Machine Learning Classification
~ Multi-Domain Data Analysis ~

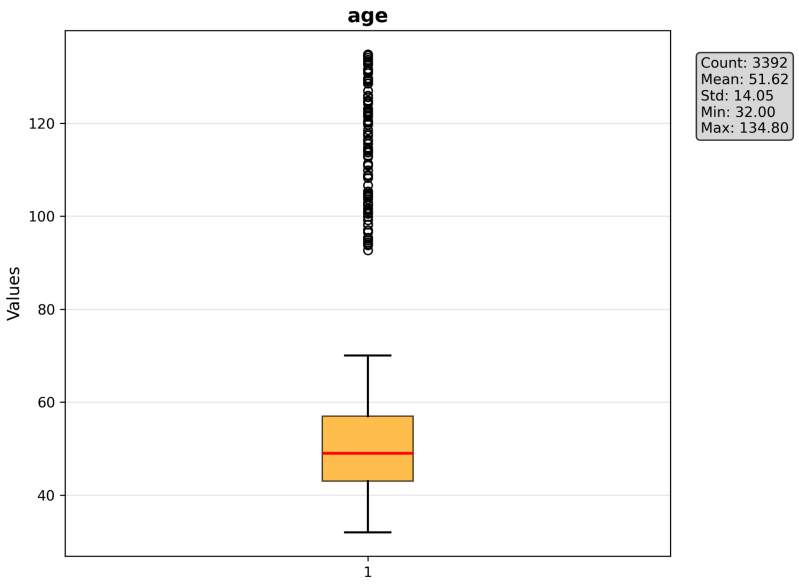
1. Data Analysis (`*analyze_data.py*`)

The first step involved extracting and storing training and testing data for two datasets: online news popularity and coronary heart disease risk. I used the load_heart_set() function to store both data types (training and testing) in a dictionary, and load_news_set() with the same functionality. The resulting dictionaries are heart_set (heart_disease_dataset) and news_set (news_popularity_dataset).

The next step was distinguishing between continuous numerical attributes and ordinal/discrete attributes. I consulted the data type specifications in the documentation appendix for each dataset's attributes. Thus, I separated news_categorical_cols / heart_categorical_cols and news_numerical_cols / heart_numerical_cols. I saved these columns for heart_set['heart_train'] and news_set['news_train'] for future processing.

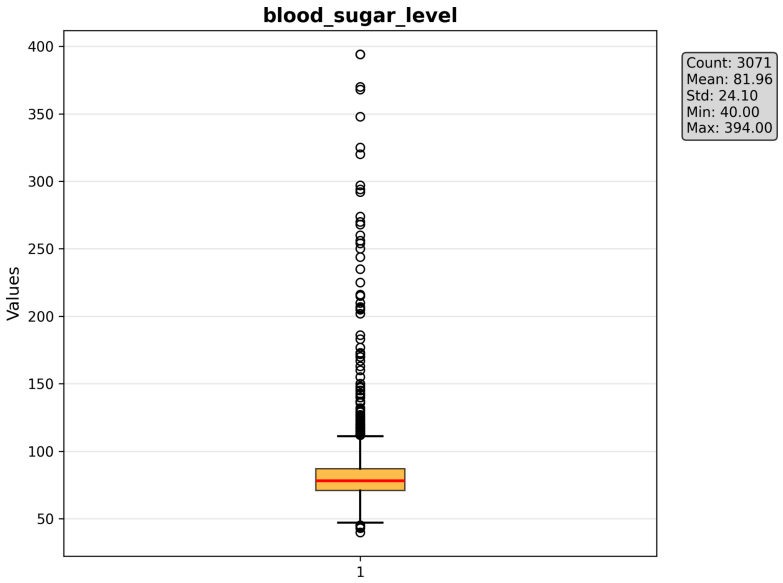
With the data properly categorized, I could begin the actual analysis of both numerical and discrete data. The function used for numerical data analysis is analyze_num_data, which calculates descriptive statistics (mean, std, min, max, quartiles) for each numerical column, generates individual boxplots for each numerical attribute, and saves the graphs as PNG files.

Graphs for Numerical Data Analysis - Coronary Heart Disease:



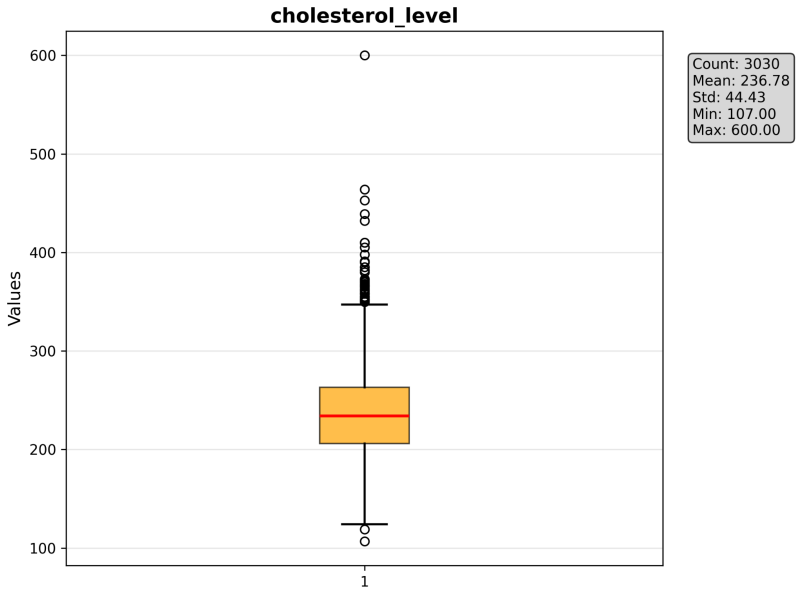
Age

- Distribution: Normal with outliers at advanced ages (>70 years)
- Significance: Mean age of 51 years represents the high-risk group for cardiovascular disease



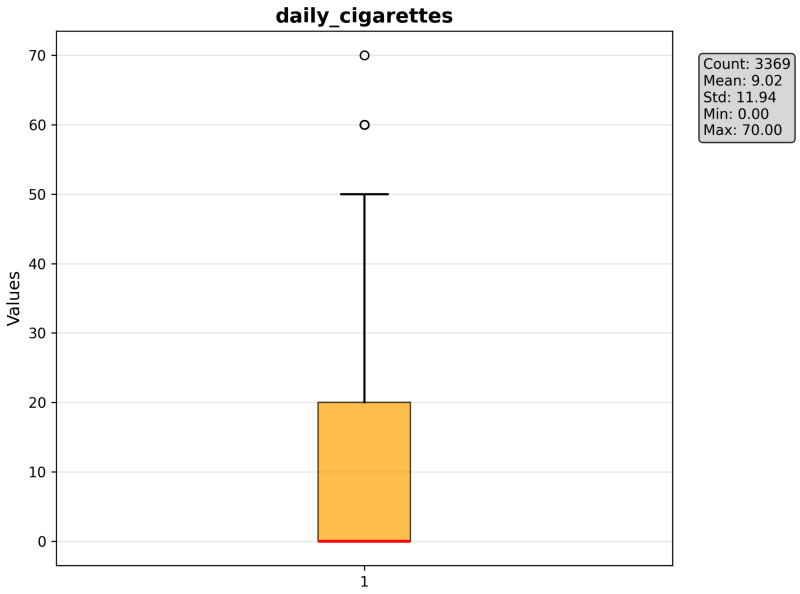
Blood glucose level

- Distribution: Normal with outliers at high values (>200)
- Clinical significance: Mean within normal limits



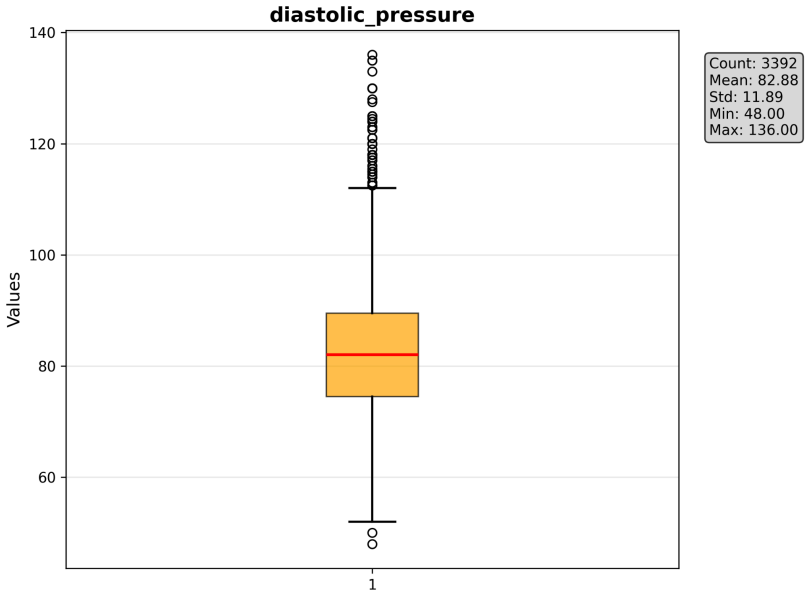
Cholesterol

- Distribution: Slightly skewed with many outliers



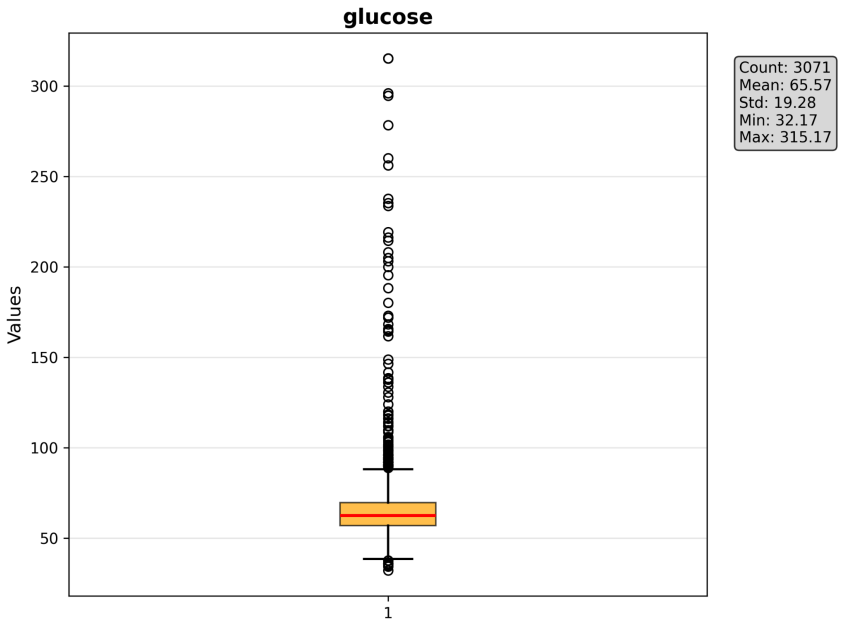
Daily Cigarettes

- Distribution: Extremely skewed
- Observation: Many non-smokers, but smokers consume up to 70 cigarettes/day



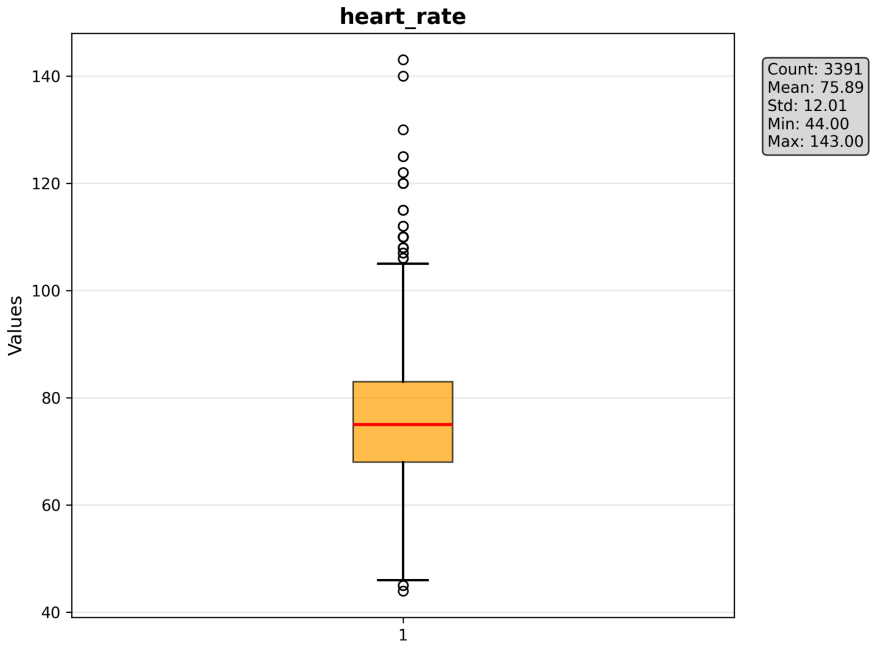
Diastolic Pressure

- Distribution: Nearly normal with some outliers
- Significance: Slightly elevated mean



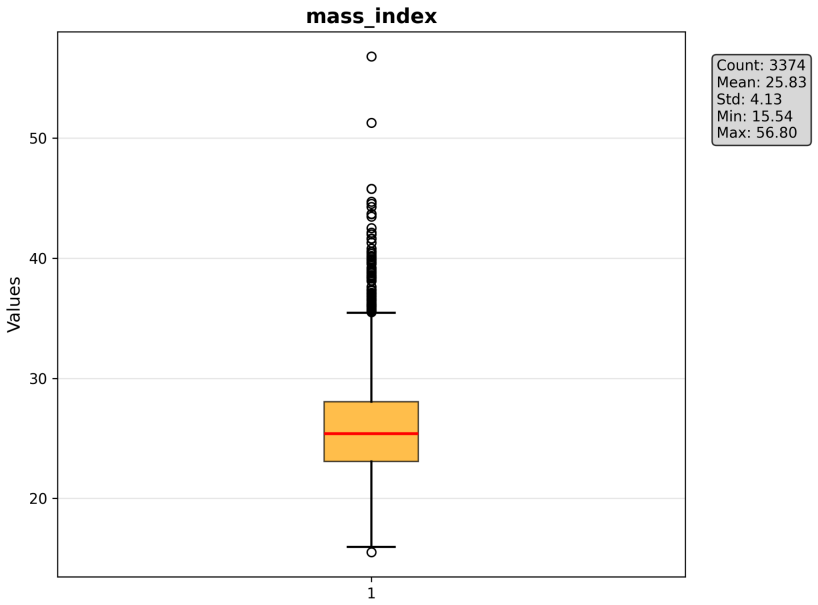
Glucose

- Distribution: Many outliers at high values
- Interpretation: High values indicate metabolic disorders



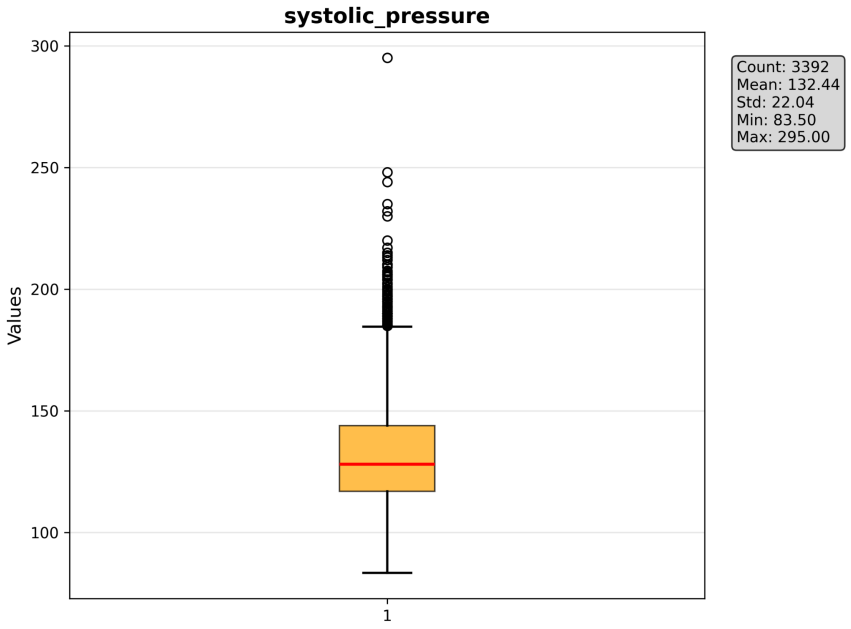
Heart Rate

- Distribution: Normal with outliers at extremes
- Significance: Mean within normal limits



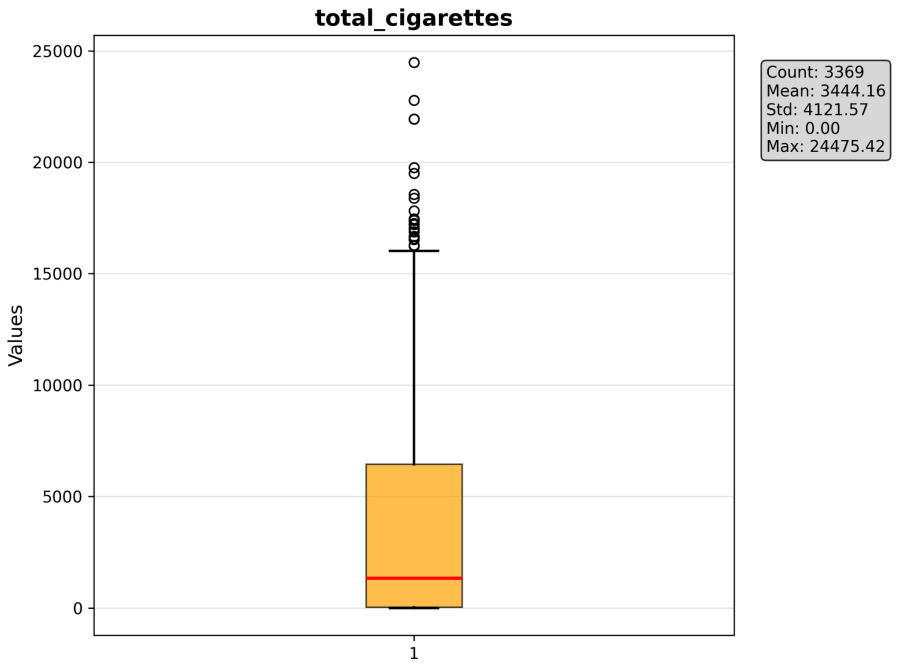
Mass index (BMI)

- Distribution: Normal with outliers at obesity levels



Systolic Pressure

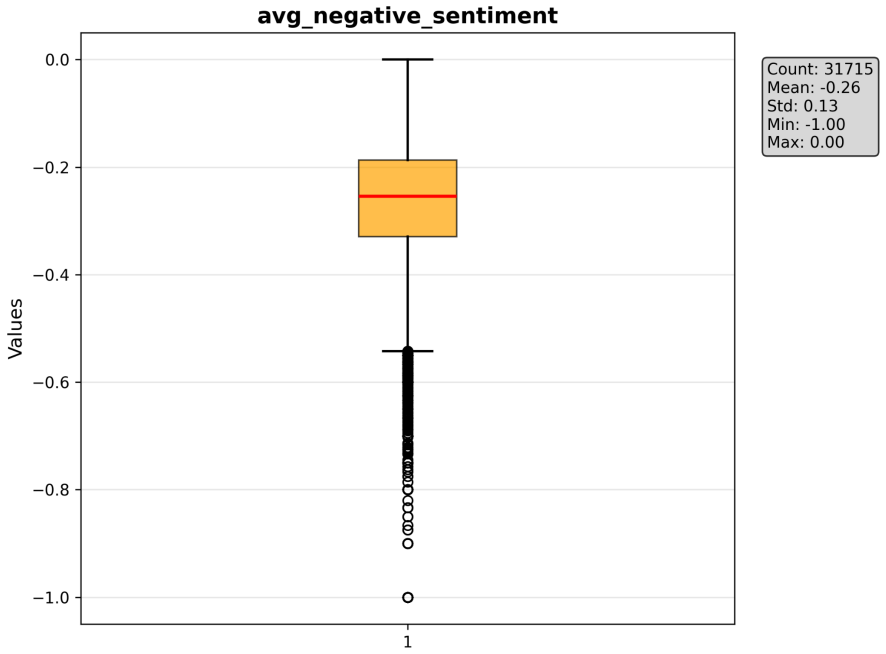
- Distribution: Many outliers at very high values (>200)



Total Cigarettes

- Distribution: Extremely skewed

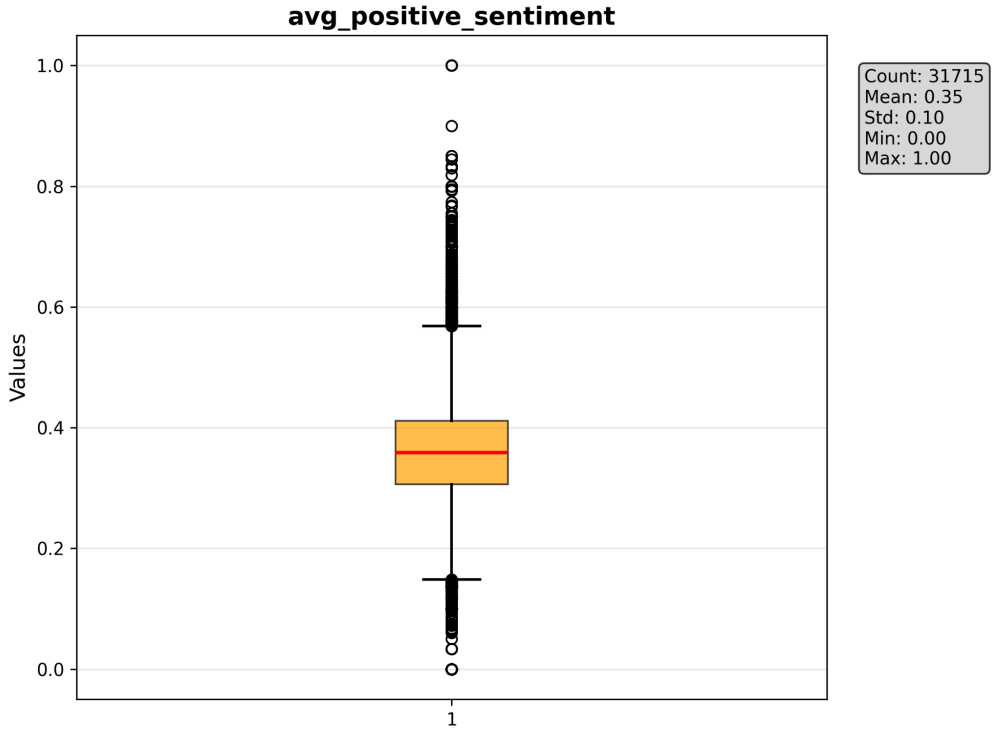
Graphs for Numerical Data Analysis - Online News Popularity:



Average Negative Feeling

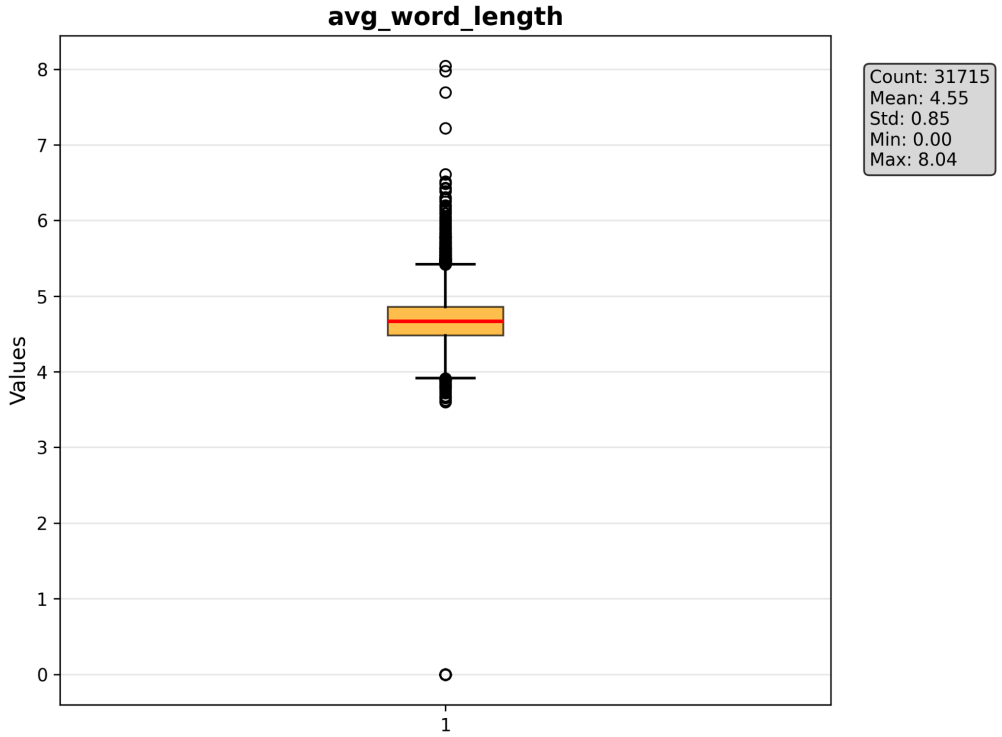
- Distribution: Concentrated between -0.4 and 0, with outliers at -1.0

- Significance: Mean of -0.26 indicates moderately negative feeling



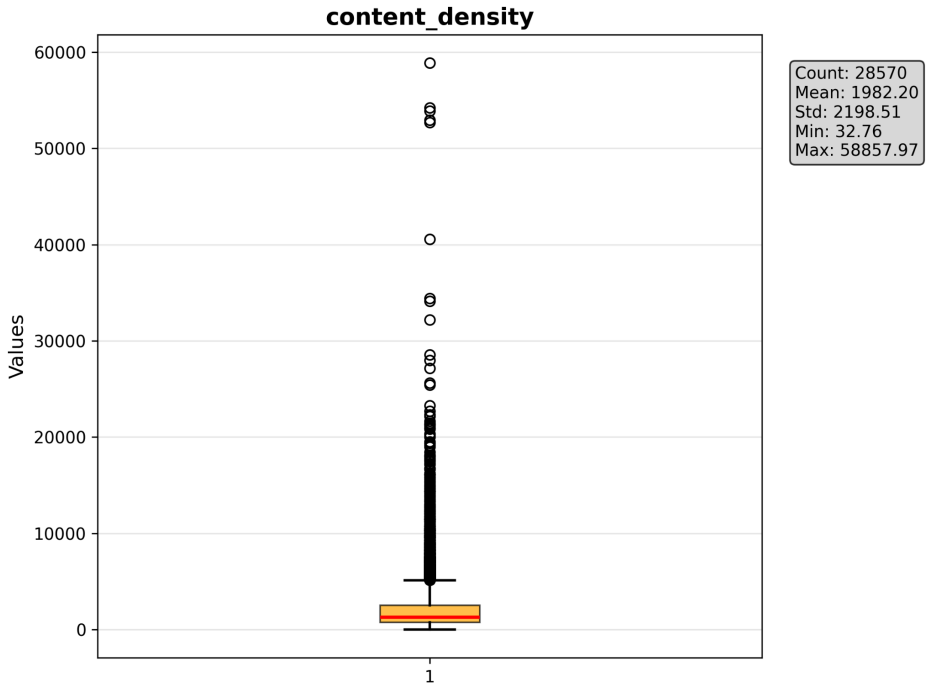
Average Positive Feeling

- Distribution: Concentrated between 0.3-0.4, with outliers up to 1.0
- Significance: Mean of 0.35 indicates moderately positive feeling



