Spearman's rank correlation is used. This method is non-parametric, meaning it doesn't assume a normal distribution of the data. It assesses the monotonic relationship between variables – whether they tend to increase or decrease together, regardless of the specific shape of the relationship.

Interpreting Correlation Values

The correlation coefficient ranges from -1 to +1:

+1: Perfect positive correlation. As one variable increases, the other increases proportionally.
-1: Perfect negative correlation. As one variable increases, the other decreases proportionally.
0: No correlation. There is no linear relationship between the variables. Values between -1 and +1 indicate varying degrees of correlation:

Strong correlation: Values closer to -1 or +1 (e.g., -0.8, +0.7) Moderate correlation: Values around -0.5 or +0.5 Weak correlation: Values closer to 0 (e.g., -0.2, +0.3)

Example Interpretation

If the correlation between vote_average and popularity is +0.7, it suggests a strong positive correlation. This means that TV shows with higher average ratings tend to be more popular.

Important Considerations

Correlation does not imply causation:

Even if two variables are strongly correlated, it doesn't mean that one causes the other. There could be other factors influencing both variables.

Context is crucial:

The interpretation of correlation should be based on the specific dataset and domain knowledge. A correlation of 0.5 might be considered strong in some contexts but weak in others. How to use the correlation results:

Feature Selection:

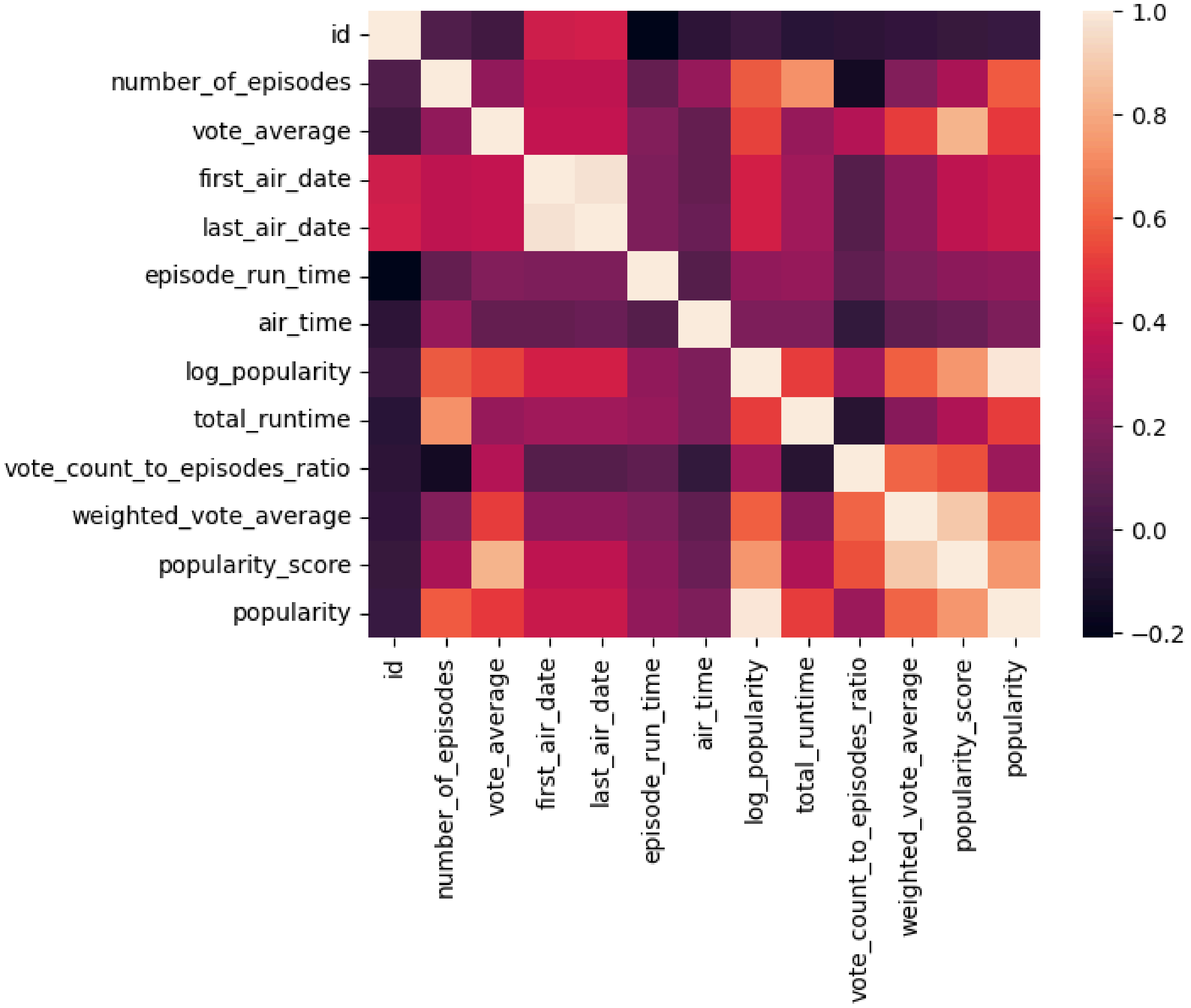
Identify features that are strongly correlated with your target variable (e.g., popularity). These features are likely to be important predictors in your machine learning model. Feature Engineering:

Consider creating new features by combining correlated features. For example, you could create a feature that combines 'vote_average' and 'vote_count' to capture a more comprehensive measure of audience approval.

Multicollinearity:

Very strong correlations between predictor features (features you are using to predict your target), you might want to remove one of the highly correlated features to avoid multicollinearity issues in your model. Multicollinearity can make it difficult to interpret the individual effects of features and can lead to unstable model estimates.

Heat Map



Chi-Square testing between episode_run_time & popularity

The Chi-Square test compares the observed frequencies of different combinations of categories in your data to the expected frequencies if there were no relationship between the variables. If the observed frequencies deviate significantly from the expected frequencies, it suggests that there is an association between the variables.

Chi-Square Test Results:

Chi-Square Statistic: 871193.5373

P-Value: 0.0000

Degrees of Freedom: 722190

5.Imbalance Techniques & Model Selection & Fine Tuning & Model

Dataset Imbalansing Techniques

The code utilizes several resampling techniques to address class imbalance in the dataset. These techniques aim to balance the distribution of the target variable ('success' in this case) by either oversampling the minority class, undersampling the majority class, or a combination of both.

RandomOverSampler (ROS): This technique randomly duplicates samples from the minority class to increase its representation in the dataset. It's a simple and straightforward approach to address imbalance.

RandomUnderSampler (RUS): This technique randomly removes samples from the majority class to reduce its dominance in the dataset. This helps balance the class distribution but can lead to loss of information.

SMOTE (Synthetic Minority Over-sampling Technique): This technique generates synthetic samples for the minority class by interpolating between existing minority class samples. It creates new data points that are similar to the existing ones, helping to diversify the minority class representation.

SMOTETomek: This is a hybrid technique that combines SMOTE for oversampling the minority class and Tomek Links for undersampling the majority class. Tomek Links identify pairs of data points from different classes that are close to each other. By removing the majority class instancResults for ROS:

```
Accuracy: 0.9989

Classification Report:

           precision    recall  f1-score   support



        0      1.00      1.00      1.00      7961

        1      1.00      1.00      1.00      7981



   accuracy                          1.00     15942

  macro avg      1.00      1.00      1.00     15942

weighted avg      1.00      1.00      1.00     15942




Results for RUS:
```

```
Accuracy: 0.7117

Classification Report:

          precision    recall  f1-score   support



        0       0.63      1.00      0.78      7961

        1       1.00      0.42      0.60      7981



   accuracy                         0.71     15942

  macro avg       0.82      0.71      0.69     15942

weighted avg       0.82      0.71      0.69     15942




/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names

  warnings.warn(

Results for SMOTE:

Accuracy: 0.6625

Classification Report:

          precision    recall  f1-score   support



        0       0.60      1.00      0.75      7961

        1       1.00      0.33      0.49      7981



   accuracy                         0.66     15942

  macro avg       0.80      0.66      0.62     15942

weighted avg       0.80      0.66      0.62     15942
```

```
Results for SMOTETomek:

Accuracy: 0.6484

Classification Report:

           precision    recall  f1-score   support



         0       0.59      1.00      0.74      7961

         1       1.00      0.30      0.46      7981



    accuracy                           0.65     15942

   macro avg       0.79      0.65      0.60     15942

weighted avg       0.79      0.65      0.60     15942
```

SMOTETomek aims to create a clearer separation between the classes.