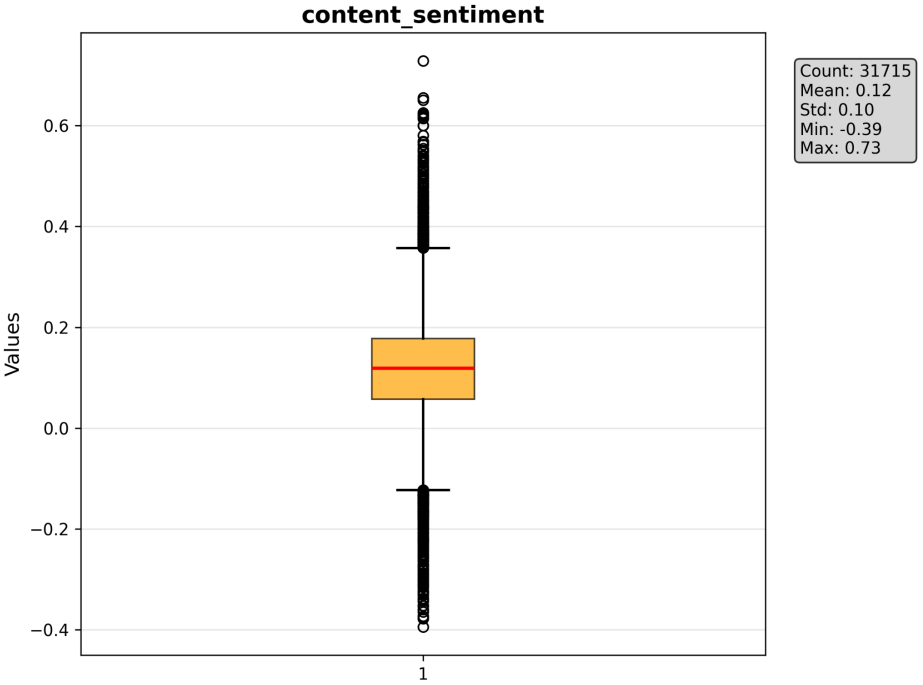
Average Word Length

- Distribution: Normal with some outliers
- Significance: ~4.5 characters/word indicates accessible language



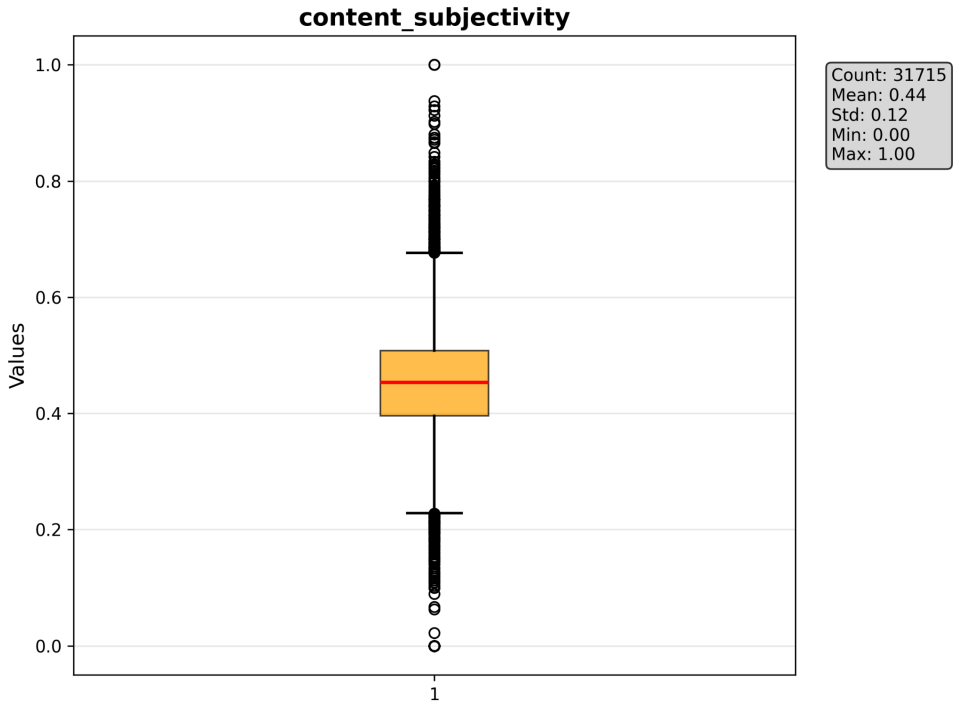
Content Density

- Distribution: Extremely skewed with massive outliers (58,857)
- Interpretation: Most posts have light content, but mega-posts exist



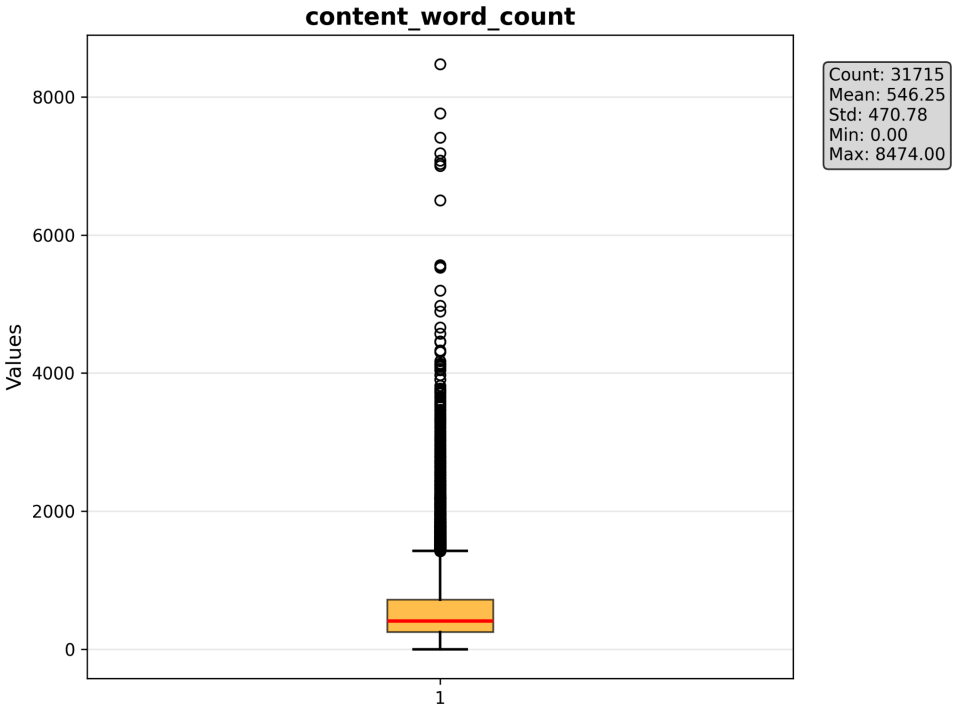
Content Feeling

- Distribution: Centered at 0, nearly neutral
- Significance: Generally neutral-positive sentiment



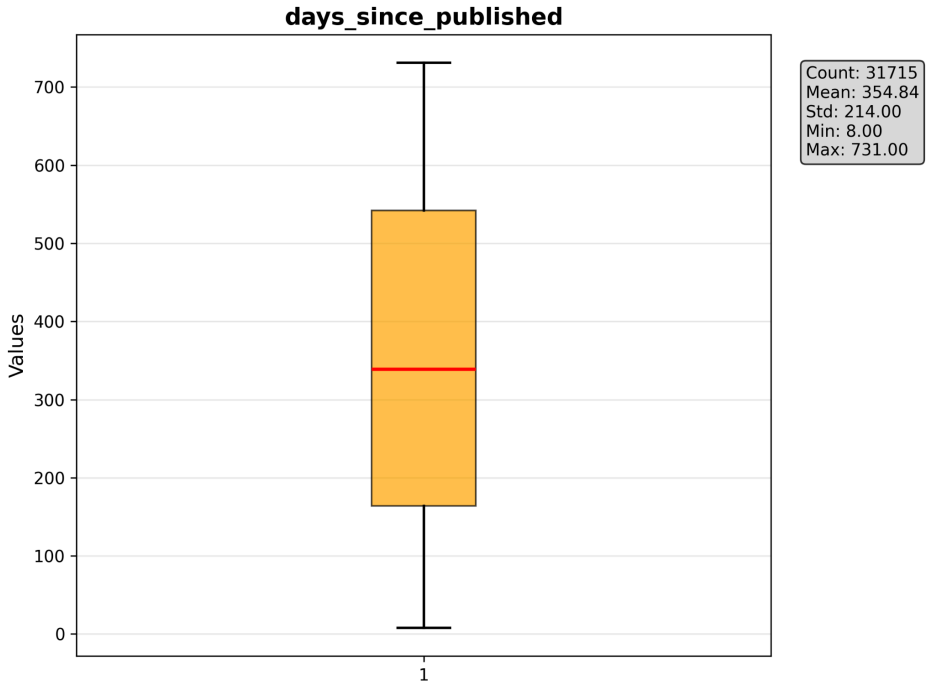
Subjectivity

- Distribution: Centered at 0.4-0.5
- Interpretation: Optimal balance between objective and subjective



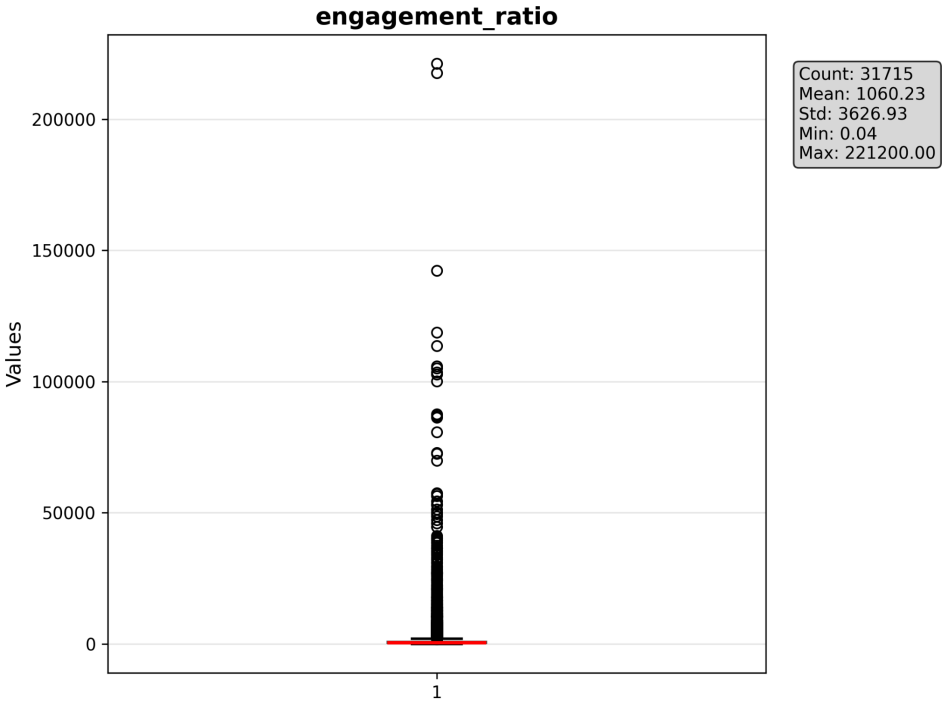
Content Word Count

- Distribution: Very skewed with outliers at 8,474 words
- Posts are of medium length



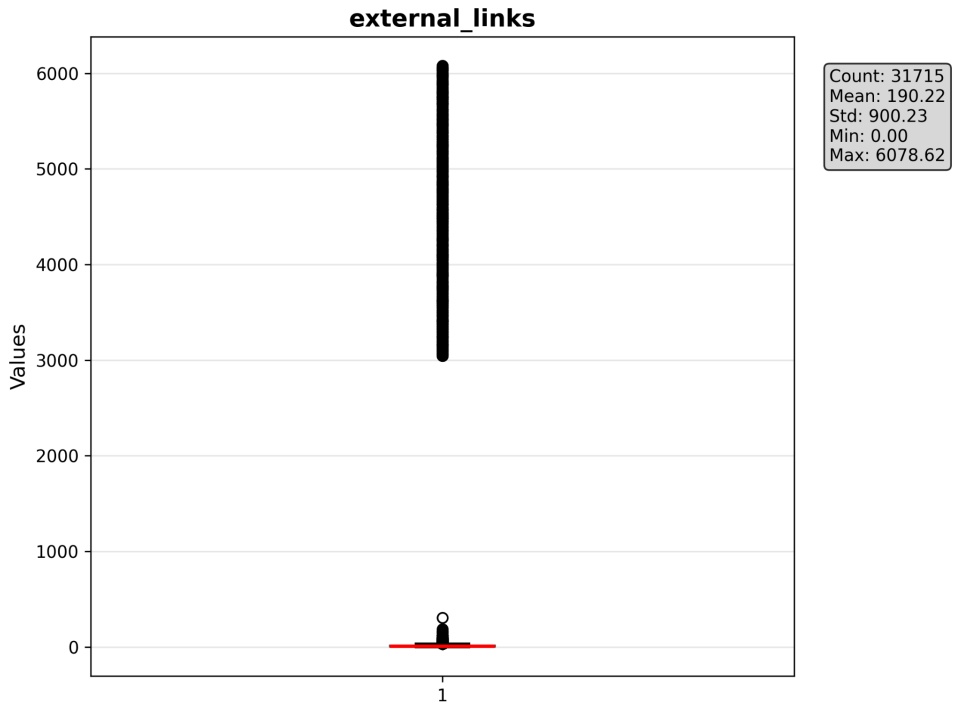
Days Since Published

- Distribution: Nearly normal (8-731 days)



Engagement Ratio

- Distribution: Extremely skewed - outliers at 221,200
- Reality: Most posts have low engagement



External links:

- Interpretation: Most posts without external links

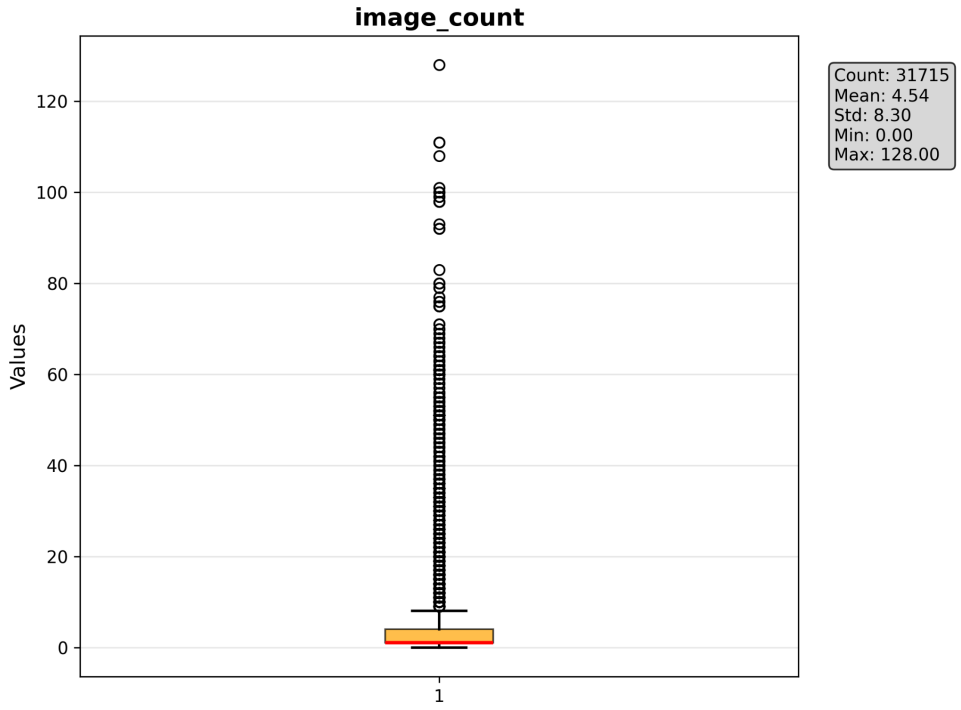
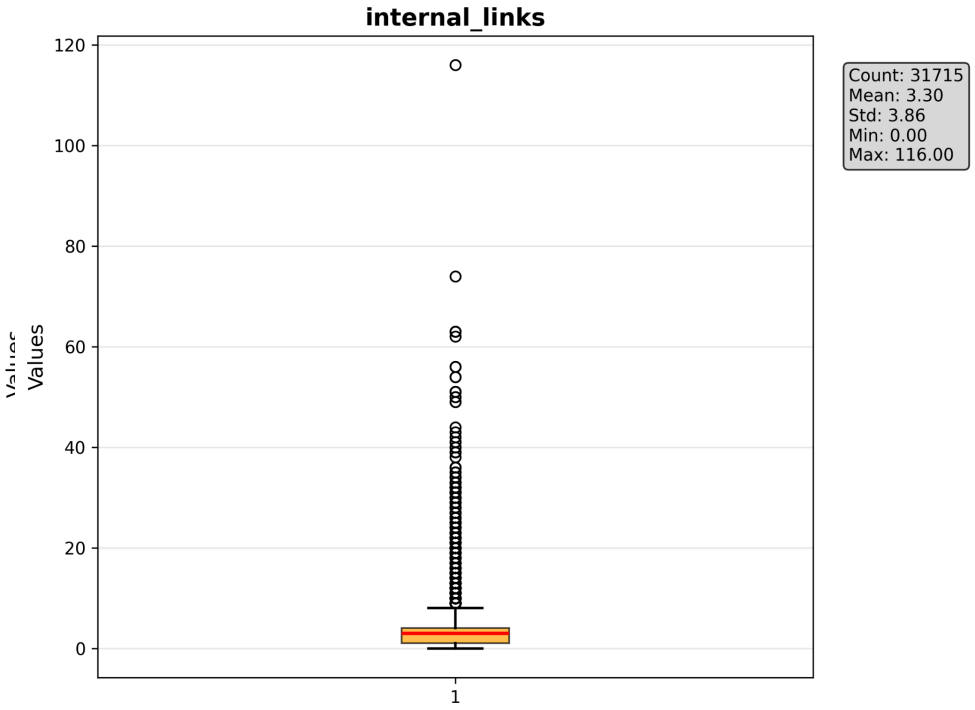


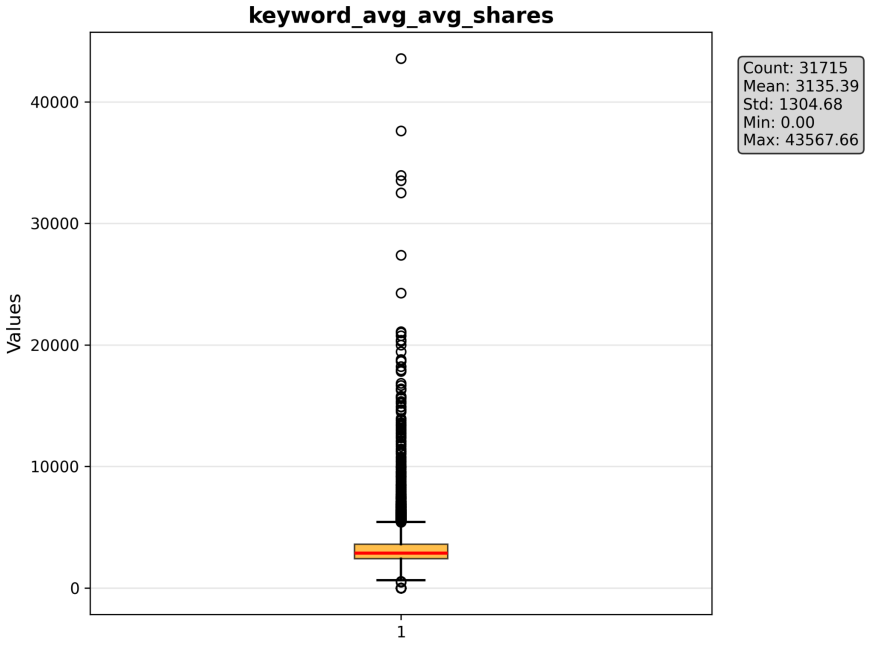
Image Count

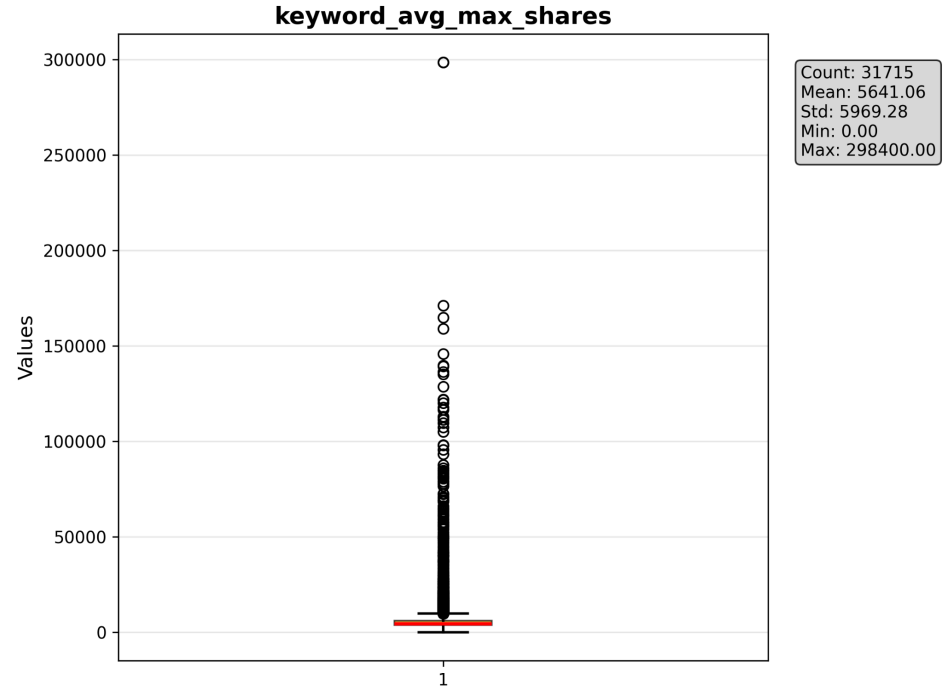
- Distribution: Very skewed ≈ 2-3 images
- Insight: Most posts have 2-3 images, but outliers exist with many images

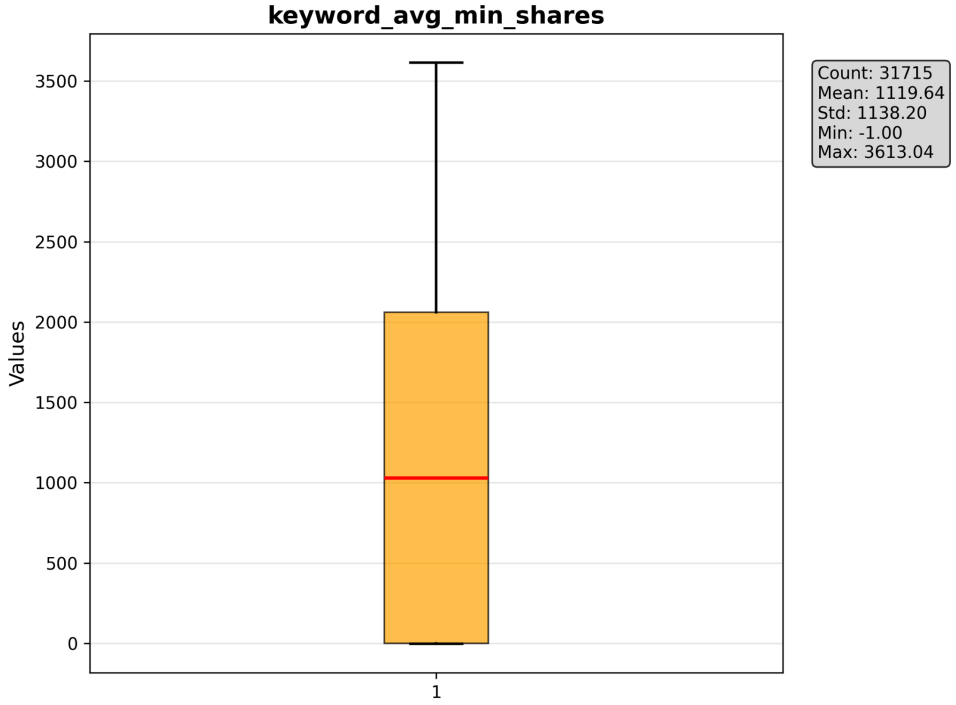


Internal Links

- Distribution: Similar to image_count - skewed toward low
- Comparison: Large difference between internal and external links



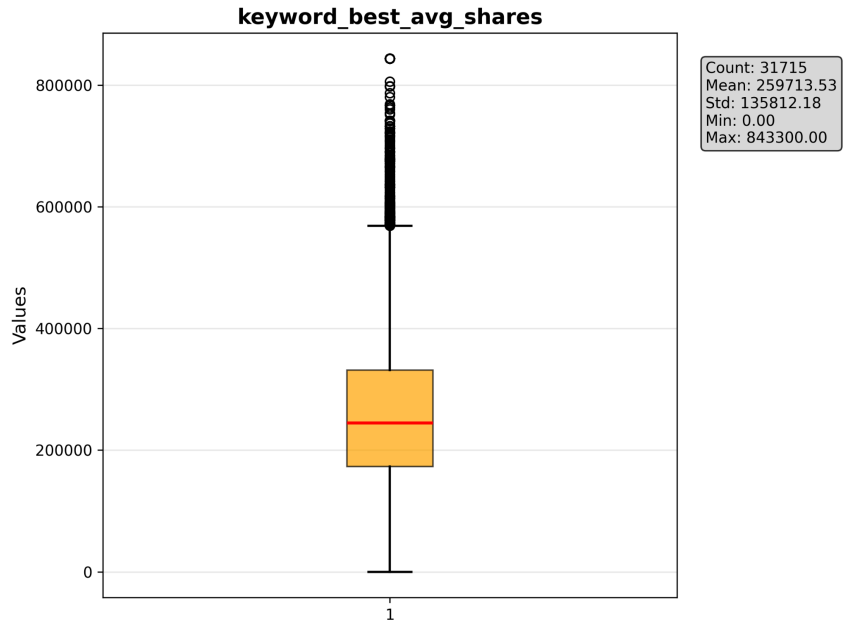


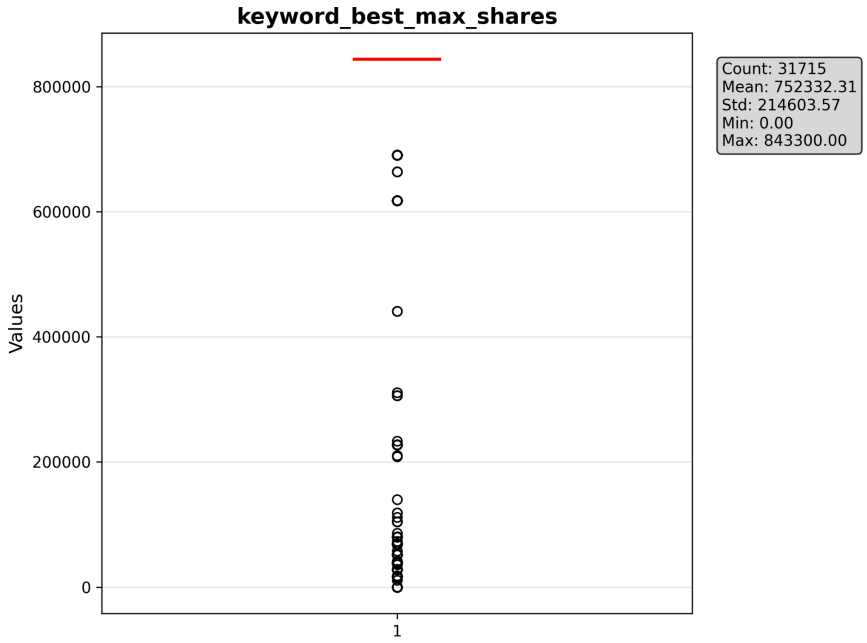


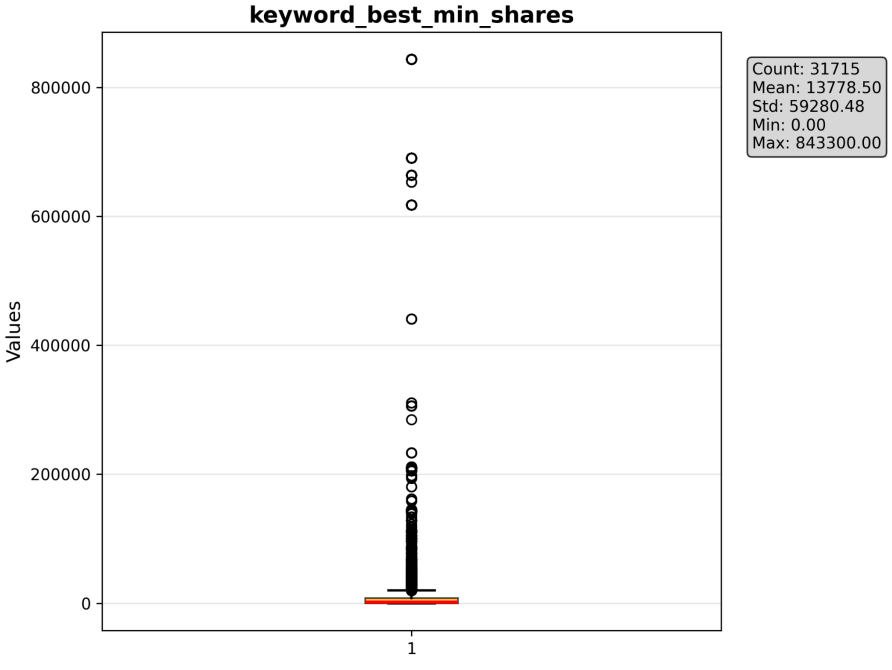
Keyword Metrics:

- avg_shares: 3,135 ± 1,305 (moderate)
- max_shares: 5,641 ± 5,969 (skewed)
- min_shares: 1,120 ± 1,138 (normal distribution)

Average keywords have predictable and consistent performance.


keyword_best_avg_shares
Count: 31715
Mean: 259713.53
Std: 135812.18
Min: 0.00
Max: 843300.00
Values
800000
600000
400000
200000
0
1

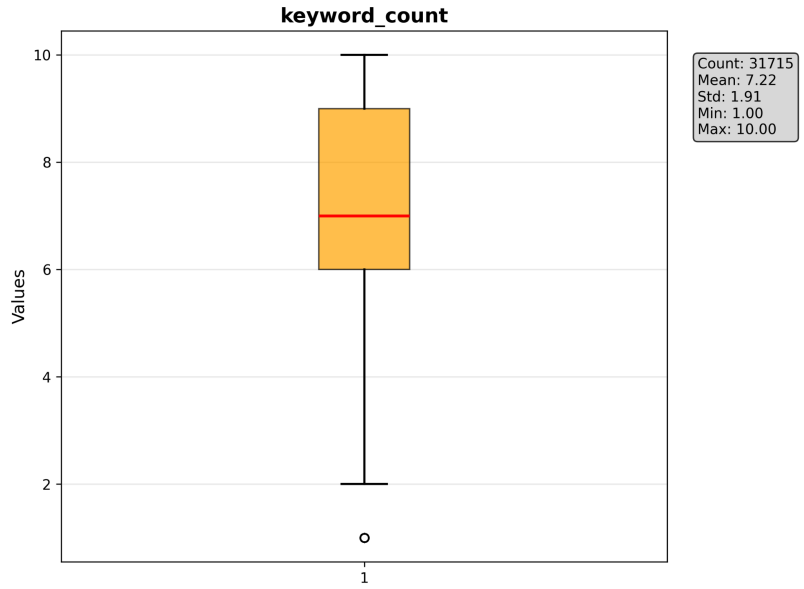

keyword_best_max_shares
Count: 31715
Mean: 752332.31
Std: 214603.57
Min: 0.00
Max: 843300.00
Values
800000
600000
400000
200000
0
1


keyword_best_min_shares
Count: 31715
Mean: 13778.50
Std: 59280.48
Min: 0.00
Max: 843300.00
Values
800000
600000
400000
200000
0
1

Best Keywords:

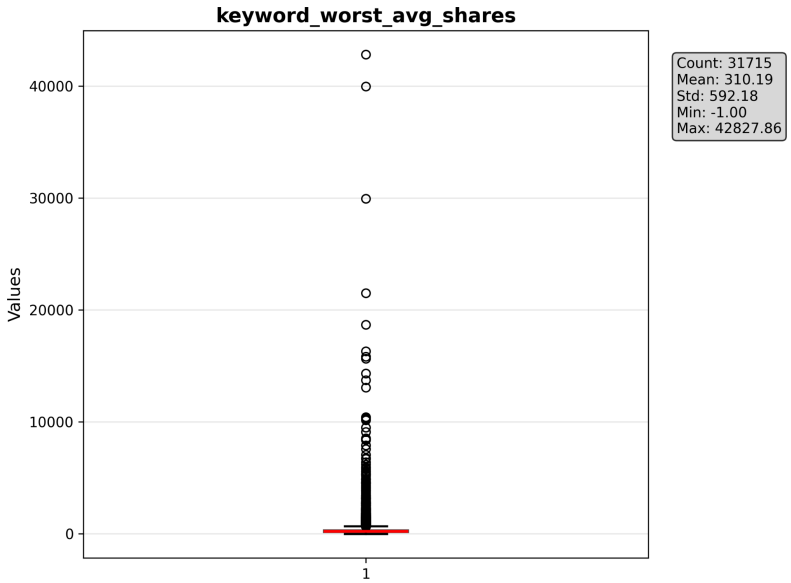
- avg_shares: 259,713 ± 135,812 (very large)
- max_shares: 752,332 ± 214,603 (extremely large)
- min_shares: 13,779 ± 59,280

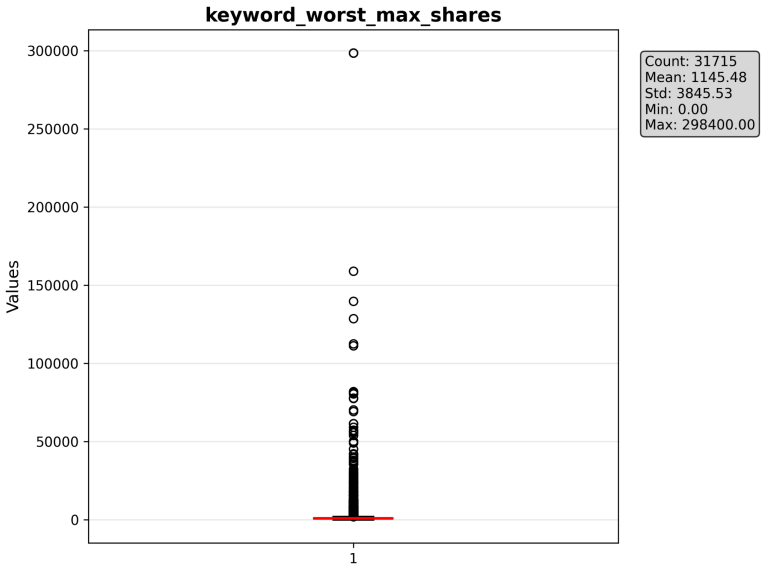
Difference between "best" vs "avg" keywords = 83x more shares

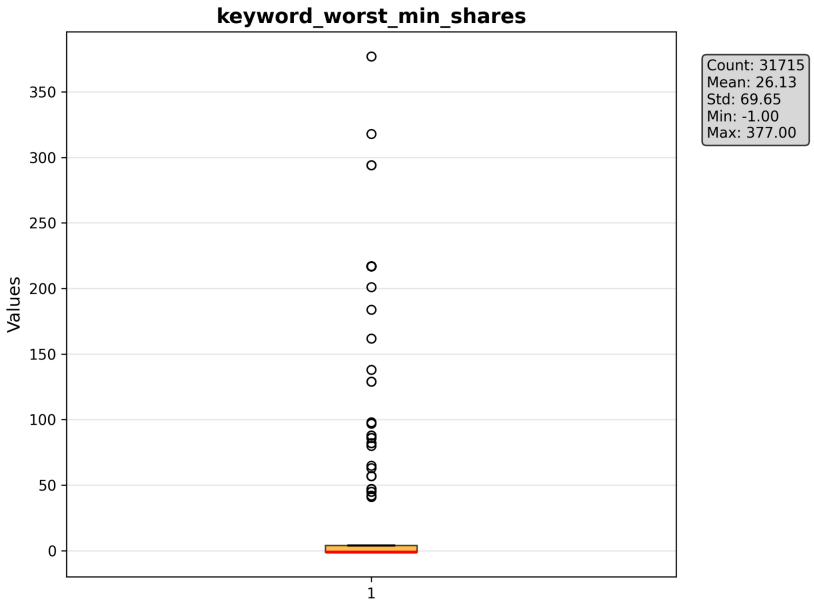


Keyword Count:

- Distribution: Normal (1-10 keywords)



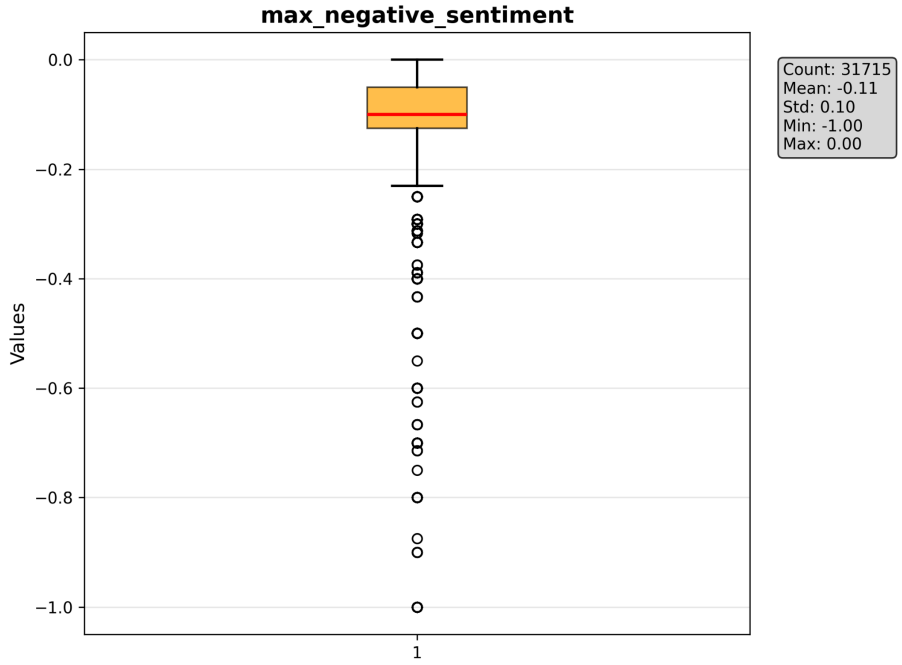


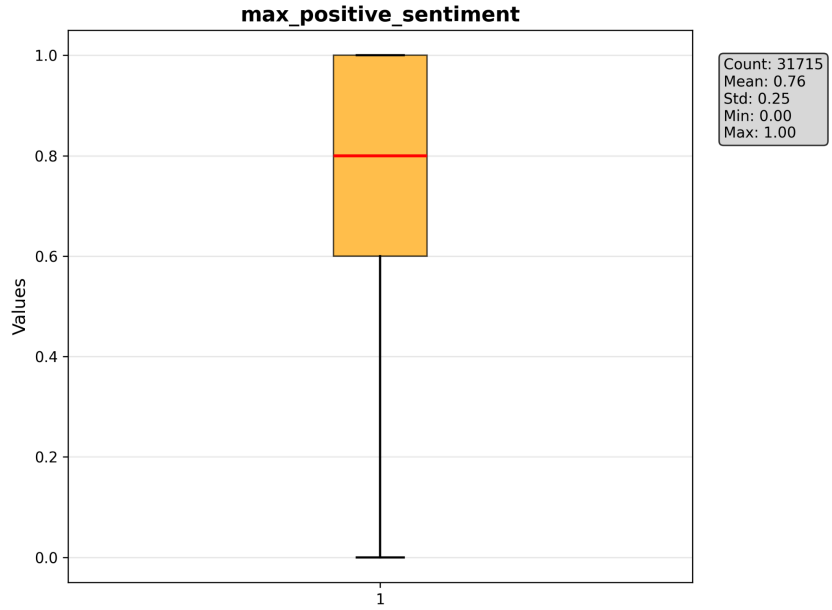


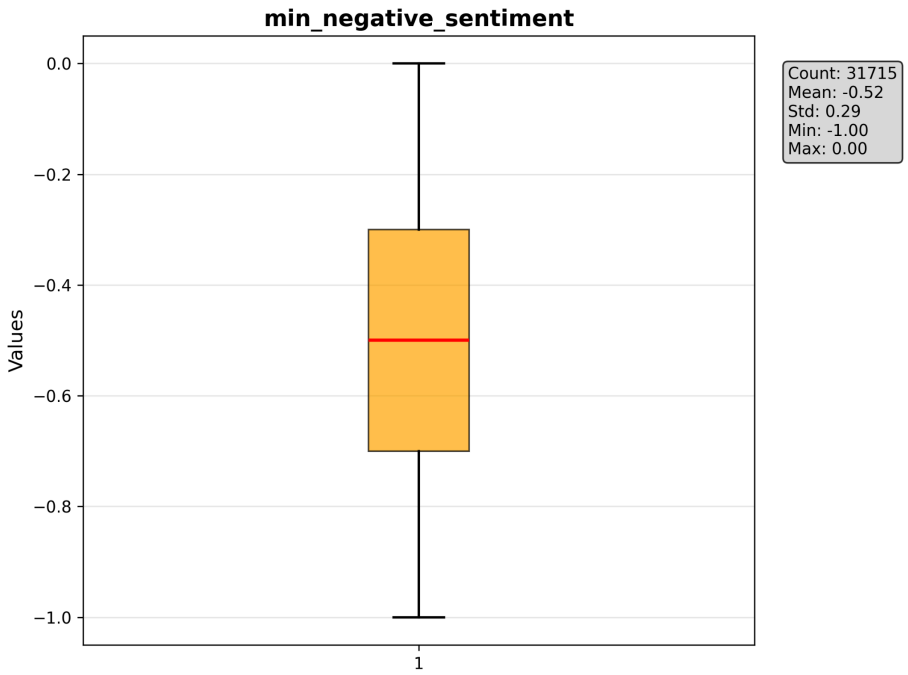
Worst Keywords:

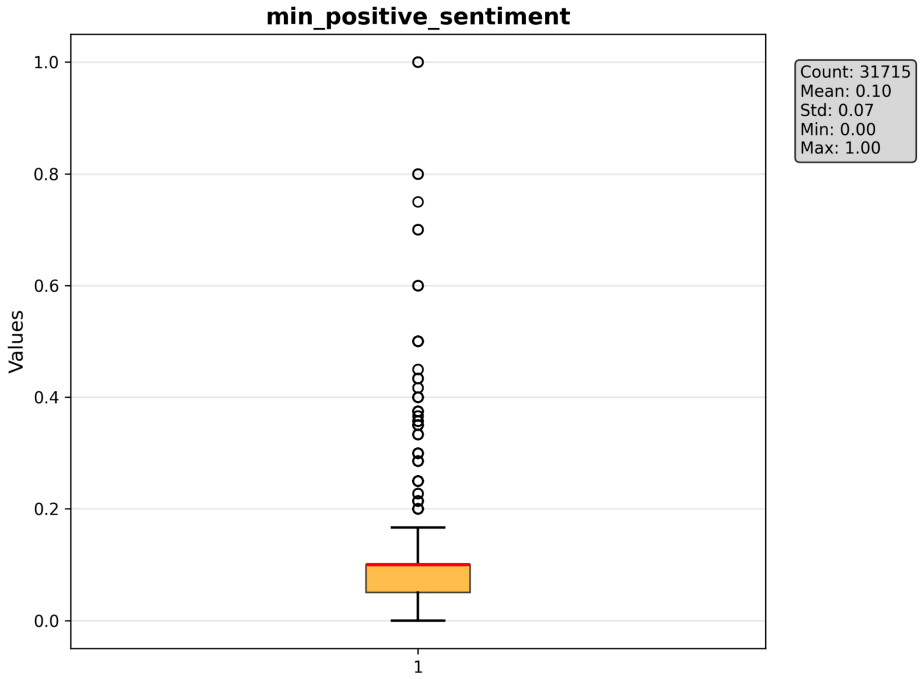
- avg_shares: 310.19 ± 592 (vs best: 259,713)
- max_shares: 1,145 ± 3,845 (vs best: 752,332)
- min_shares: 26.13 ± 69 (nearly zero engagement)

Difference between best vs worst keywords = 2,369x more shares


max_negative_sentiment
Values
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
1
Count: 31715
Mean: -0.11
Std: 0.10
Min: -1.00
Max: 0.00


max_positive_sentiment
Values
1.0
0.8
0.6
0.4
0.2
0.0
1
Count: 31715
Mean: 0.76
Std: 0.25
Min: 0.00
Max: 1.00

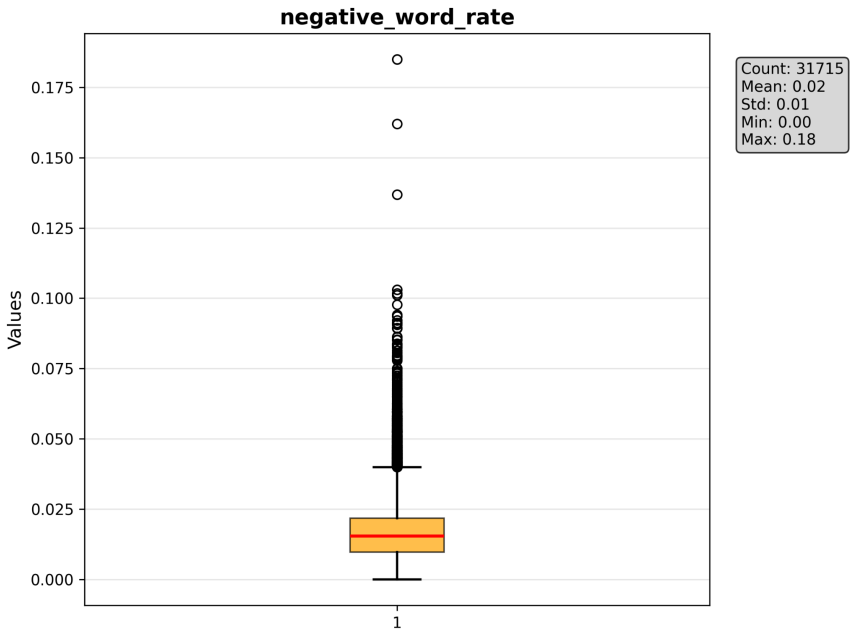


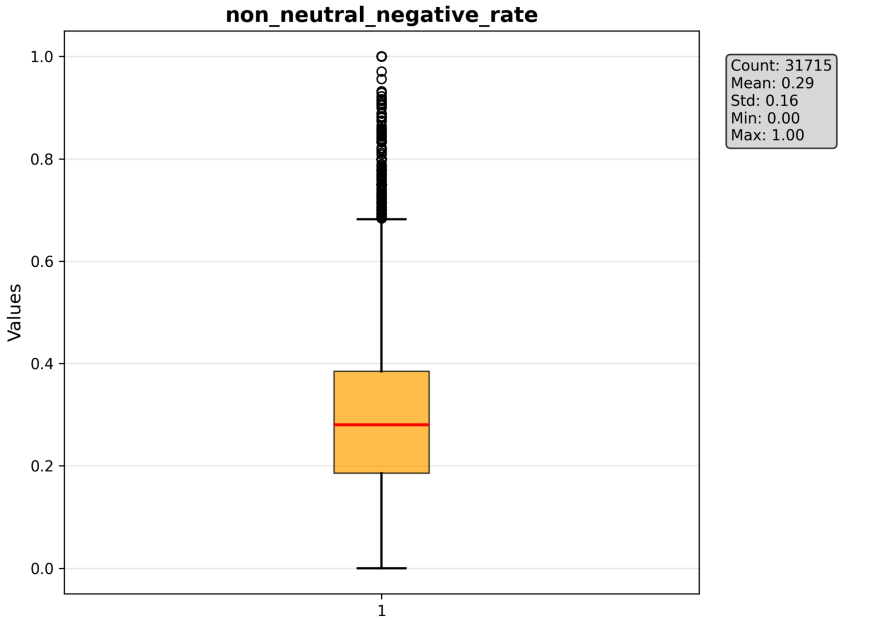


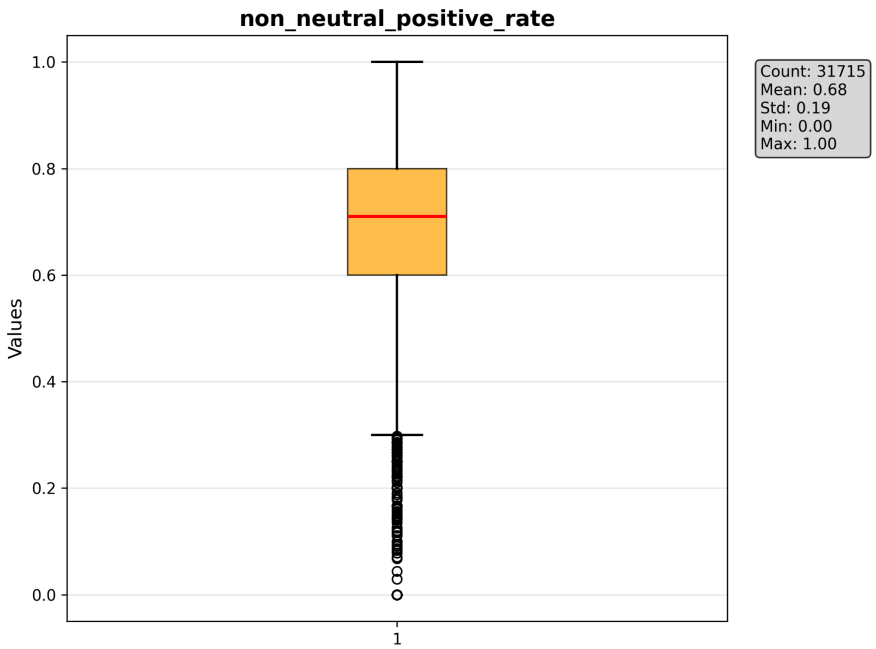
Feeling Extremes:

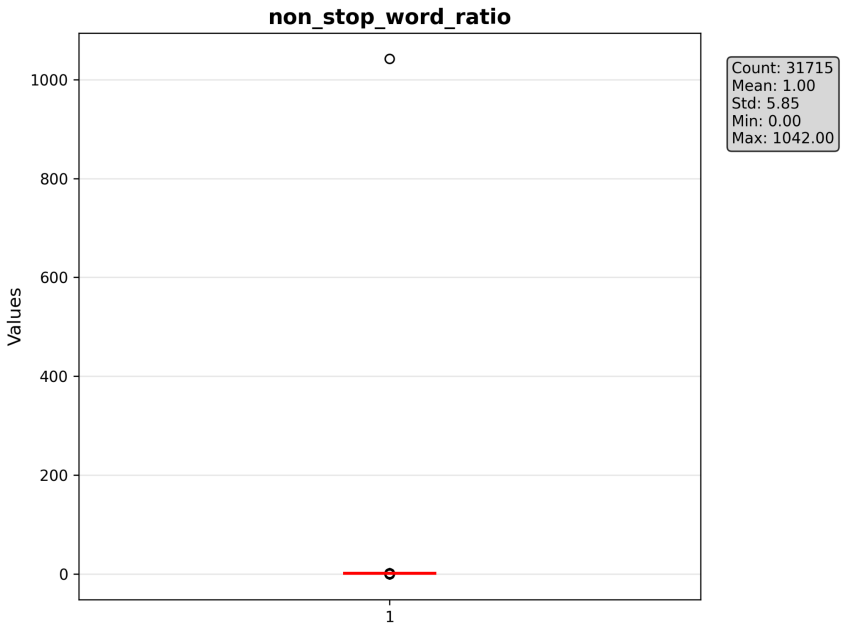
- max_negative: -0.11 ± 0.10 (surprisingly weakly negative)
- max_positive: 0.76 ± 0.25 (strongly positive in most posts)
- min_negative: -0.52 ± 0.29 (wide distribution)
- min_positive: 0.10 ± 0.07 (consistently minimal positive)

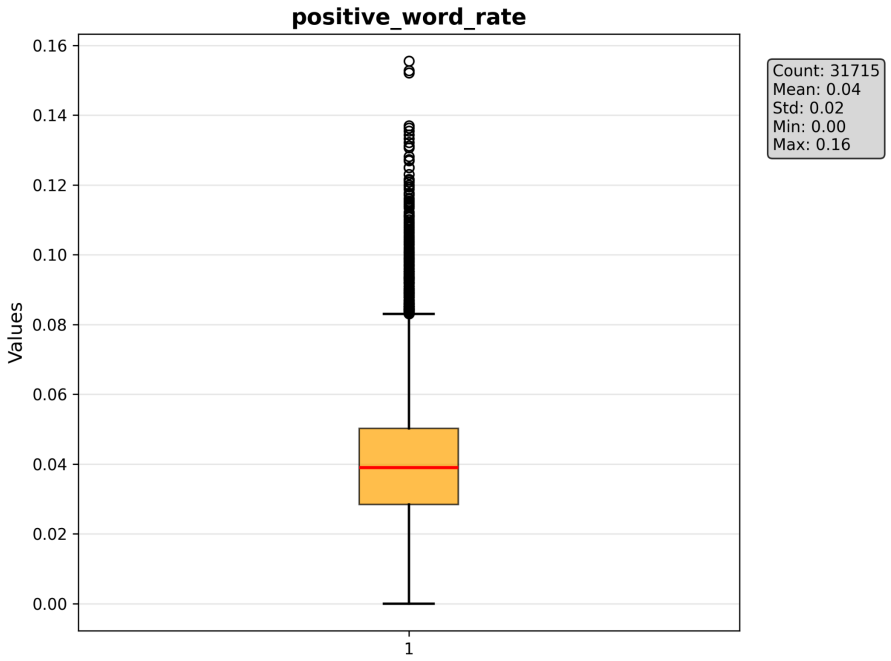
Social media content is generally positive (0.76 avg positive vs -0.11 avg negative)


negative_word_rate
Count: 31715
Mean: 0.02
Std: 0.01
Min: 0.00
Max: 0.18
Values
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
1


non_neutral_negative_rate
Count: 31715
Mean: 0.29
Std: 0.16
Min: 0.00
Max: 1.00
Values
1.0
0.8
0.6
0.4
0.2
0.0
1