Data Splitting Schem :

```
+-------------------------------------+

| Original Dataset (X, y)             |

+-------------------------------------+

      /       \

     /         \

    /           \

+-----------------+  +-----------------+

| Training Set    |  | Temporary Set   |

| (70%)           |  | (30%)           |

+-----------------+  +-----------------+

                 /       \

                /         \

               /           \
+-----------------+  +-----------------+
| Validation Set  |  | Test Set        |
| (15%)           |  | (15%)           |
+-----------------+  +-----------------+
```

Classification Report- Model HistGradientBoostingClassifier

The HistGradientBoostingClassifier is a powerful machine learning algorithm for classification tasks, introduced in scikit-learn. It's a variant of Gradient Boosting that uses histograms to bin continuous features, leading to significant speed improvements, especially for large datasets.

Accuracy: 0.9990

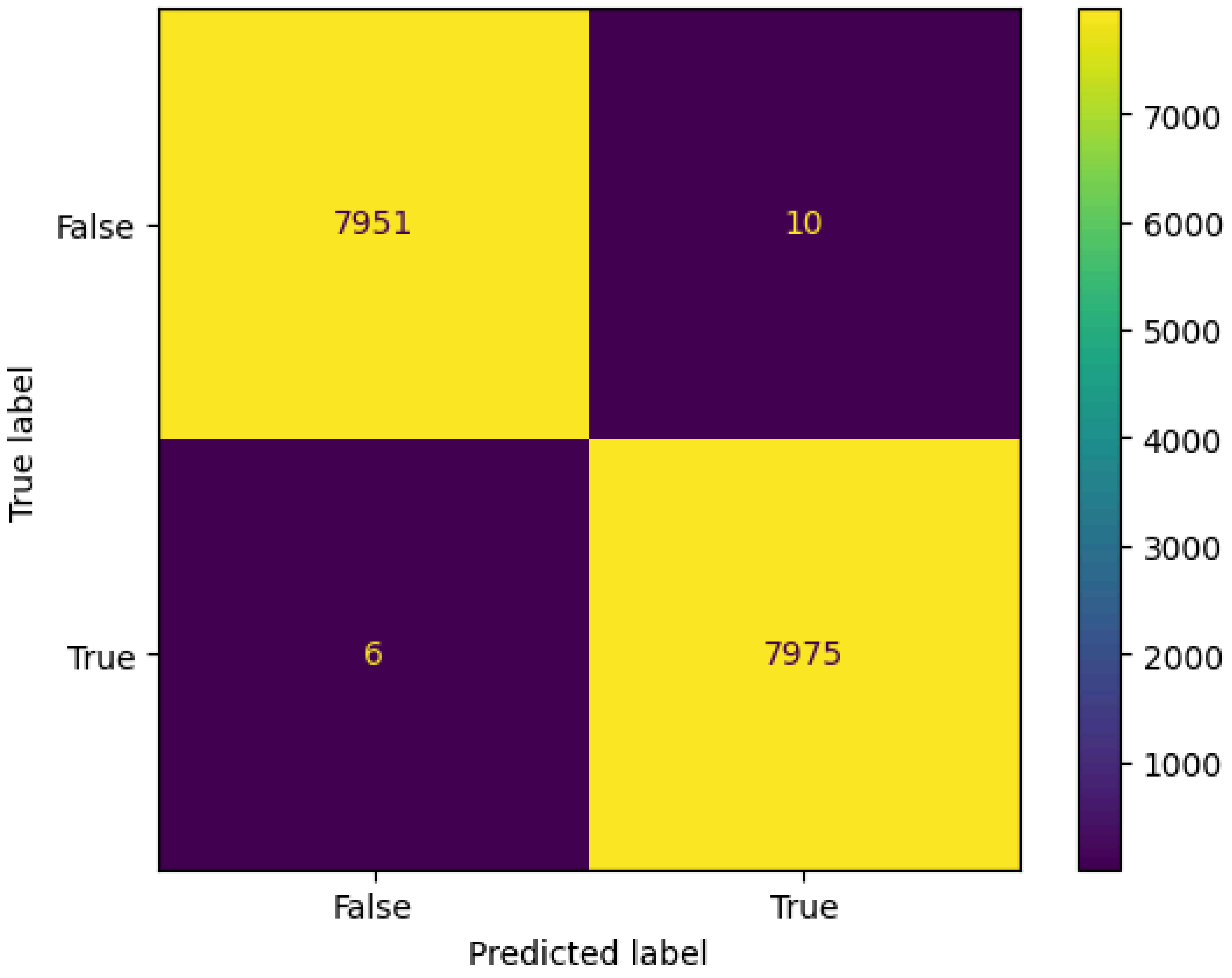
Classification Report:

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      7961
           1       1.00      1.00      1.00      7981

    accuracy                           1.00     15942
   macro avg       1.00      1.00      1.00     15942
weighted avg       1.00      1.00      1.00     15942
```

Confusion Metrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. It allows us to visualize the performance of a supervised learning algorithm by showing the counts of:

True Positives (TP): Correctly predicted positive instances. True Negatives (TN): Correctly predicted negative instances. False Positives (FP): Incorrectly predicted positive instances (Type I error). False Negatives (FN): Incorrectly predicted negative instances (Type II error).



Gridsearch:

The machine learning find the optimal hyperparameters for a model. It helps improve model performance by systematically exploring different combinations of hyperparameter values and selecting the combination that yields the best results.

hyperparameter trade-off
Gread Search Cross-Validation
C - achieving a low training error and a low testing error.0.1 week regulation
gamma - controls the width of the radial basis function
RFB: Radial Basis Function (linear, sigmoid, poly)

RBF Kernel: The Core Idea

The RBF kernel is a popular choice for SVMs because it's flexible and can handle non-linear relationships in data. Here's the basic idea:

Mapping to Higher Dimensions:

The RBF kernel implicitly maps your data points into a higher-dimensional space. This is done without actually calculating the new coordinates in that space – it's a mathematical trick. Measuring Similarity:

In this higher-dimensional space, the RBF kernel measures the similarity between data points based on their distance from each other. Points closer together are considered more similar. Creating a Decision Boundary:

The SVM then uses this similarity information to construct a decision boundary (a hyperplane) that separates the classes as well as possible.

Grid Search Cross-Validation:

Improved Model Performance:

By systematically exploring different hyperparameter values, we find the optimal settings that lead to better model performance.

Reduced Overfitting:

Cross-validation helps prevent overfitting by evaluating the model on multiple subsets of the data.

Automation:

Grid search automates the process of hyperparameter tuning, saving you time and effort.

Result : Best parameters: {'classifier__C': 10, 'classifier__gamma': 1, 'classifier__kernel': 'linear'}

SVC Classifier Best Parameters

The best parameters found by GridSearchCV represent the combination of hyperparameter values that resulted in the highest performance (as measured by your chosen scoring metric) on the validation data during cross-validation.

serch matrix :

```
[[7955    6]

 [   4 7977]]
```

Model GridSearchCV results:

```
          precision    recall  f1-score   support



        0      1.00      1.00      1.00      7961

        1      1.00      1.00      1.00      7981
```

```
    accuracy                        1.00     15942
   macro avg       1.00      1.00      1.00     15942
weighted avg       1.00      1.00      1.00     15942
```

SVC - Support Vector Classifier

The Support Vector Classifier (SVC) is a supervised machine learning algorithm used for classification tasks. It aims to find an optimal hyperplane that best separates data points into different classes.

Accuracy: 0.9957

Classification Report:

```
              precision    recall  f1-score   support

           0       0.99      1.00      1.00      7961
           1       1.00      0.99      1.00      7981

    accuracy                           1.00     15942
   macro avg       1.00      1.00      1.00     15942
weighted avg       1.00      1.00      1.00     15942
```

SVM-Support Vector Machine

SVM is a supervised ML Algorithm - used for both classification and regresstion models.

Mean Absolute Error (MAE): 0.038

Mean Squared Error (MSE): 0.002

Root Mean Squared Error (RMSE): 0.046

R-squared (R^2): 0.998

Root Mean Squared Logarithmic Error (RMSLE): 0.119

Linear Regression- Using pipeline with imputation and scaling before the model

Mean Squared Error (MSE): 0.002

Mean Absolute Error (MAE): 0.038

R-squared (R²): 0.998

Decision Tree

A Decision Tree is a supervised machine learning algorithm used for both classification and regression tasks. It's like a flowchart that makes decisions based on a series of questions (features) about the data. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

Mean Squared Error (MSE): 0.000

Mean Absolute Error (MAE): 0.000

R-squared (R²): 1.000

Linear Regression - Using Recomende Features :

Linear regression is a classic and widely used regression model that assumes a linear relationship between the features and the target variable (popularity).

Mean Squared Error: 0.002858312975857872

R-squared: 0.996858551693562

Predicted Popularity for sample data: 490.6484978484135

Random Forest Regressor:

An ensemble learning method that builds multiple decision trees and averages their predictions to improve accuracy and robustness.

Mean Squared Error: 5.7072098763723585e-09

R-squared: 0.9999999937274522

Predicted Popularity for sample data: 2.3432450043779682

The XGBoost model- Create and train the model

XGBoost (Extreme Gradient Boosting) is a powerful and popular machine learning algorithm that belongs to the gradient boosting family. It's widely used for both classification and regression tasks and is known for its high accuracy and efficiency.

Root Mean Squared Error (RMSE): 0.004885703271730615

R-squared (R2): 0.9999737654082215

Feature Importance:

```
8        log_popularity  5.749546e-01
6        popularity_log  4.157990e-01
11              success  8.881676e-03
10     popularity_score  2.871540e-04
1    number_of_episodes  6.745893e-05
9   weighted_vote_average  3.520056e-06
0                    id  3.031281e-06
2          vote_average  2.764642e-06
7              air_time  8.600433e-07
3                genres  0.000000e+00
4        origin_country  0.000000e+00
5      episode_run_time  0.000000e+00
```

6.Model Selection and Fine Tuning

Models Selection

Model selection is the process of choosing the best machine learning model from a set of candidate models for a particular task. The goal is to find a model that generalizes well to unseen data and provides the most accurate and reliable predictions.

Model Selection Importance:

Performance: Different models have different strengths and weaknesses. Model selection helps you find the model that performs best for your specific data and problem.

Overfitting: Some models are more prone to overfitting (performing well on training data but poorly on new data). Model selection techniques help identify models that generalize well.

Efficiency: Some models are more computationally expensive than others. Model selection helps you choose a model that balances performance and efficiency.

Interpretability: Some models are easier to understand and interpret than others. Model selection can help you choose a model that is suitable for your needs in terms of interpretability.

Best Model Selection

Metrics:

MSE - Mean Squared Error

RMSE Root Mean Squared Error

MAE Mean Absolute Error - Calculates the average of the absolute differences between predicted and actual values.

RMSLE Root Mean Squared Logarithmic Error

The Best Mosel By MAE :

	model	MSE	RMSE	MAE	RMSLE
1	Decision Tree	4.345310e-26	2.084541e-13	1.425385e-13	8.647858e-13
2	RandomForest	4.818348e-10	2.195073e-05	6.843273e-06	1.567954e-05
6	XGB	4.372417e-06	2.091033e-03	1.092134e-03	1.952858e-03
4	GBM	2.034649e-05	4.510708e-03	2.395548e-03	4.528648e-03
0	Linear Regression	2.577780e-03	5.077184e-02	3.738503e-02	7.455566e-02
5	SVR	2.108591e-03	4.591940e-02	3.870112e-02	1.212030e-01
3	ADABoost	3.898010e-03	6.243404e-02	5.584206e-02	1.860008e-01

The Best Model By RSME :B

	model	MSE	RMSE	MAE	RMSLE
1	Decision Tree	4.345310e-26	2.084541e-13	1.425385e-13	8.647858e-13
2	RandomForest	4.818348e-10	2.195073e-05	6.843273e-06	1.567954e-05
6	XGB	4.372417e-06	2.091033e-03	1.092134e-03	1.952858e-03
4	GBM	2.034649e-05	4.510708e-03	2.395548e-03	4.528648e-03
5	SVR	2.108591e-03	4.591940e-02	3.870112e-02	1.212030e-01
0	Linear Regression	2.577780e-03	5.077184e-02	3.738503e-02	7.455566e-02
3	ADABoost	3.898010e-03	6.243404e-02	5.584206e-02	1.860008e-01

The Best Model By RSMLE:

	model	MSE	RMSE	MAE	RMSLE
1	Decision Tree	4.345310e-26	2.084541e-13	1.425385e-13	8.647858e-13
2	RandomForest	4.818348e-10	2.195073e-05	6.843273e-06	1.567954e-05
6	XGB	4.372417e-06	2.091033e-03	1.092134e-03	1.952858e-03
4	GBM	2.034649e-05	4.510708e-03	2.395548e-03	4.528648e-03
0	Linear Regression	2.577780e-03	5.077184e-02	3.738503e-02	7.455566e-02
5	SVR	2.108591e-03	4.591940e-02	3.870112e-02	1.212030e-01
3	ADABoost	3.898010e-03	6.243404e-02	5.584206e-02	1.860008e-01

Finding the hyperparameters for Best Model -Decision Tree

Grid Search is a hyperparameter tuning technique used in machine learning to find the optimal combination of hyperparameters for a model. It works by systematically exploring a predefined set of hyperparameter values and evaluating the model's performance for each combination. The combination that yields the best performance is selected as the optimal set of hyperparameters.

Best Parameters: {'classifier__max_depth': 3, 'classifier__min_samples_leaf': 1, 'classifier__min_samples_split': 2}

Best Score: -1.0

Best Parameters for DecisionTree Regressor :

1. Regressor_max_depth : 10
2. Regressor_min_samples_leaf: 1
3. Reggressor_min_samples_split 2

Mean Squared Error : 4.967849782870243e-07

R-Squared : 0.9999994503652786

Cross-Validation:

Cross-validation is a crucial technique in machine learning used to evaluate the performance of a model and ensure it generalizes well to unseen data. It helps prevent overfitting, where a model performs well on the training data but poorly on new data.

1. k-Fold Cross-Validation

Reasoning: k-Fold Cross-Validation is a widely used technique that provides a robust estimate of model performance. It involves splitting your data into 'k' equal-sized folds. The model is trained on k-1 folds and tested on the remaining fold. This process is repeated 'k' times, with each fold serving as the test set once. The performance metric (e.g., R-squared, MAE) is averaged across all iterations to obtain a more reliable performance estimate.

```
Average R-squared (5-fold CV): 0.996911629434349

R-squared on Test Set: 0.9968585777947316
```

2. Stratified k-Fold Cross-Validation

Stratified k-Fold Cross-Validation is an extension of k-Fold Cross-Validation that is particularly useful when dealing with imbalanced datasets, where the distribution of the target variable is uneven across classes. It ensures that each fold maintains the same proportion of target classes as the original dataset.

```
Average R-squared (5-fold CV): 0.996911629434349

R-squared on Test Set: 0.9968585777947316
```

Model DecisionTreeRegressor Evaluation

Performance Matrics:

Model evaluating using appropriate metrics such as accuracy, precision, recall, F1-score, ROC-AUC for classification, or RMSE, MAE for regression.

```
Mean Squared Error (MSE): 1.9582028268211006e-08

Mean Absolute Error (MAE): 3.51781870720756e-05

R-squared (R2): 0.999999959224015

Explained Variance Score: 0.9999999592328748

Max Error: 0.003110011475784269

Median Absolute Error: 2.220446049250313e-16

Root Mean Squared Logarithmic Error (RMSLE): 6.293151148013716e-05

Mean Poisson Deviance: 2.1670655455556853e-08

Mean Gamma Deviance: 1.8393206429720925e-07
```

```
D2 Tweedie Score (power=1): 0.999999978696244

D2 Pinball Score (alpha=0.5): 0.9999024951046834

D2 Absolute Error Score: 0.9999024951046834
```