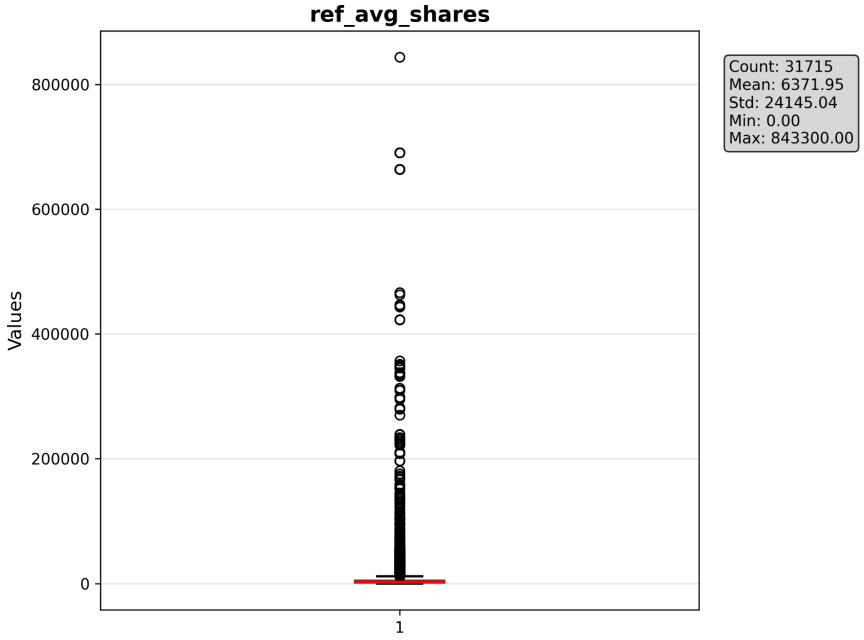

non_neutral_positive_rate
Count: 31715
Mean: 0.68
Std: 0.19
Min: 0.00
Max: 1.00
Values
1.0
0.8
0.6
0.4
0.2
0.0
1

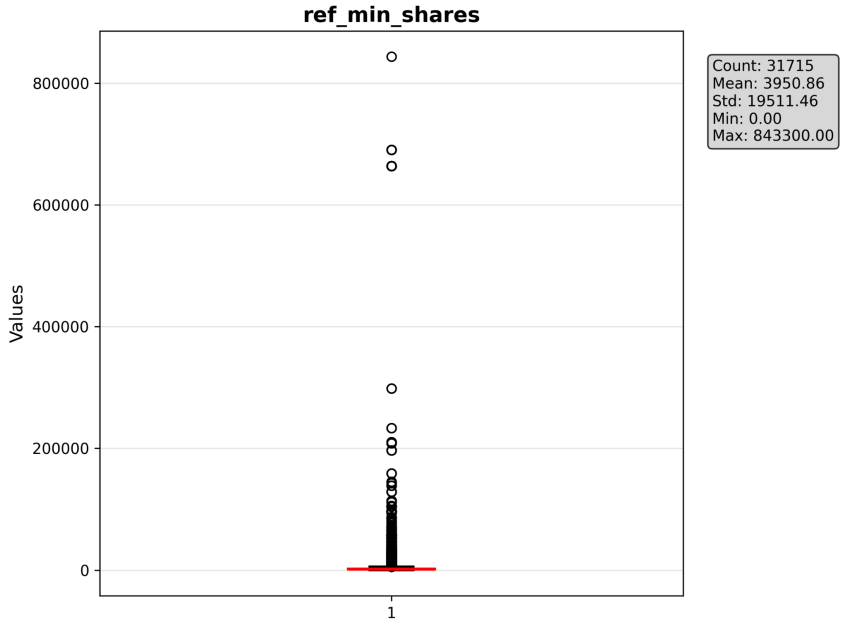


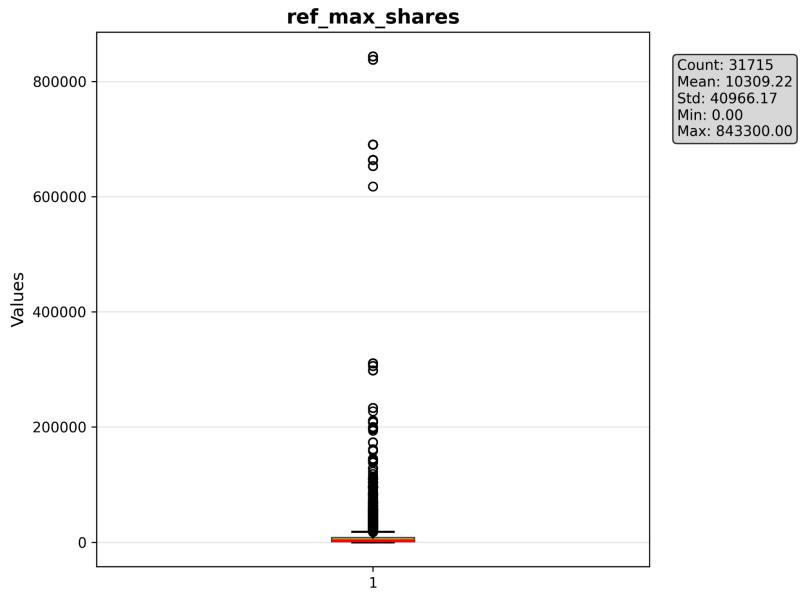


Word Rate Metrics:

- negative_word_rate: 0.02 ± 0.01 (only 2% negative words)
- positive_word_rate: 0.04 ± 0.02 (4% positive words - double)
- non_neutral_negative: 0.29 ± 0.16 (29% non-neutral negative content)
- non_neutral_positive: 0.68 ± 0.19 (68% positive content - dominant)
- non_stop_word_ratio: 1.00 ± 5.85 (extreme outliers)

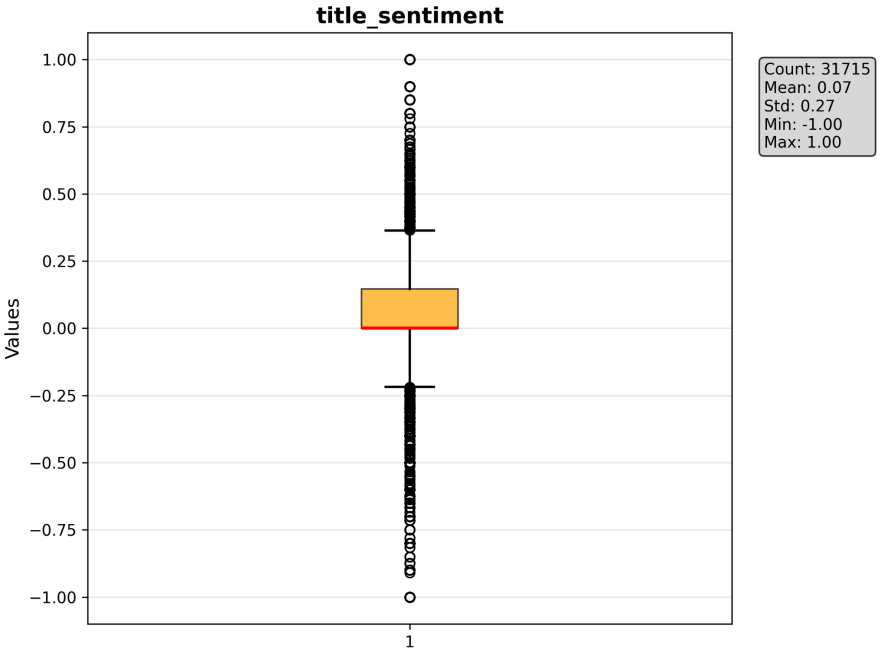

ref_avg_shares
Count: 31715
Mean: 6371.95
Std: 24145.04
Min: 0.00
Max: 843300.00
Values
800000
600000
400000
200000
0
1

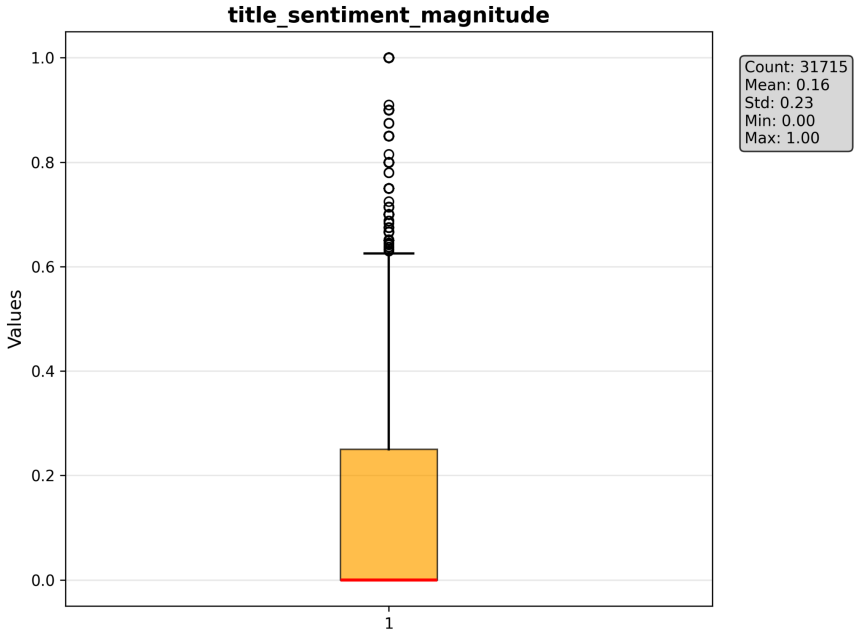

ref_min_shares
Count: 31715
Mean: 3950.86
Std: 19511.46
Min: 0.00
Max: 843300.00
Values
800000
600000
400000
200000
0
1


ref_max_shares
Count: 31715
Mean: 10309.22
Std: 40966.17
Min: 0.00
Max: 843300.00
Values
800000
600000
400000
200000
0
1

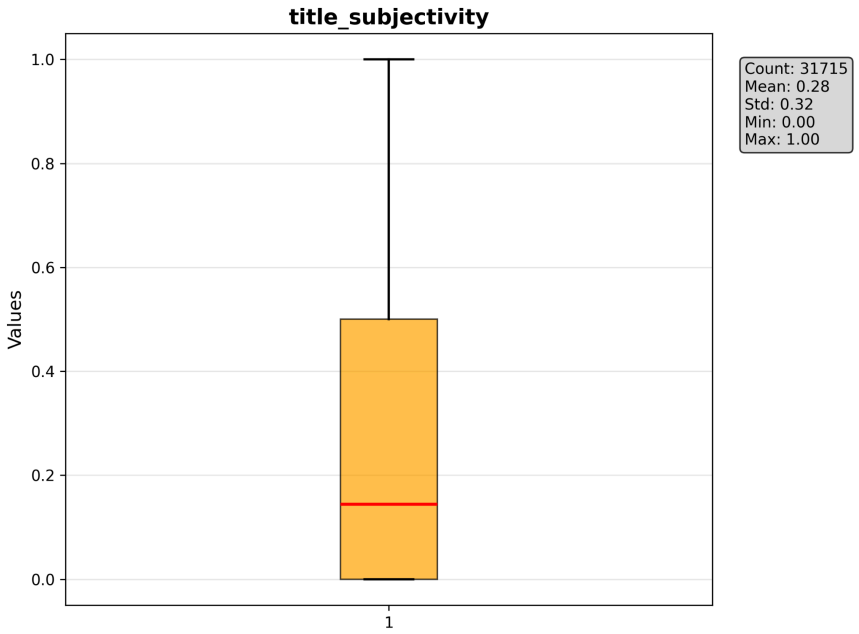
Reference Shares:

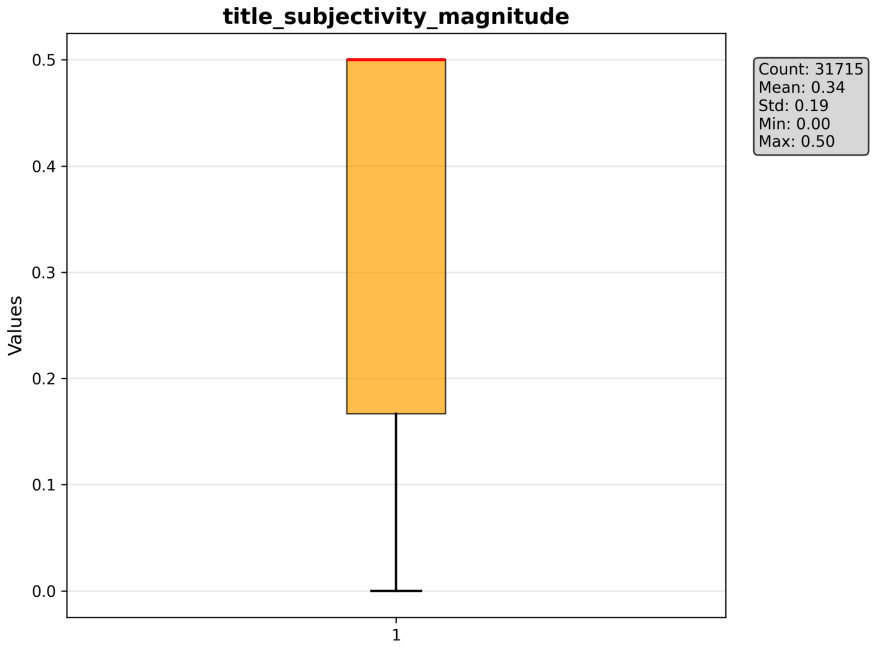
- ref_avg_shares: Most articles remain unseen
- ref_max_shares: Certain article references can perform better than the article itself
- ref_min_shares: Indicates a baseline engagement level for most content



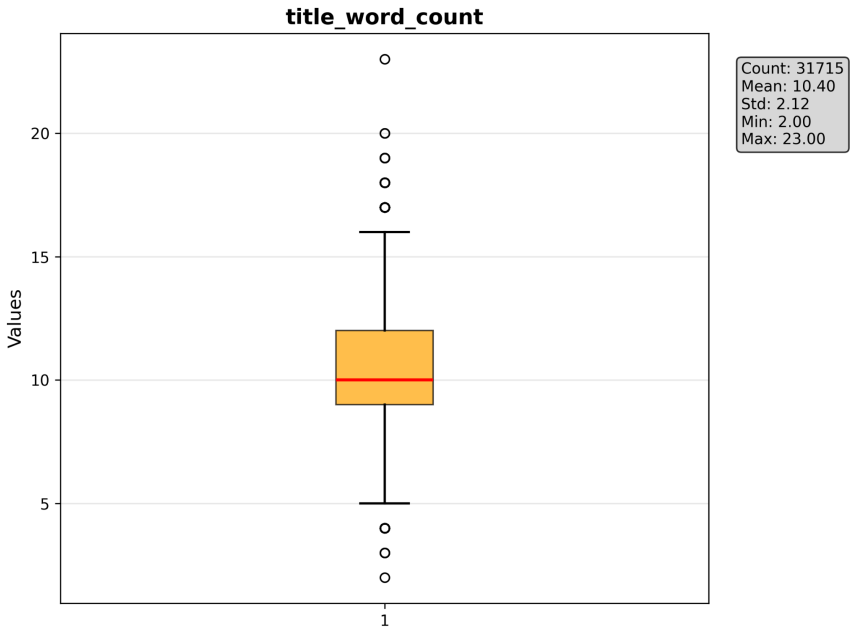


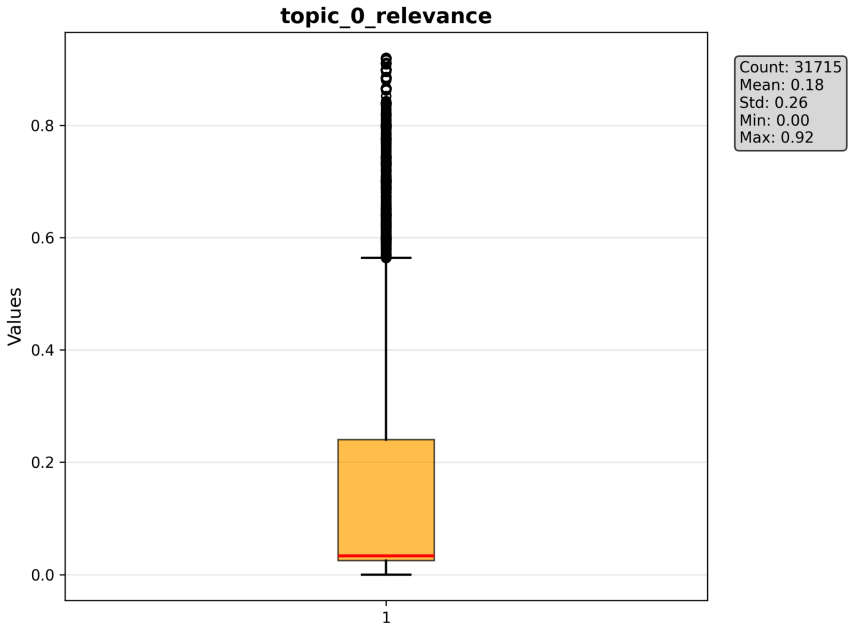
- **Slight positive tendency overall** - mean feeling is 0.07, indicating titles tend to be slightly positive rather than negative. However, distribution is fairly balanced around neutrality.
- **Low emotional intensity** - feeling magnitude averages only 0.16, meaning most titles are emotionally neutral rather than strongly positive or negative.

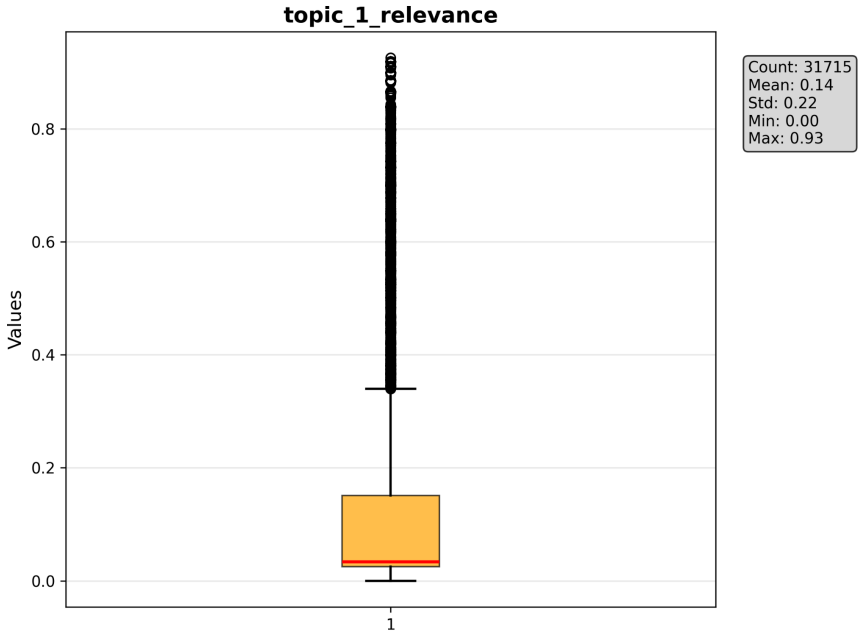


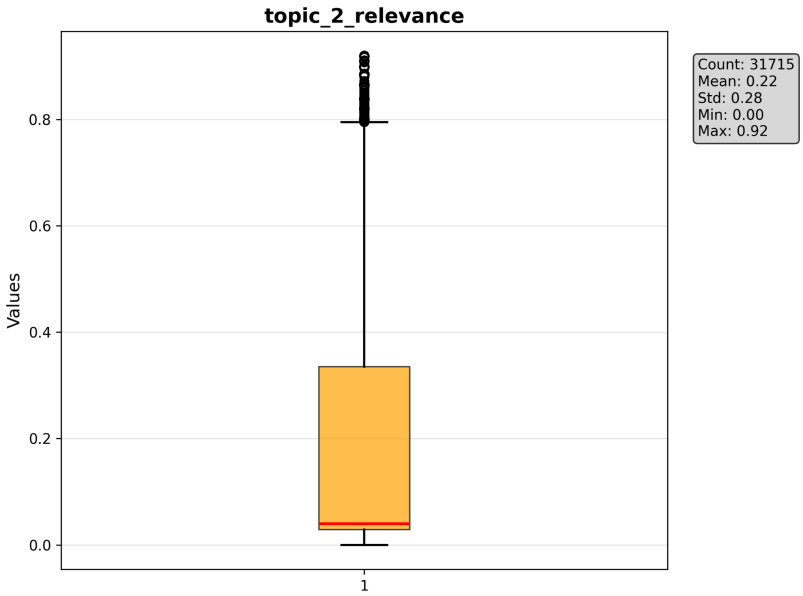


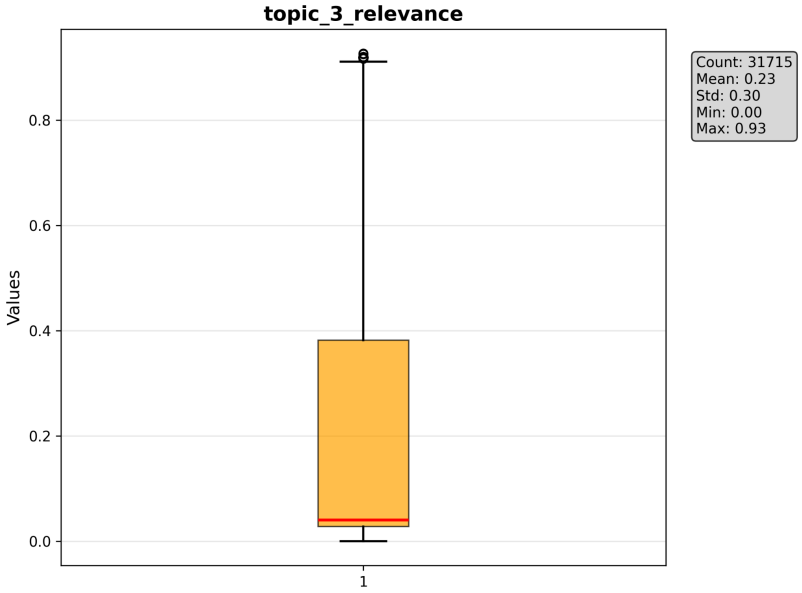
- **Predominantly objective -** with mean subjectivity of 0.28, most titles present information factually rather than opinion-based.
- **Consistent objectivity -** subjectivity magnitude metric (mean 0.34) shows relatively tight clustering, indicating consistent editorial standards across the dataset.

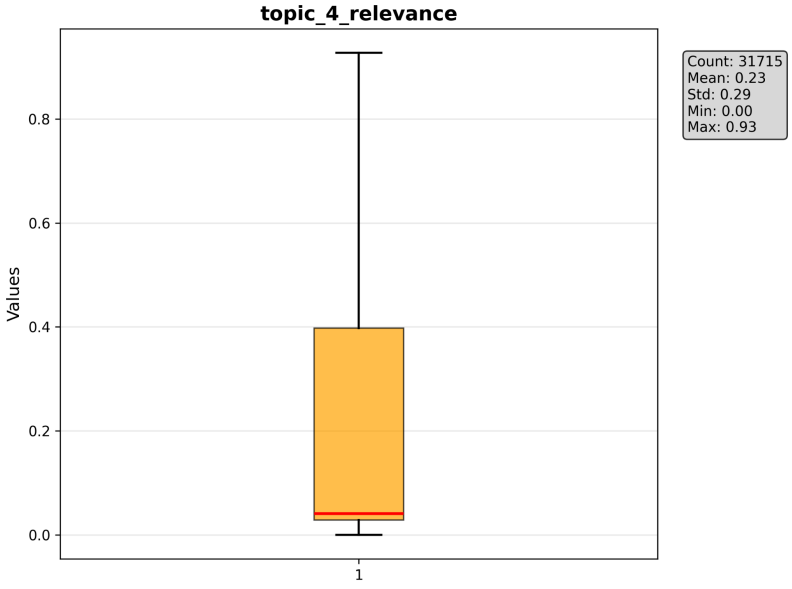

title_word_count
Count: 31715
Mean: 10.40
Std: 2.12
Min: 2.00
Max: 23.00
Values
20
15
10
5
1


topic_0_relevance
Count: 31715
Mean: 0.18
Std: 0.26
Min: 0.00
Max: 0.92
Values
0.8
0.6
0.4
0.2
0.0
1


topic_1_relevance
Count: 31715
Mean: 0.14
Std: 0.22
Min: 0.00
Max: 0.93
Values
0.8
0.6
0.4
0.2
0.0
1


topic_2_relevance
Count: 31715
Mean: 0.22
Std: 0.28
Min: 0.00
Max: 0.92
Values
0.0
0.2
0.4
0.6
0.8
1

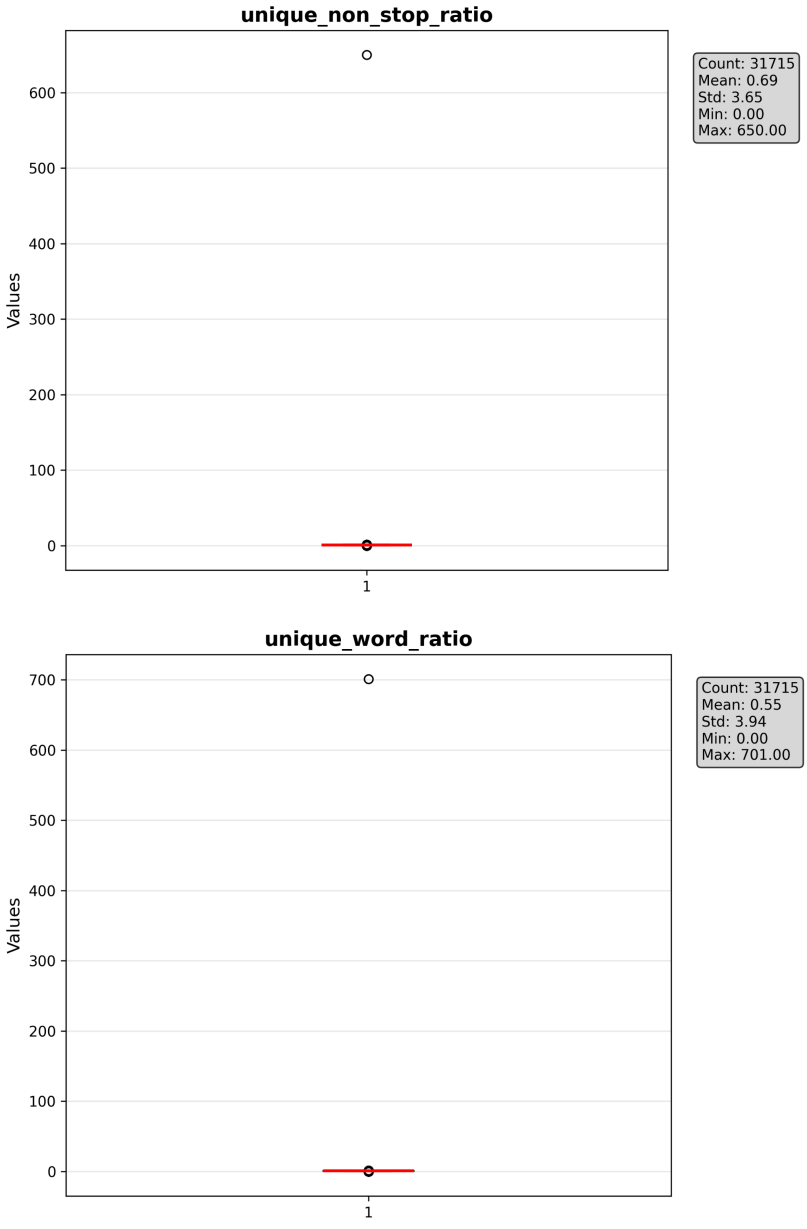

topic_3_relevance
Count: 31715
Mean: 0.23
Std: 0.30
Min: 0.00
Max: 0.93
Values
0.0
0.2
0.4
0.6
0.8
1


topic_4_relevance
Count: 31715
Mean: 0.23
Std: 0.29
Min: 0.00
Max: 0.93
Values
0.0
0.2
0.4
0.6
0.8
1

Title Length Analysis: Consistent professional standards

Topic Relevance:

- Secondary topics (0 and 1): Topic 0 - mean 0.18, strong concentration toward zero, suggesting it's a specialized topic applying only to a small portion / Topic 1 - smallest mean (0.14), indicating the least relevant of the 5 topics, possibly a very specific subject
- Dominant topics (2, 3, 4): Balanced relevance - all three have similar means (0.22-0.23), suggesting these are the main content categories, likely covering major news topics

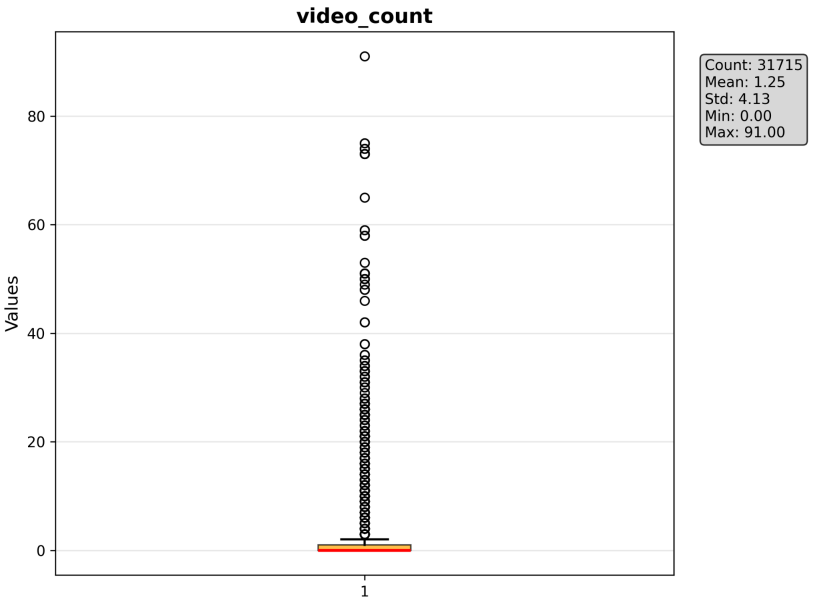


Unique Non-Stop Word Ratio: Most articles are conventional

- Exceptional outliers - extreme values up to 650 suggest existence of articles with highly specialized or technical terminology, possibly scientific articles

Total Unique Word Ratio

- Similar but narrower pattern - mean of 0.55 indicates that when common words are included, uniqueness decreases, confirming most content uses standard linguistic structures

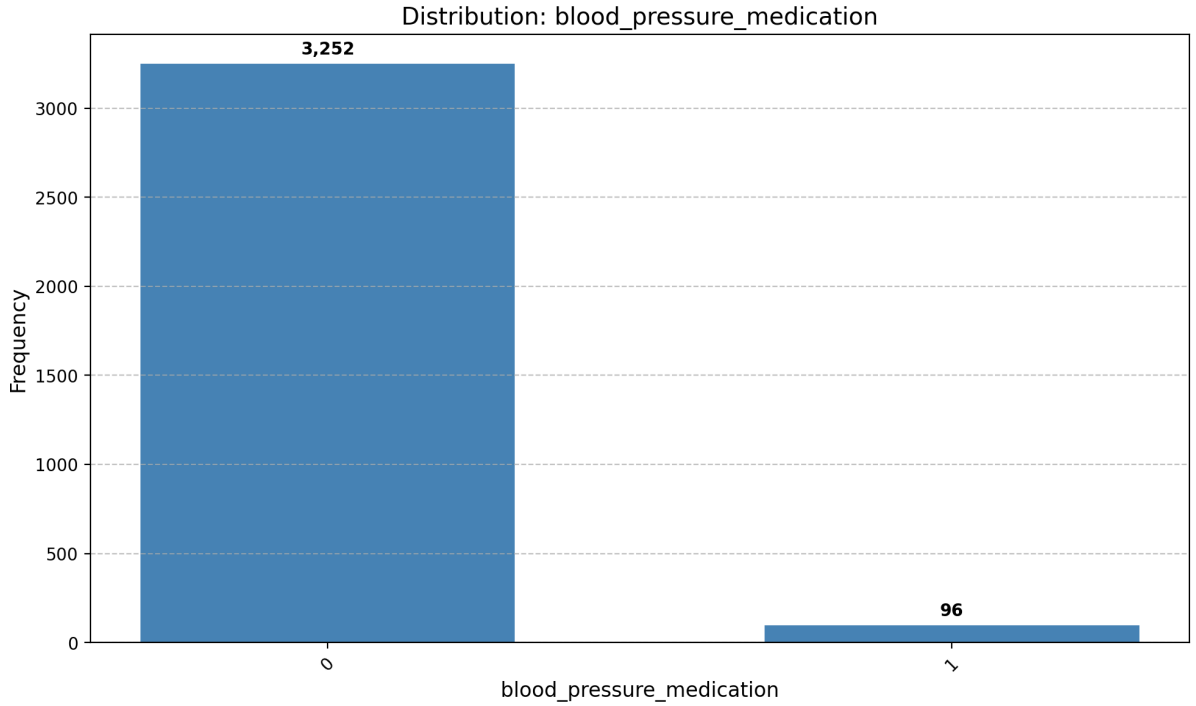


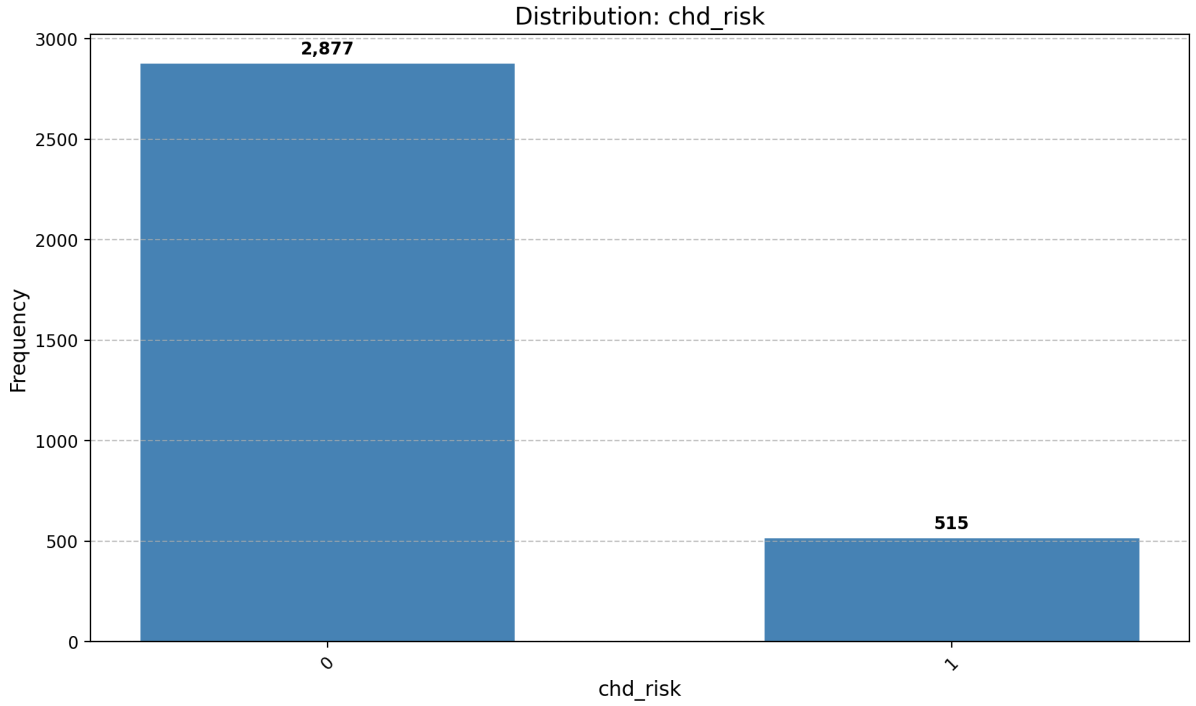
Video Count

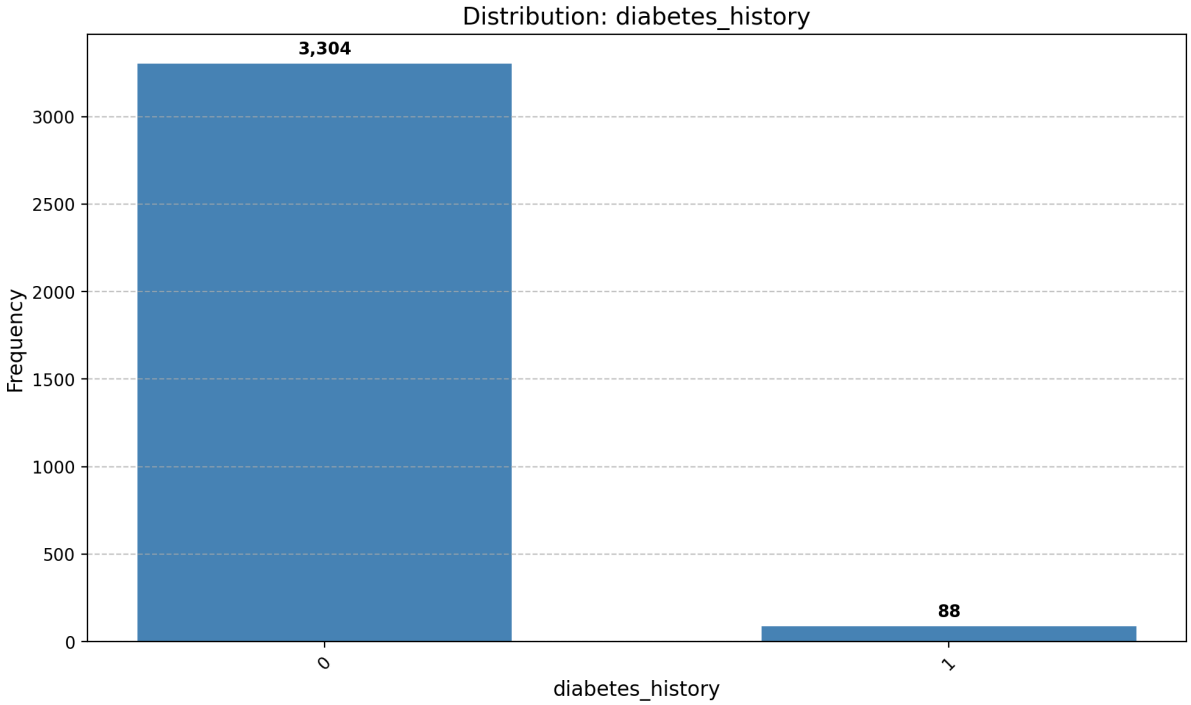
- Text dominance - with mean of only 1.25 videos per article and most concentrated at 0-1, content is predominantly textual

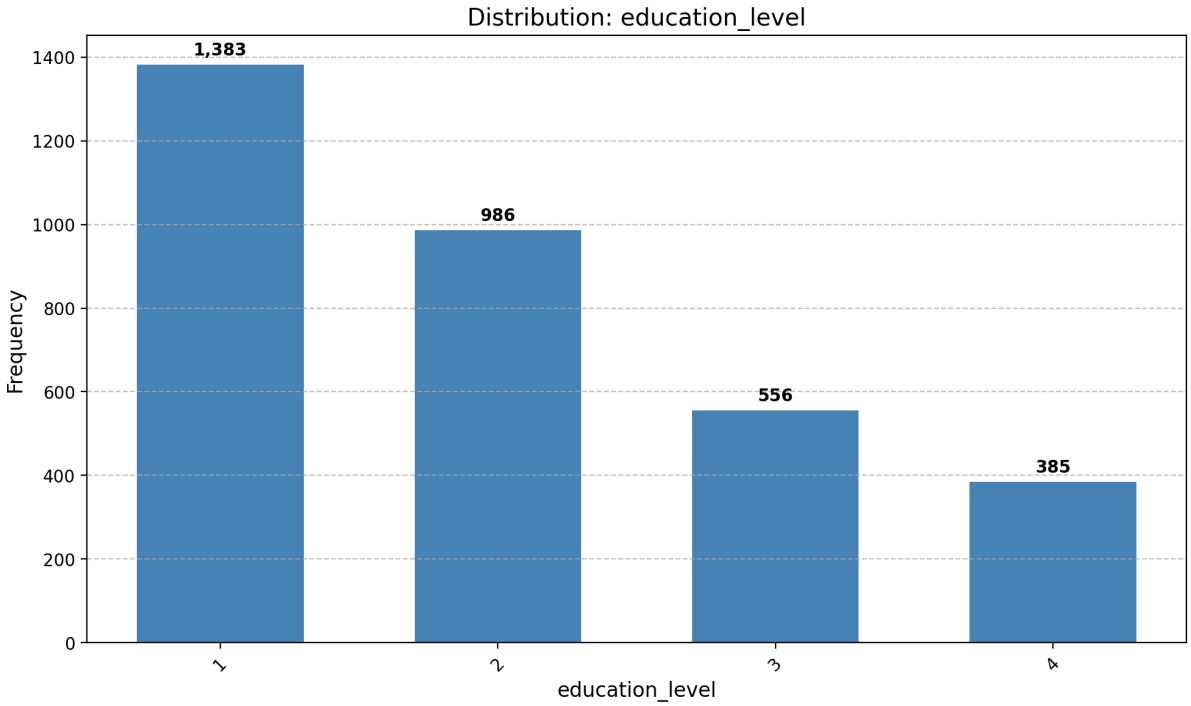
The function used for categorical data analysis is analyze_cat_data, which calculates basic statistics for each discrete attribute: valid (non-null) values, distinct values in a column, and generates a histogram for each attribute comparing these two metrics.

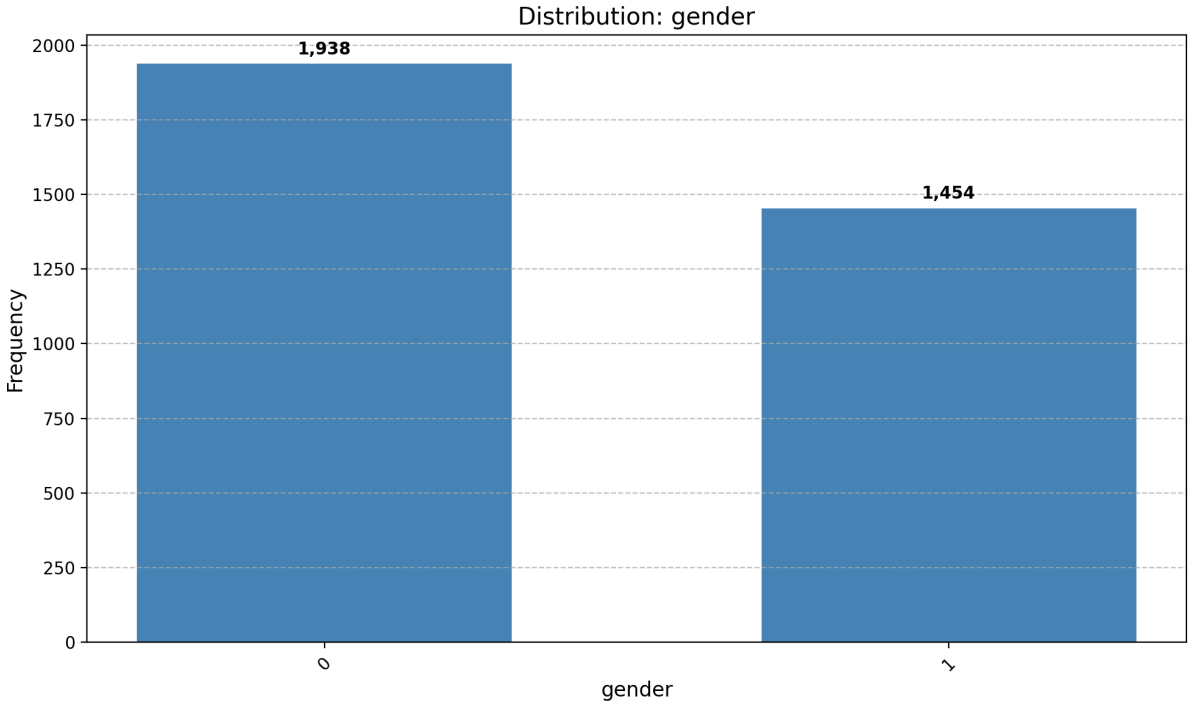
Graphs for Discrete Attribute Analysis - Coronary Heart Disease Risk:

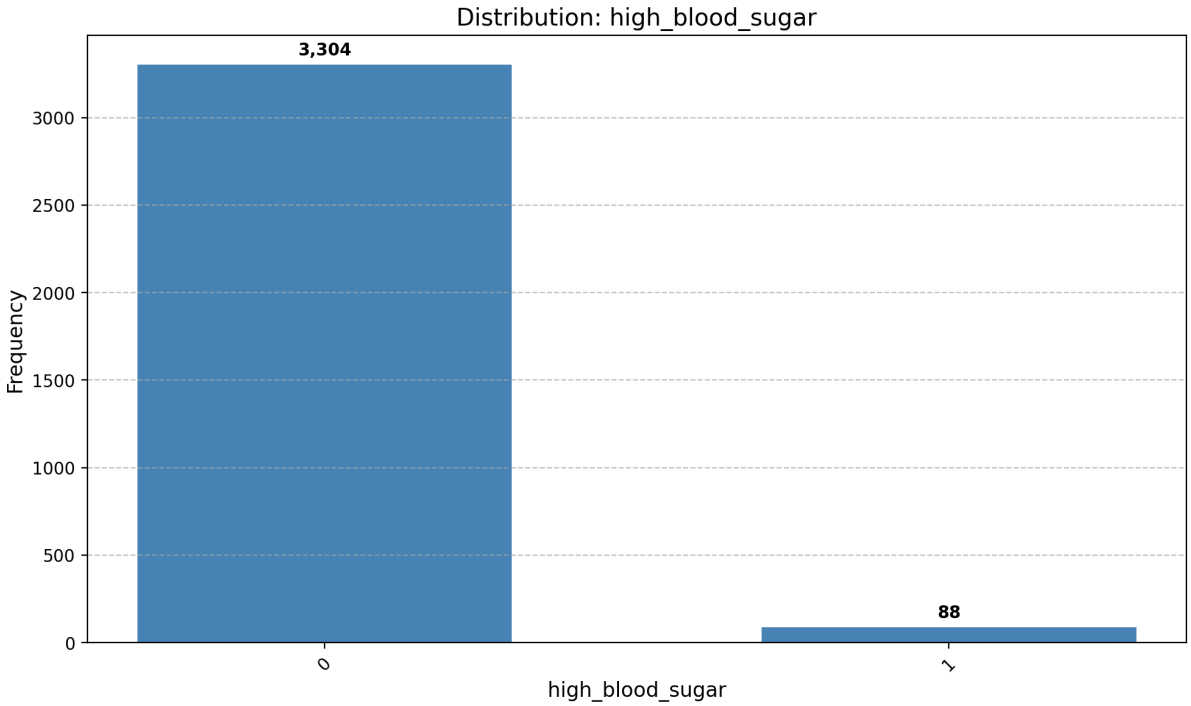


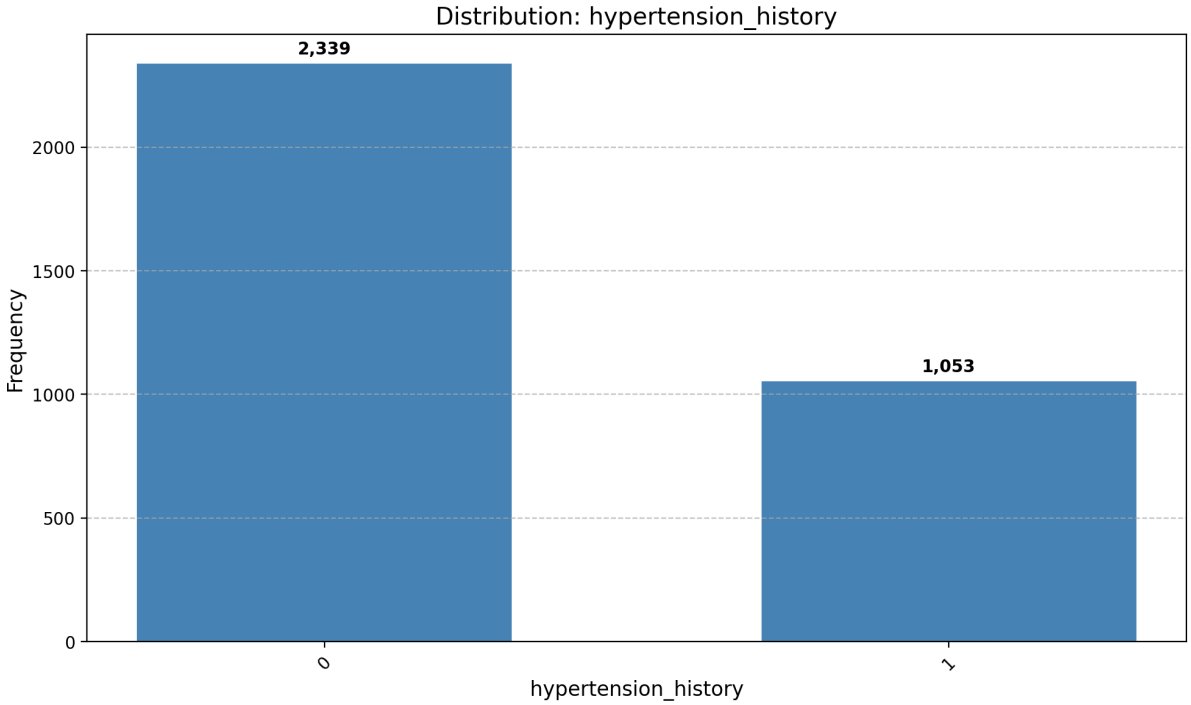


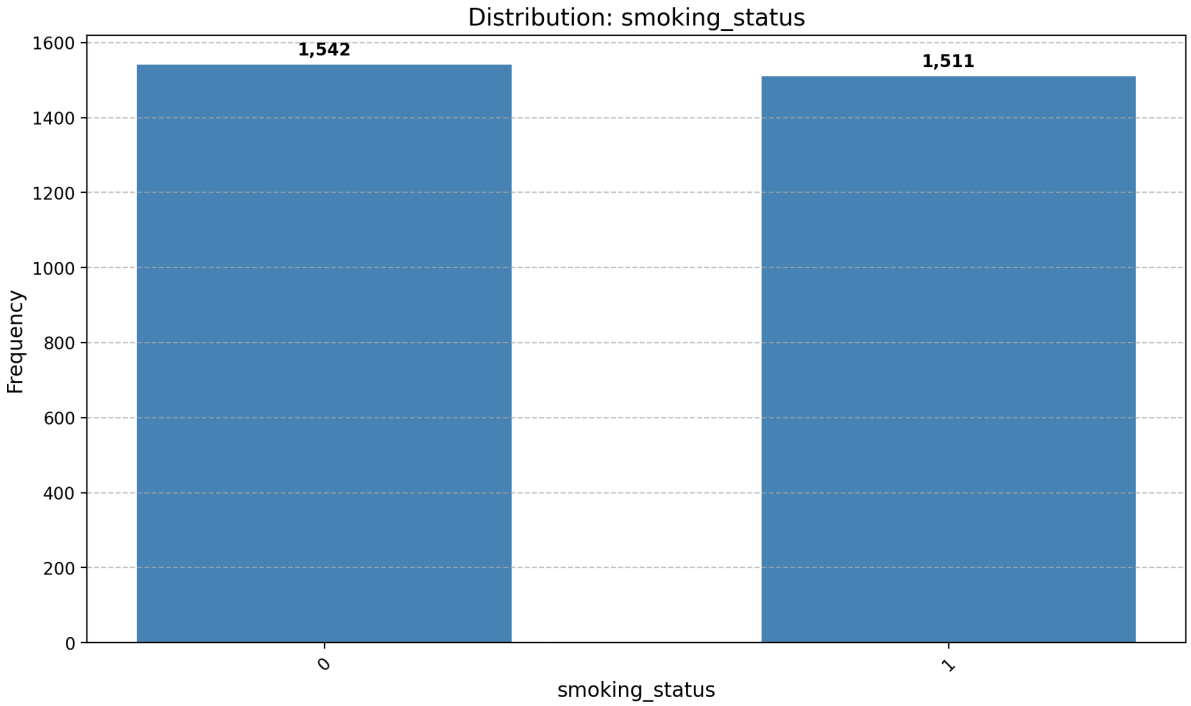

Distribution: diabetes_history
3,304
88
Frequency
0
500
1000
1500
2000
2500
3000
0
1
diabetes_history

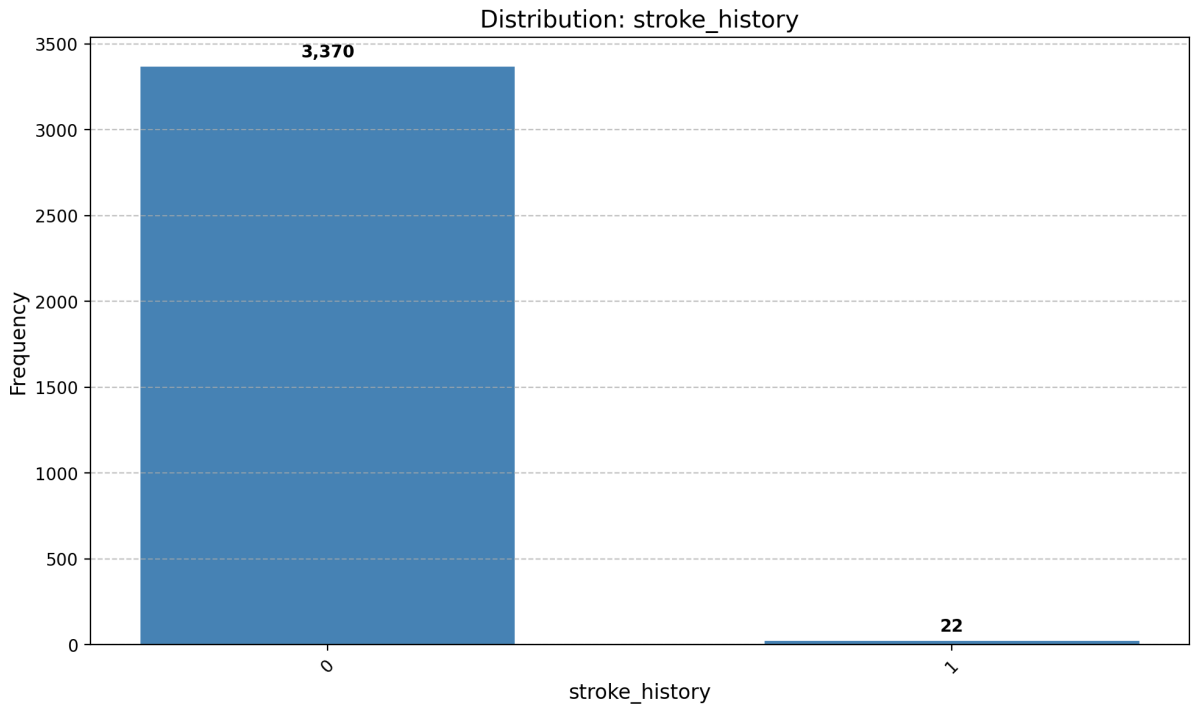

Distribution: education_level
1,383
986
556
385
Frequency
0
200
400
600
800
1000
1200
1400
1
2
3
4
education_level


Distribution: gender
Frequency
2000
1750
1500
1250
1000
750
500
250
0
1,938
1,454
0
1
gender


Distribution: high_blood_sugar
Frequency
3000
2500
2000
1500
1000
500
0
3,304
88
0
1
high_blood_sugar


Distribution: hypertension_history
2,339
1,053
Frequency
2000
1500
1000
500
0
0
1
hypertension_history


Distribution: smoking_status
1,542
1,511
Frequency
1600
1400
1200
1000
800
600
400
200
0
0
1
smoking_status



Complete Attributes (3,392 values):

- chd_risk, diabetes_history, gender, high_blood_sugar, hypertension_history, stroke_history

Attributes with Missing Data:

- smoking_status: 3,053 values (missing ~339, ~10.0%)
- education_level: 3,310 values (missing ~82, ~2.4%)
- blood_pressure_medication: 3,348 values (missing ~44, ~1.3%)

Binary Attributes (2 categories - 0/1):

- blood_pressure_medication, chd_risk, diabetes_history, gender, high_blood_sugar, hypertension_history, smoking_status, stroke_history

Exception - Multinomial Attribute:

- education_level: 4 distinct categories (1, 2, 3, 4 - ordered educational levels)

Categorical Value Distribution

Extreme Imbalances (>95% in one category):

- stroke_history: 99.4% = 0 (3,370 vs 22)
- diabetes_history: 97.4% = 0 (3,304 vs 88)
- high_blood_sugar: 97.4% = 0 (3,304 vs 88)
- blood_pressure_medication: 97.1% = 0 (3,252 vs 96)

Balanced Distributions:

- smoking_status: 50.5% = 0, 49.5% = 1 (1,542 vs 1,511)
- gender: 57.1% = 0, 42.9% = 1 (1,938 vs 1,454)

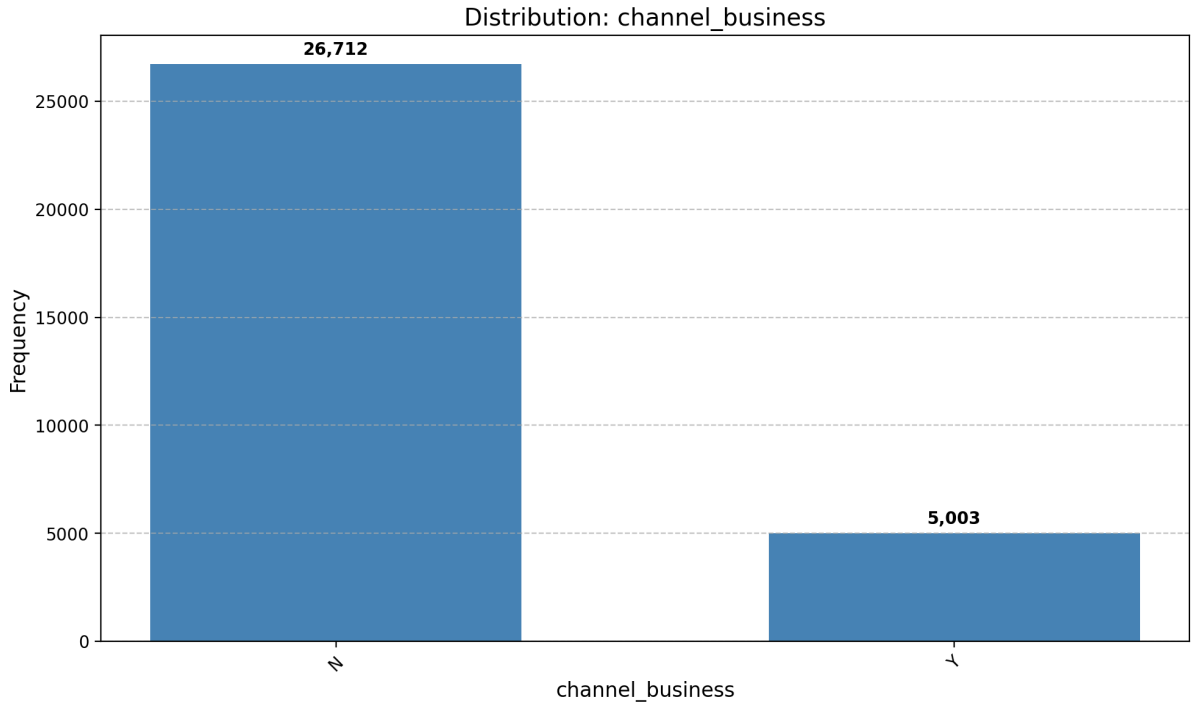
Moderate Distributions:

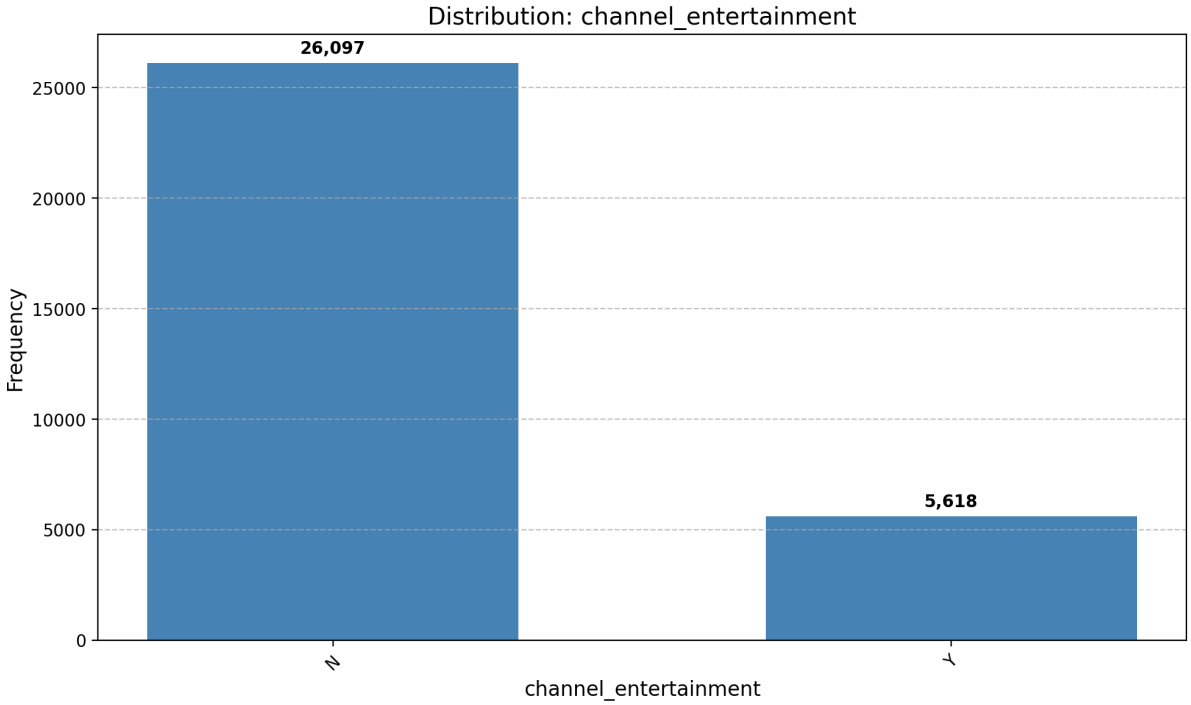
- chd_risk (target): 84.8% = 0, 15.2% = 1 (2,877 vs 515)
- hypertension_history: 69.0% = 0, 31.0% = 1 (2,339 vs 1,053)

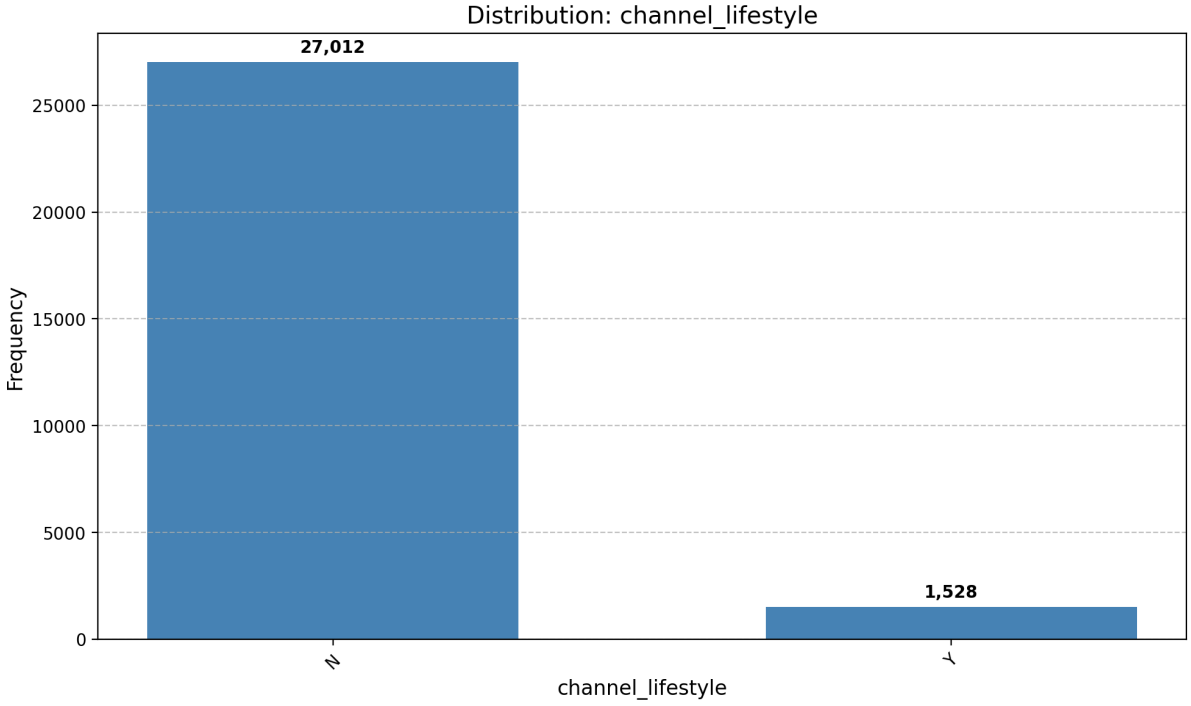
Multinomial Distribution:

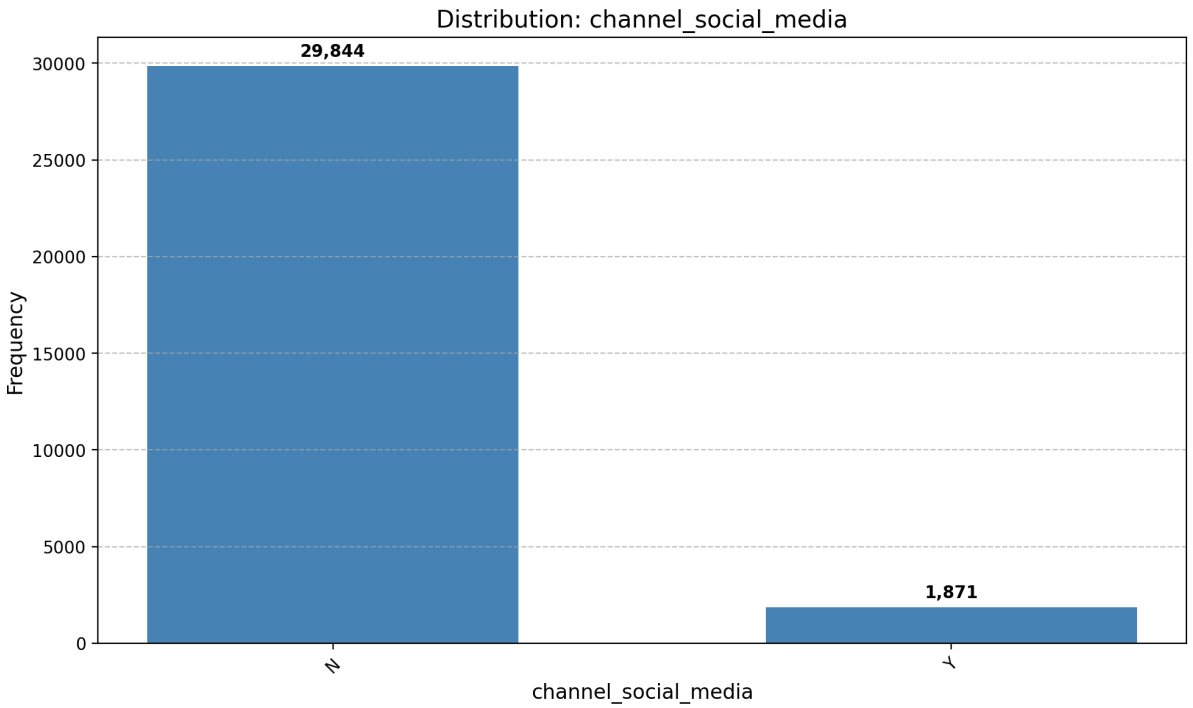
- education_level:
    - Level 1: 41.8% (1,383)
    - Level 2: 29.8% (986)
    - Level 3: 16.8% (556)
    - Level 4: 11.6% (385)

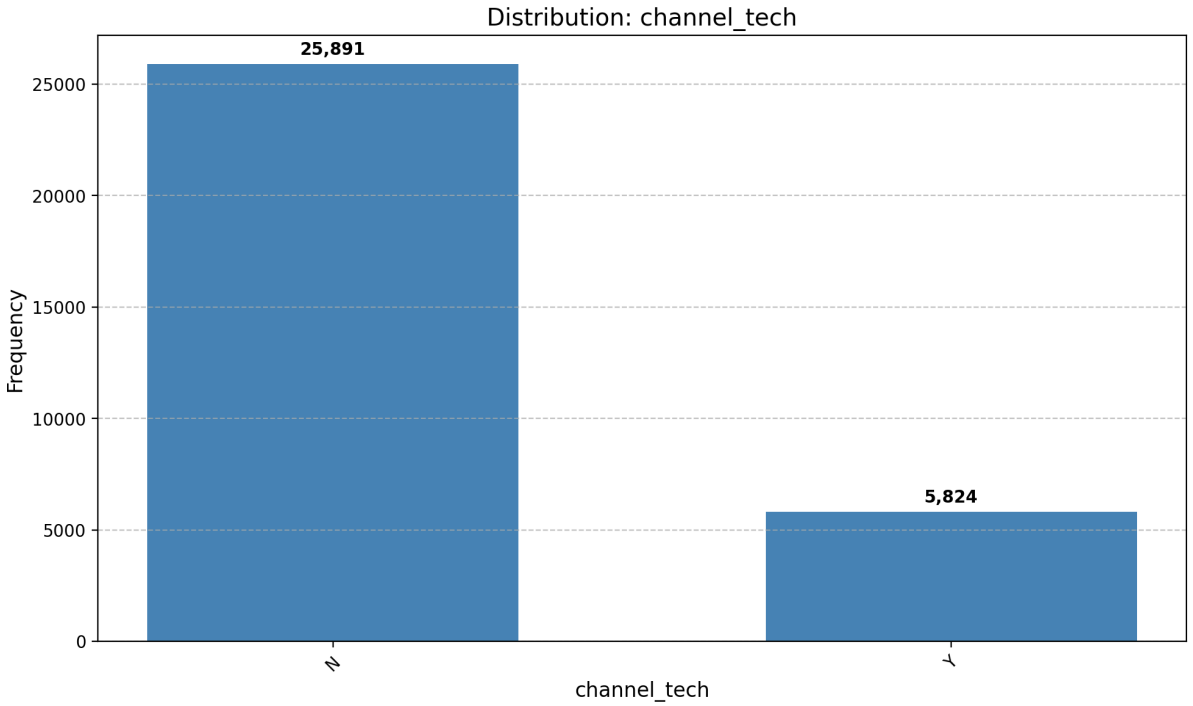
Graphs for Discrete Attribute Analysis - Online News Popularity:

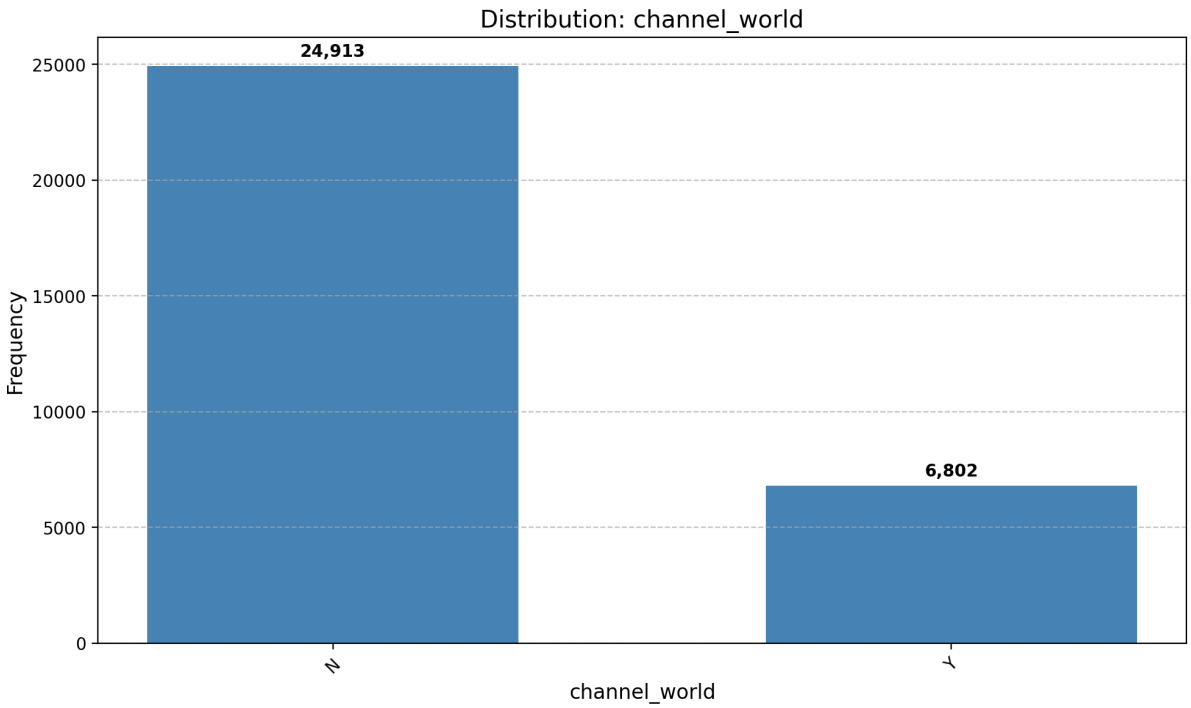


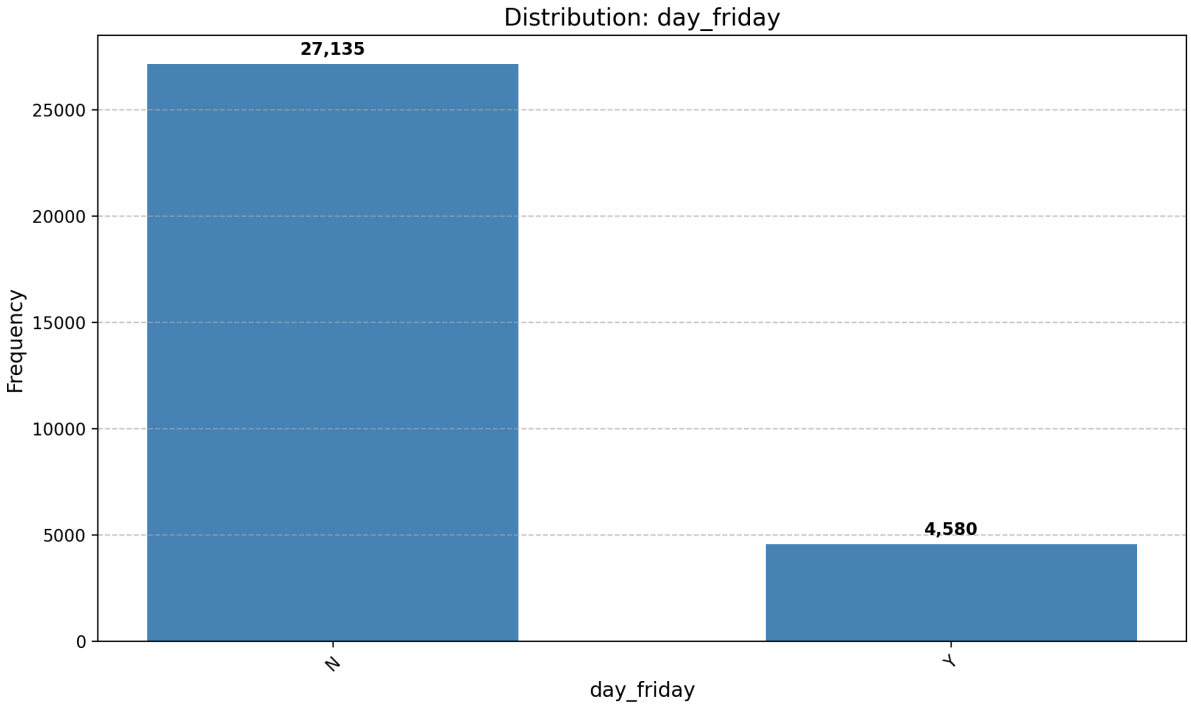

Distribution: channel_entertainment
26,097
5,618
25000
20000
15000
10000
5000
0
Frequency
N
Y
channel_entertainment

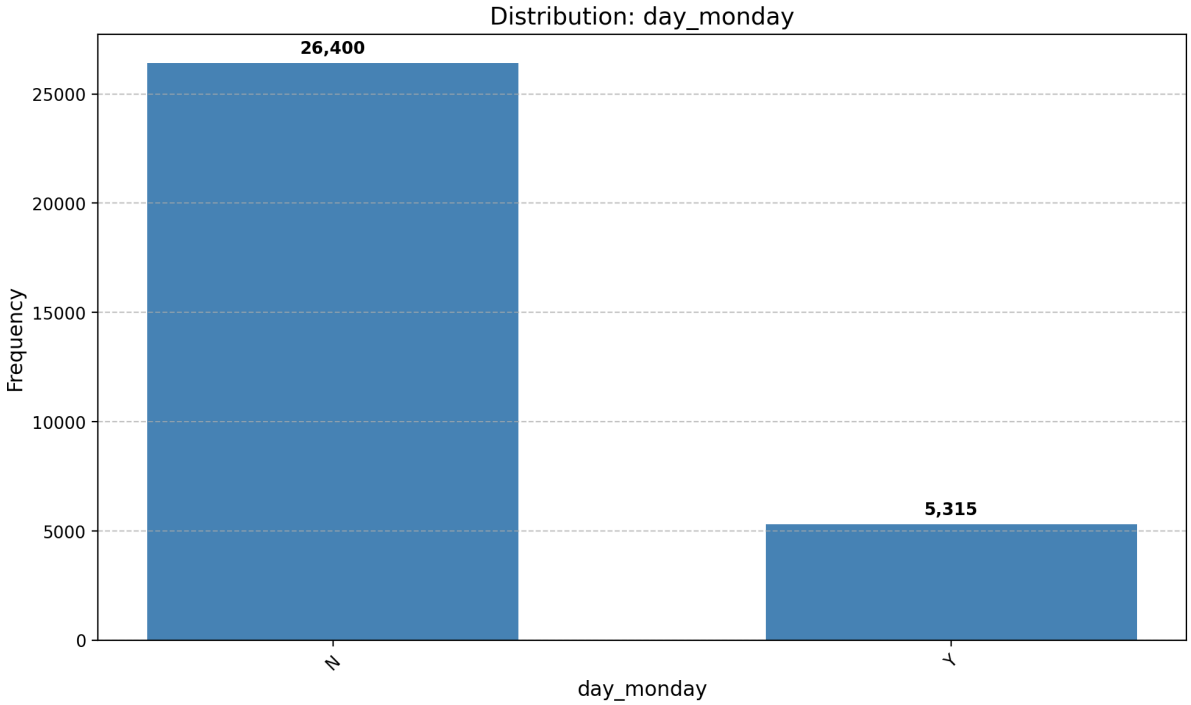

Distribution: channel_lifestyle
27,012
1,528
25000
20000
15000
10000
5000
0
Frequency
N
Y
channel_lifestyle

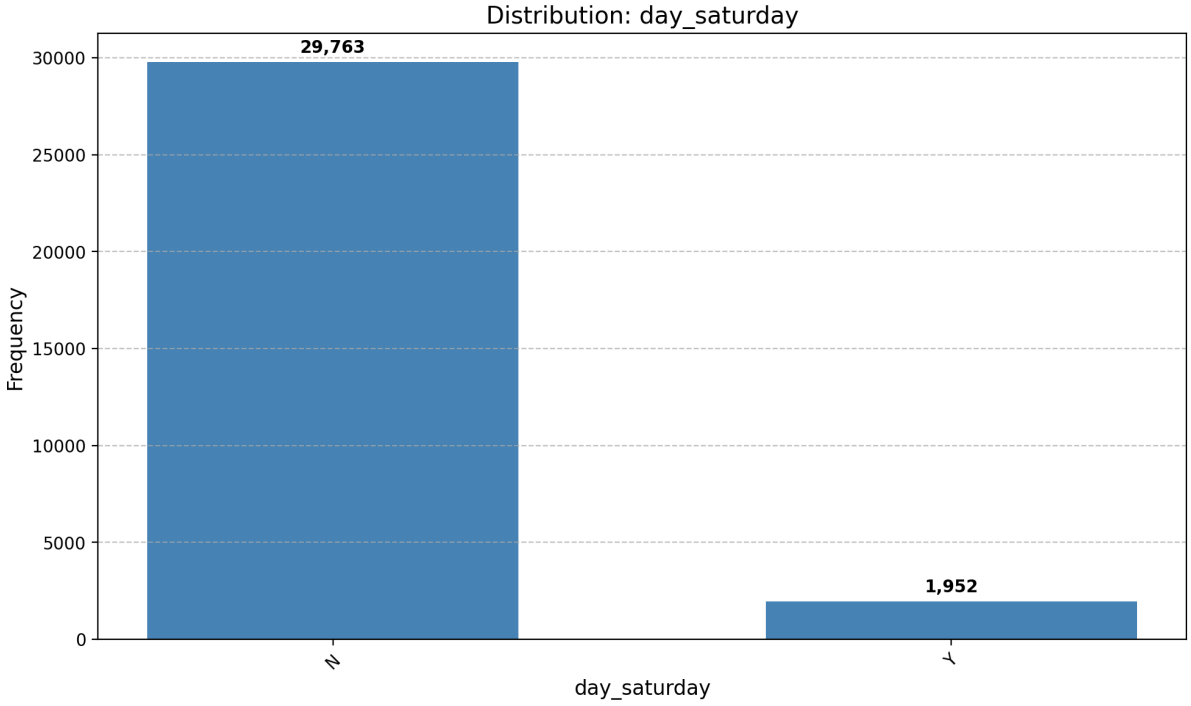

Distribution: channel_social_media
29,844
1,871
30000
25000
20000
15000
10000
5000
0
Frequency
N
Y
channel_social_media

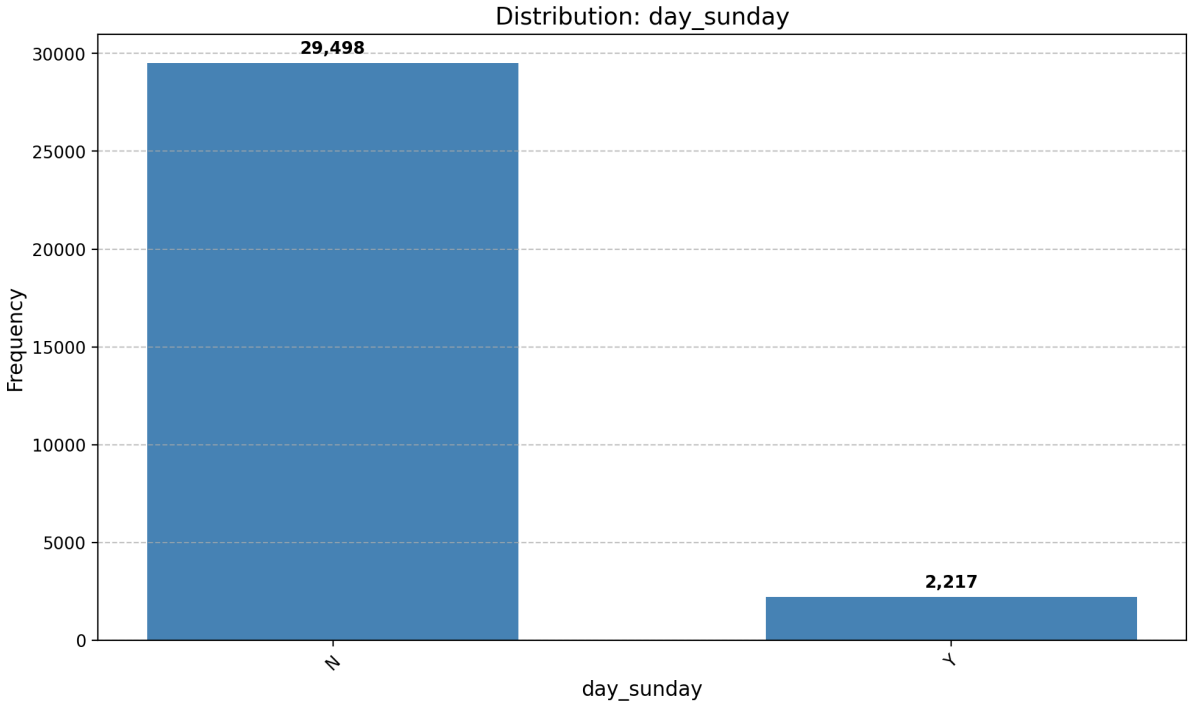

Distribution: channel_tech
25,891
5,824
25000
20000
15000
10000
5000
0
Frequency
N
Y
channel_tech

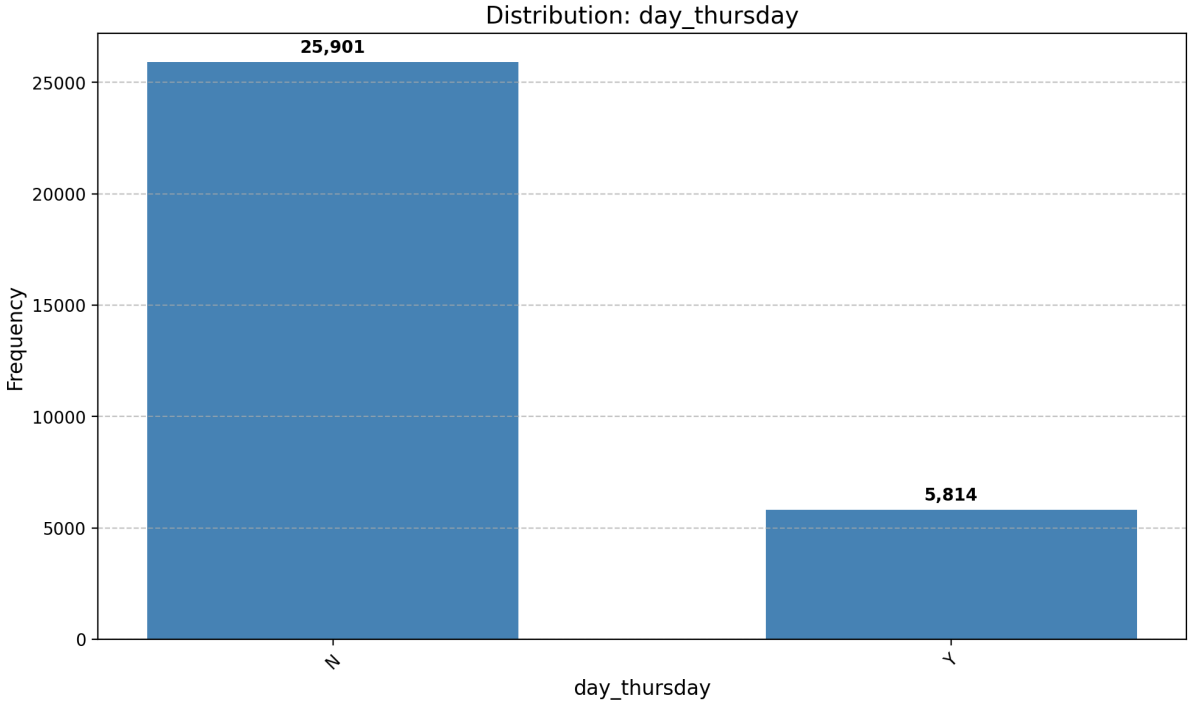

Distribution: channel_world
24,913
6,802
Frequency
25000
20000
15000
10000
5000
0
N
Y
channel_world

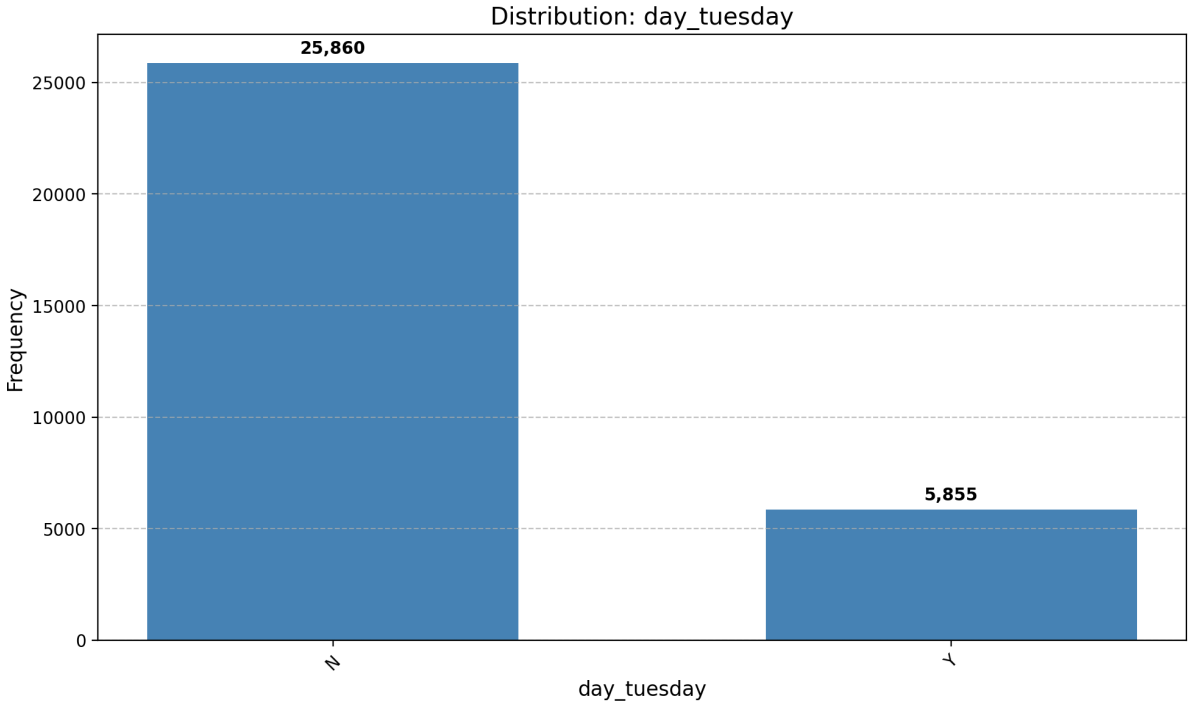

Distribution: day_friday
27,135
4,580
Frequency
25000
20000
15000
10000
5000
0
N
Y
day_friday

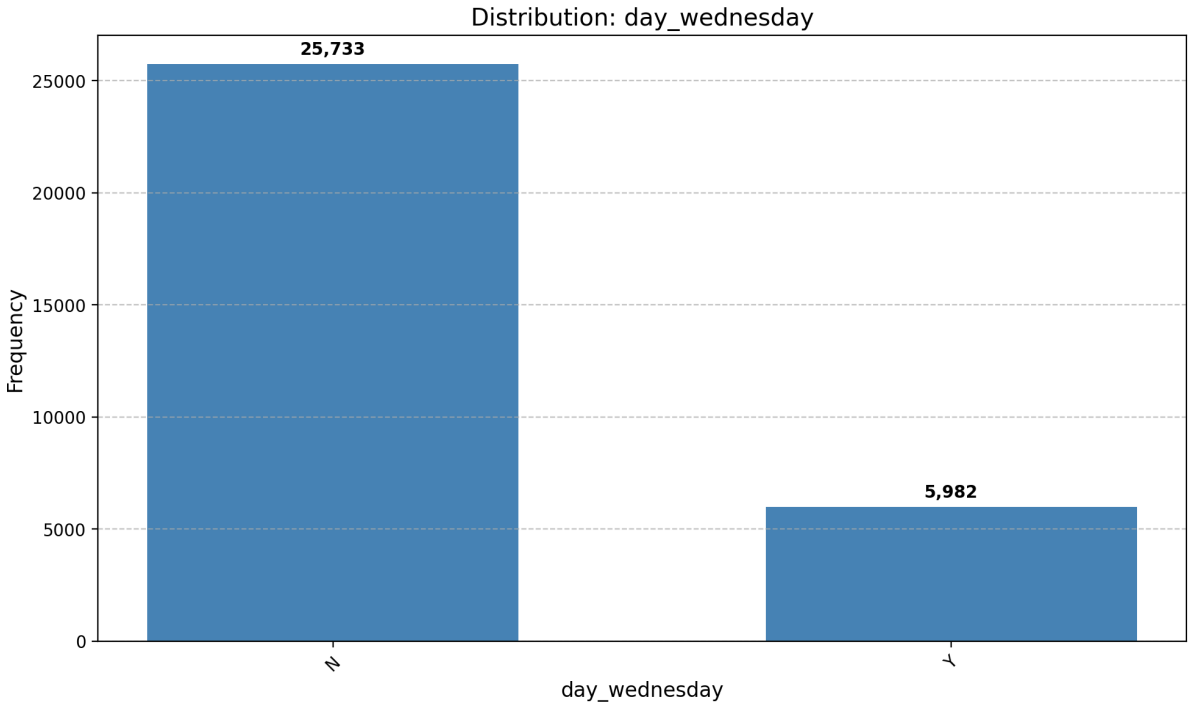

Distribution: day_monday
26,400
5,315
25000
20000
15000
10000
5000
0
Frequency
N
Y
day_monday

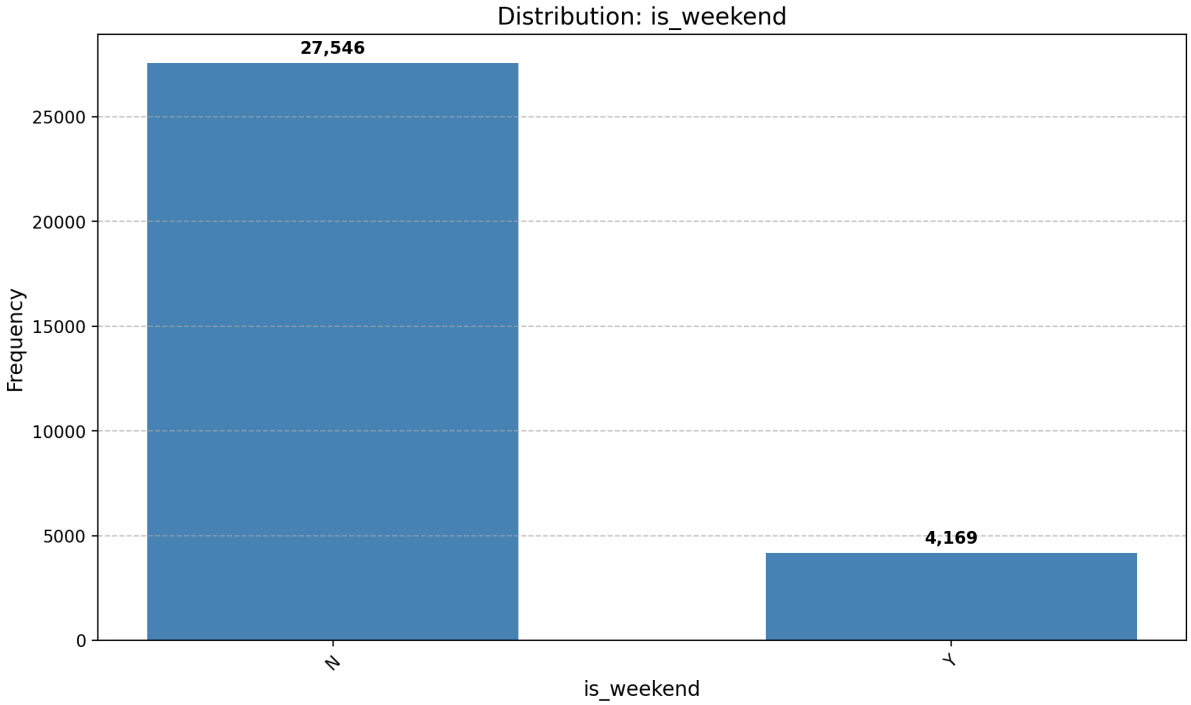

Distribution: day_saturday
29,763
1,952
30000
25000
20000
15000
10000
5000
0
Frequency
N
Y
day_saturday

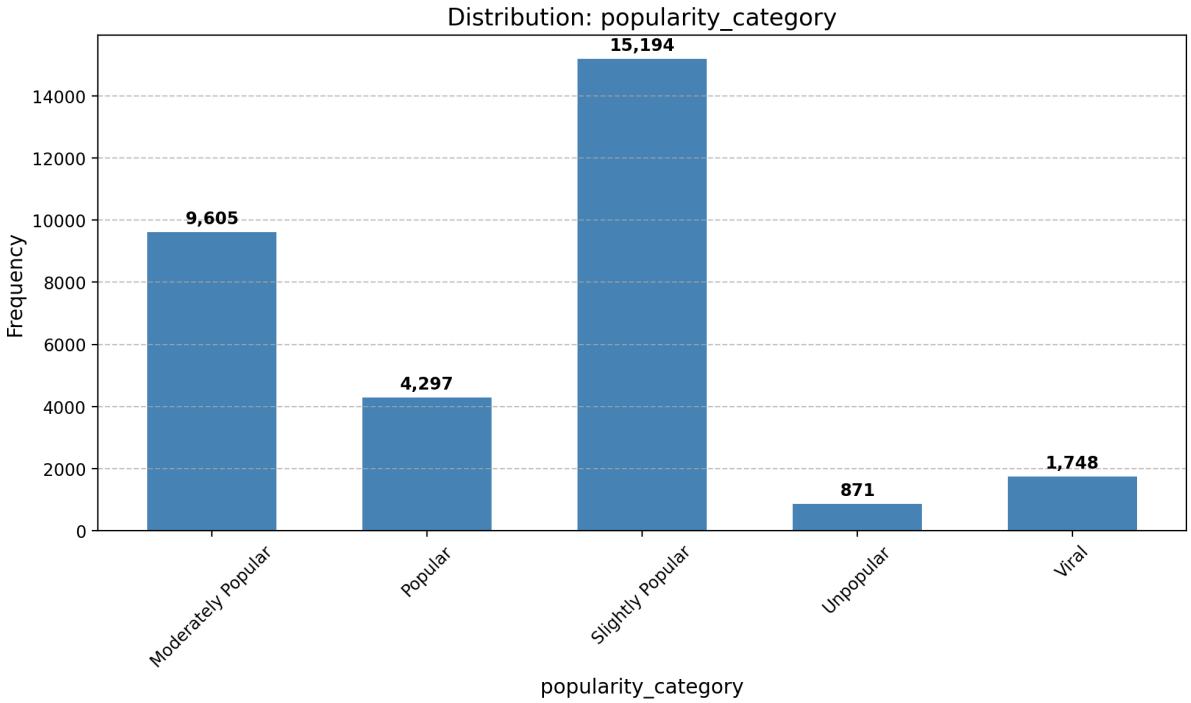

Distribution: day_sunday
Frequency
30000
25000
20000
15000
10000
5000
0
29,498
2,217
N
Y
day_sunday

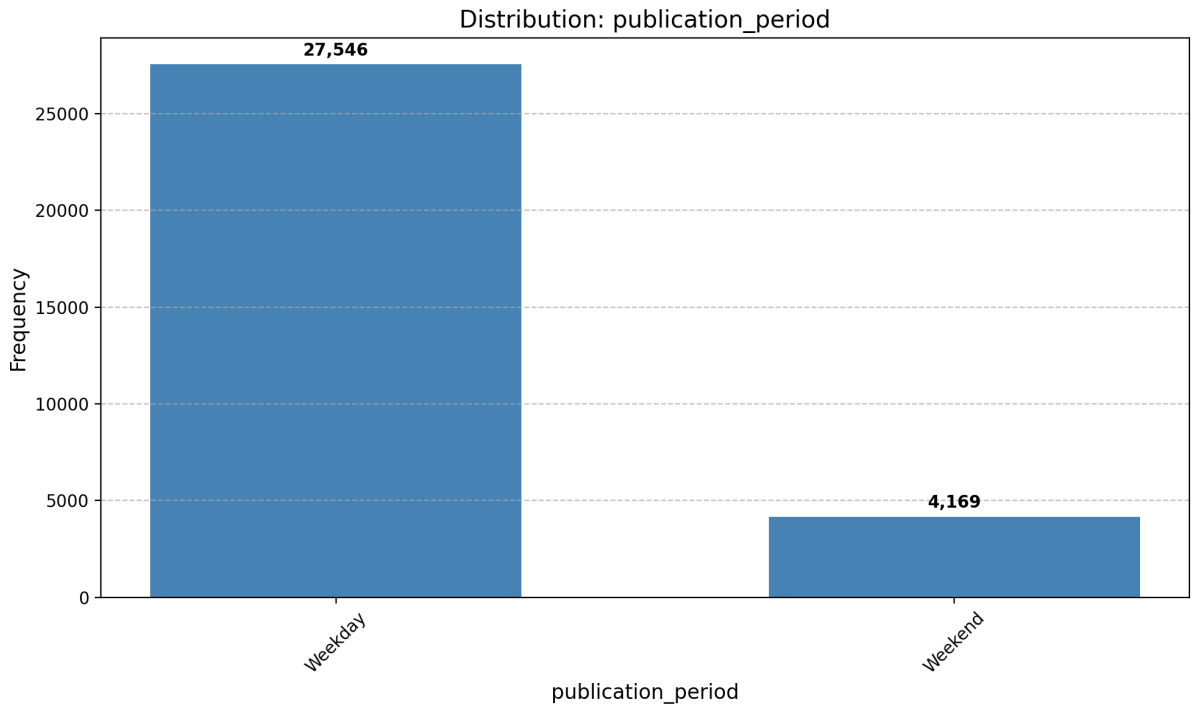

Distribution: day_thursday
Frequency
25000
20000
15000
10000
5000
0
25,901
5,814
N
Y
day_thursday


Distribution: day_tuesday
25,860
5,855
Frequency
0
5000
10000
15000
20000
25000
N
Y
day_tuesday


Distribution: day_wednesday
25,733
5,982
Frequency
0
5000
10000
15000
20000
25000
N
Y
day_wednesday


Distribution: is_weekend
27,546
4,169
25000
20000
15000
10000
5000
0
Frequency
N
Y
is_weekend


Distribution: popularity_category
15,194
9,605
4,297
1,748
871
14000
12000
10000
8000
6000
4000
2000
0
Frequency
Moderately Popular
Popular
Slightly Popular
Unpopular
Viral
popularity_category



Categorical Value Distribution

Content Channels:

- channel_world: 78.5% = 0, 21.5% = 1 (most balanced)
- channel_tech: 81.6% = 0, 18.4% = 1 (imbalanced)
- channel_entertainment: 82.3% = 0, 17.7% = 1 (imbalanced)
- channel_business: 84.2% = 0, 15.8% = 1 (imbalanced)
- channel_social_media: 94.1% = 0, 5.9% = 1 (extremely imbalanced)
- channel_lifestyle: 94.6% = 0, 4.8% = 1 (extremely imbalanced)

Days of the Week:

- day_wednesday: 81.1% = 0, 18.9% = 1 (most balanced among days)
- day_tuesday: 81.5% = 0, 18.5% = 1 (more balanced)
- day_thursday: 81.7% = 0, 18.3% = 1 (more balanced)
- day_monday: 83.2% = 0, 16.8% = 1 (imbalanced)
- day_friday: 85.6% = 0, 14.4% = 1 (imbalanced)
- day_saturday: 93.8% = 0, 6.2% = 1 (extremely imbalanced)
- day_sunday: 93.0% = 0, 7.0% = 1 (extremely imbalanced)

Derived Variables:

- is_weekend = publication_period: 86.9% = 0, 13.1% = 1 (identical, indicating data redundancy)

Popularity Categories (Multinomial):

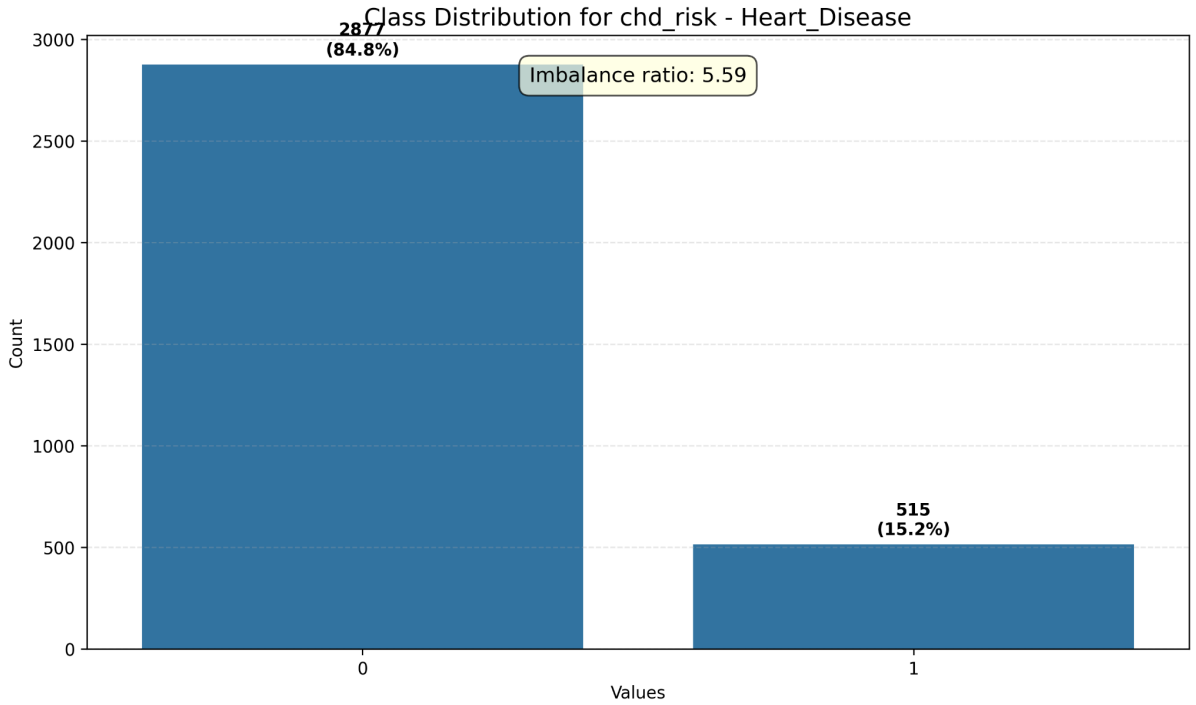
- Slightly Popular: 47.9% (15,194) - Majority

- Moderately Popular: 30.3% (9,605)
- Popular: 13.5% (4,297)
- Viral: 5.5% (1,748)
- Unpopular: 2.7% (871)

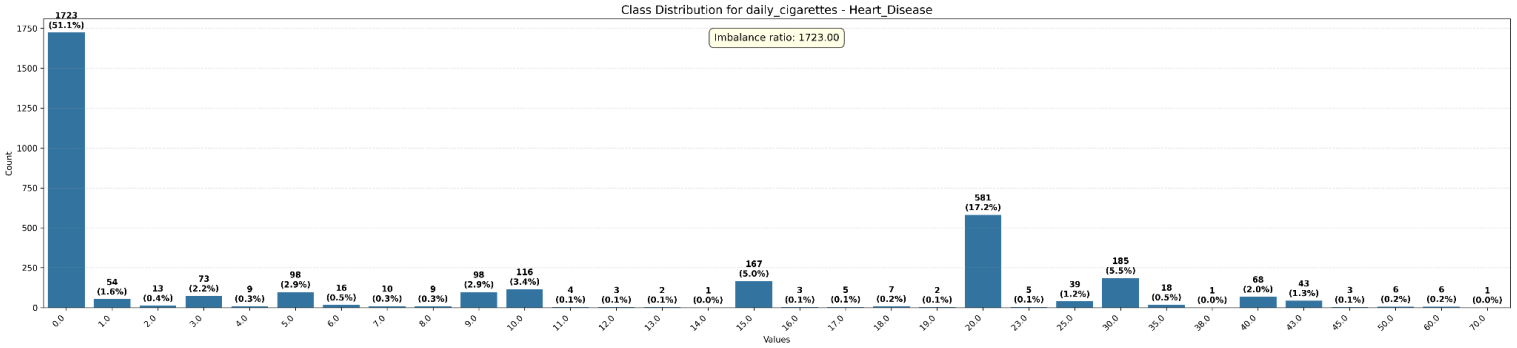
Class Balance Analysis

Class balance analysis was performed using the analyze_class_balance function, which accepts specific columns as input since the news popularity dataset is too large and the program freezes if all attributes are processed simultaneously. The function standardizes input (string or list of columns), calculates statistics for each column and imbalance ratio, and displays results using plots.

Balance Graphs for Coronary Heart Disease Risk:

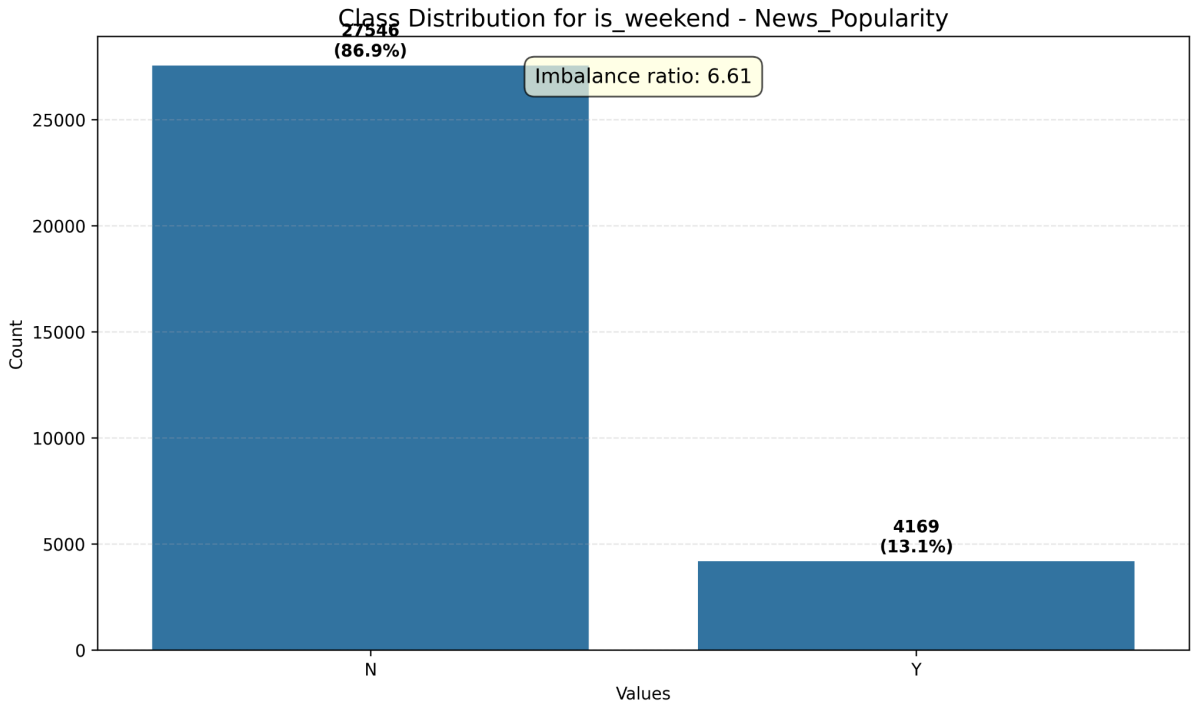


This binary variable shows moderate imbalance with 84.8% of patients classified as low risk (0) and only 15.2% with high risk (1). The imbalance ratio of 5.59 indicates the majority class is nearly 6 times larger than the minority class.

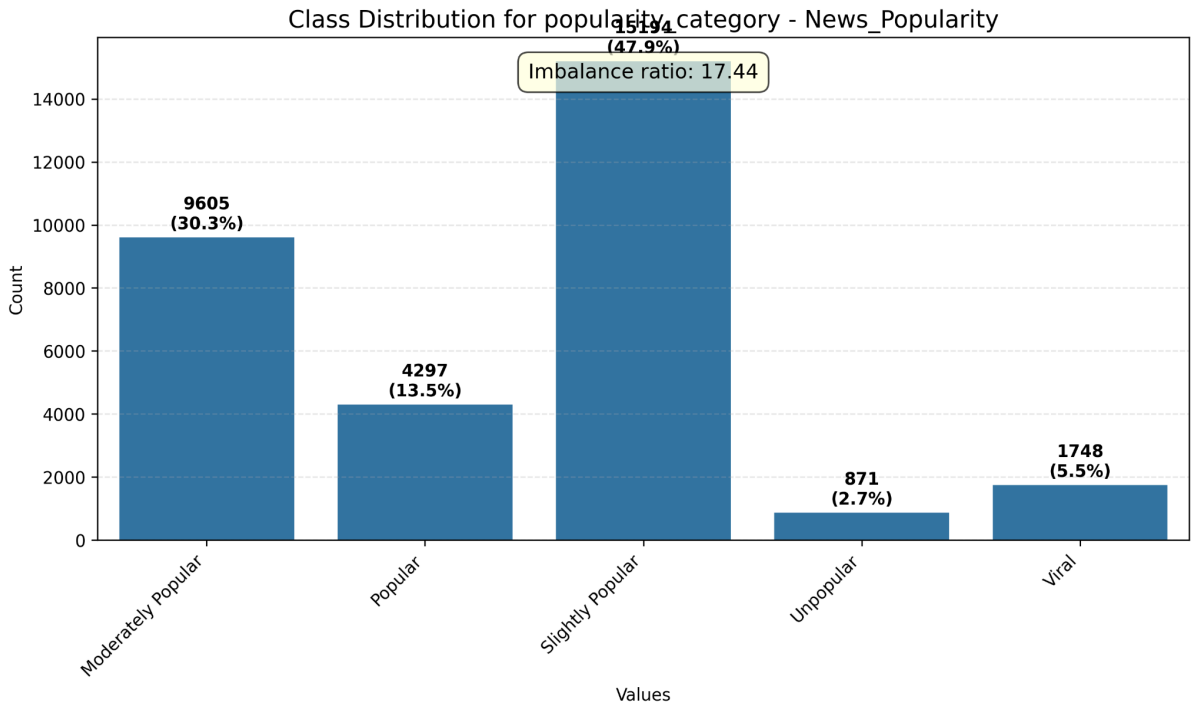


The remaining attributes are relatively imbalanced due to their diversity, with imbalance ratios ranging from 1.00 (glucose) to 1723.00 (daily_cigarettes).

Balance Graphs for Online News Popularity:



The graph reveals extremely imbalanced distribution between articles published during the week and those published on weekends. With 86.9% of articles published on weekdays (N) and only 13.1% on weekends (Y), an imbalance ratio of 6.61 is observed, suggesting journalistic activity is predominantly concentrated during the week.

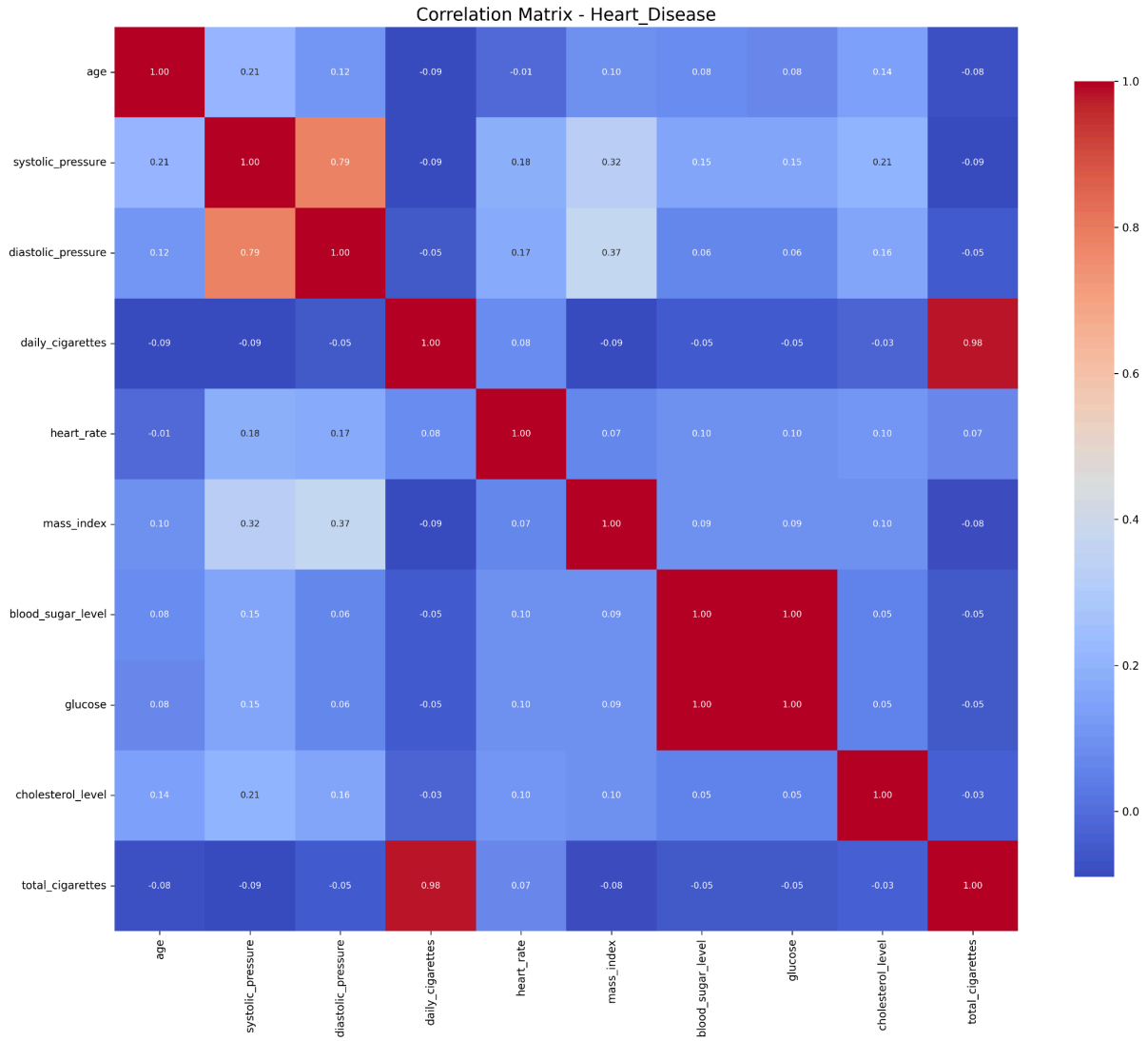


The popularity category distribution shows the most pronounced imbalance with a ratio of 17.44. The "Slightly Popular" category dominates with 47.9% of articles, followed by "Moderately Popular" at 30.3%. Extreme categories are much rarer: "Popular" at 13.5%, "Viral" at only 5.5%, and "Unpopular" is the smallest at 2.7%.

Correlation Analysis for Redundant Attribute Elimination

Correlation analysis was performed using analyze_numerical_correlation and analyze_categorical_correlation functions.

The analyze_numerical_correlation function analyzes Pearson correlations between numerical attributes in a dataset and identifies attributes to eliminate due to high correlation. The function calculates the correlation matrix for all numerical attributes, then creates a plot saved as a PNG image. For each pair of attributes with correlation greater than 0.8 in absolute value, the function decides to eliminate the attribute with more missing values, while always preserving the target attribute.

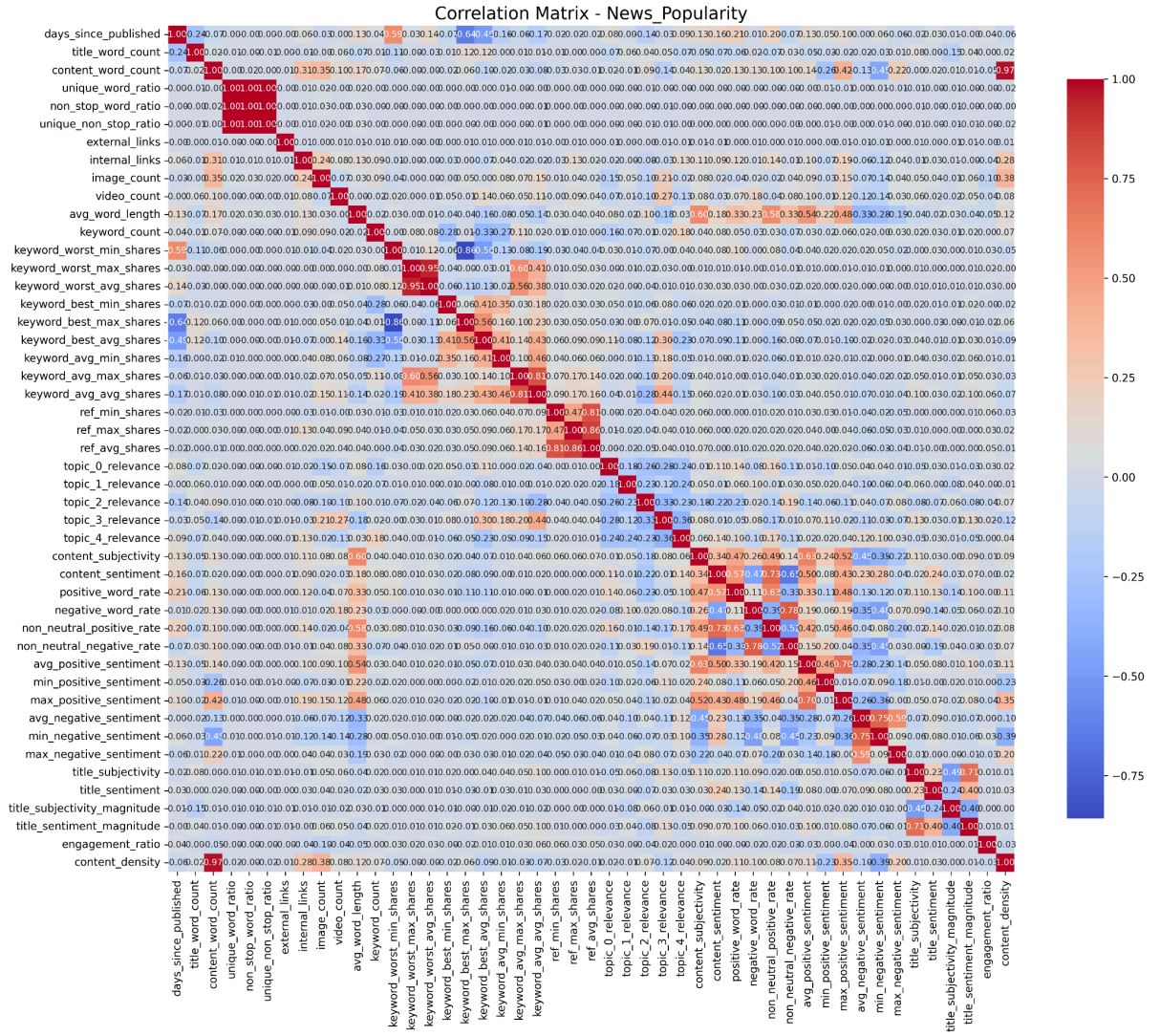


Heart Disease Dataset - Highly Correlated Pairs:

1. (total_cigarettes, daily_cigarettes)
2. (high_blood_sugar, diabetes_history)

3. (glucose, blood_sugar_level)

It's evident how one attribute influences its paired attribute. Therefore, one attribute is considered redundant and will be eliminated.



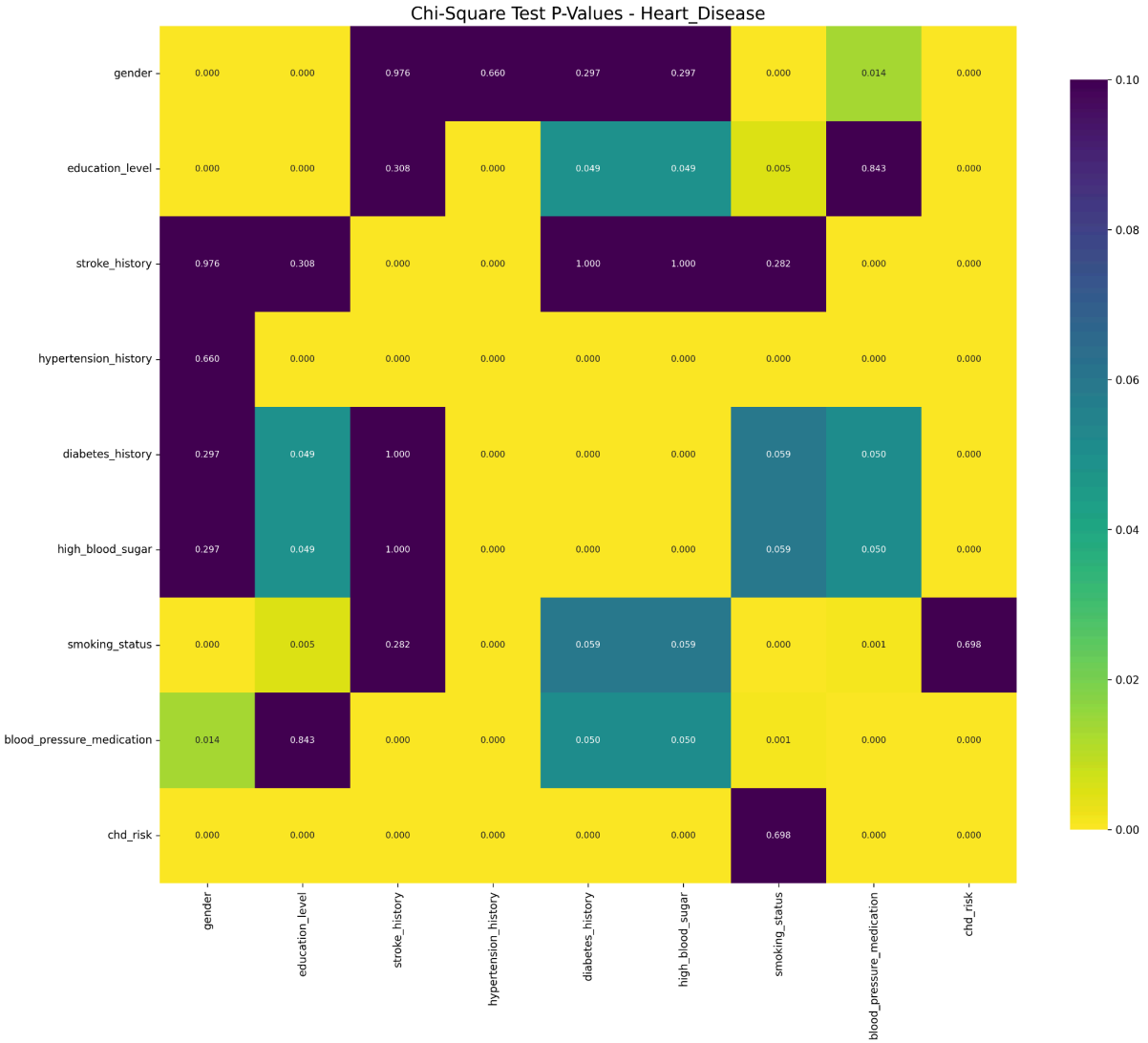
News Popularity Dataset - Highly Correlated Pairs:

1. (non_stop_word_ratio, unique_word_ratio)
2. (unique_non_stop_ratio, unique_word_ratio)
3. (unique_non_stop_ratio, non_stop_word_ratio)
4. (keyword_worst_avg_shares, keyword_worst_max_shares)
5. (keyword_best_max_shares, keyword_worst_min_shares)
6. (keyword_avg_avg_shares, keyword_avg_max_shares)
7. (ref_avg_shares, ref_min_shares)
8. (ref_avg_shares, ref_max_shares)

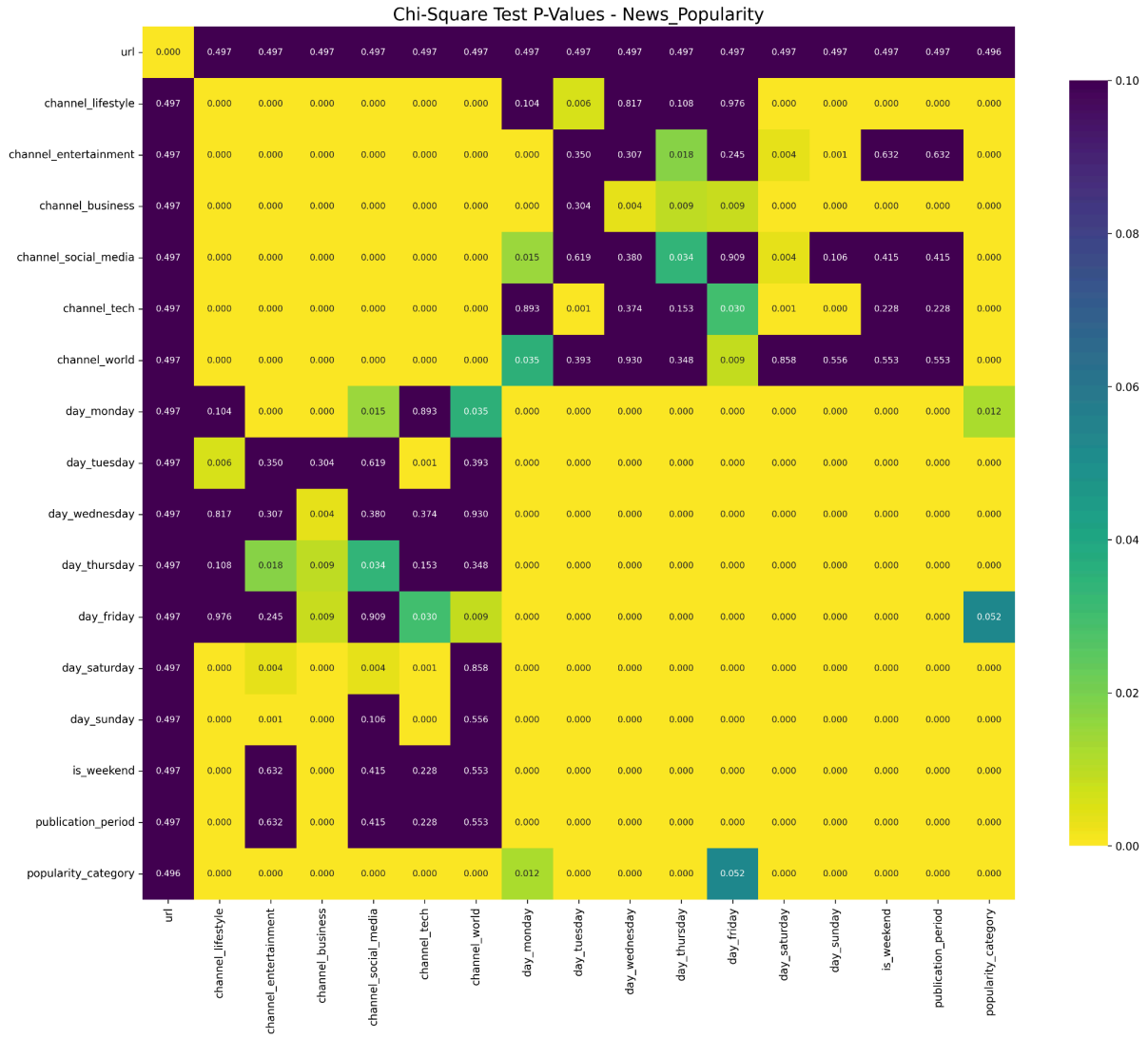
9. (content_density, content_word_count)

One attribute from each pair will be eliminated.

The analyze_categorical_correlation function evaluates statistical association between categorical attributes using the Chi-Square test and identifies correlated attributes for elimination. The function constructs a p-value matrix by calculating the Chi-Square test for each pair of categorical attributes, then creates a visualization of these p-values. Pairs of attributes with p-values ≤ 0.05 are considered significantly correlated, and from these pairs, the function eliminates the attribute with more missing values while always preserving the target attribute.



Heart Disease Dataset: Variables with significant association with chd_risk ($p < 0.05$): gender, education_level, stroke_history, hypertension_history, diabetes_history, high_blood_sugar, blood_pressure_medication.



News Popularity Dataset - Correlated Pairs:

1. (day_monday, channel_tech)
2. (day_wednesday, channel_lifestyle)
3. (day_wednesday, channel_world)
4. (day_friday, channel_lifestyle)
5. (day_friday, channel_social_media)
6. (day_saturday, channel_world)

In these matrices, uncorrelated pairs are shown in yellow, while strongly correlated pairs are shown in purple.

2. Data Preprocessing (`data_preprocessing.py`)

1. Missing Data Imputation

Functions used: process_news_popularity_missing_data and process_heart_disease_missing_data.

The purpose is to check each dataset for missing values. If they exist, data is split into numerical and categorical, imputation is performed for each data type, then data is combined. Otherwise, data is returned directly.

For imputation, the following functions are used:

- impute_categorical_data: Receives initial data and categorical column list, uses SimpleImputer(strategy='most_frequent') to replace missing values with the most frequent value in each column, returns a DataFrame with completed categorical data.
- impute_numerical_data: Receives initial data and numerical columns, uses IterativeImputer (an advanced algorithm that imputes each column based on others, iterating up to max_iter=300 times to improve estimates), returns a DataFrame with completed numerical data.

The function combining numerical and categorical imputed data is combine_imputed_data.

Imputed data is stored for each dataset at the 'heart_train_filled' key in heart_set and 'news_train_filled' in news_set.

2. Outlier Treatment

The `replace_outliers_data` function detects and replaces extreme values (outliers) in numerical attributes using the IQR (interquartile range) rule to identify these values. Instead of removing outliers, the function temporarily replaces them with NaN, then applies IterativeImputer to fill these values. Finally, imputed values are reintroduced only in positions where outliers were detected, thus preserving the original data structure and avoiding information loss.

3. Redundant Attribute Removal

The remove_redundant_news_attributes and remove_redundant_heart_attributes functions eliminate redundant columns from the news popularity and heart disease datasets. They receive an attributes_to_drop list with column names to be eliminated and a preprocessed numerical DataFrame (with treated outliers). If the list is empty, nothing is eliminated and a copy of the data is returned. If the list contains attributes, these are eliminated using drop(columns=...), and the function displays in the console how many and which columns were eliminated, as well as the DataFrame dimensions before and after cleaning.

Removing 8 redundant attributes from Heart Disease:

- diabetes_history
- education_level
- daily_cigarettes
- smoking_status
- stroke_history
- blood_pressure_medication
- blood_sugar_level
- hypertension_history

Heart Disease: (3392, 19) → (3392, 11)

Removing 22 redundant attributes from News Popularity:

News Popularity: (31715, 64) → (31715, 42)

4. Numerical Data Standardization

The standardization operation (bringing to the same scale) was performed using the standardize_numerical_data function, which:

- Makes a copy of original data to avoid direct DataFrame modification
- Verifies that numerical columns exist in the received data
- Applies standardization if valid columns exist

For this last operation, I used StandardScaler which calculates mean and standard deviation for each numerical column, transforms values using the formula: $z = (x - \mu) / \sigma$ where x is the value, μ is the mean, and σ is the standard deviation.

Output files:

- heart_final_cleaned.csv, news_final_cleaned.csv
- replaced_extreme_heart_num_data.csv, replaced_extreme_news_num_data.csv
- standardized_heart_data.csv, standardized_news_data.csv
- filled_heart_data.csv, filled_news_data.csv

3. Machine Learning Algorithm Implementation

Three algorithms were implemented: Decision Trees, Random Forests, and MLP.

The prepare_heart_data_for_rf and prepare_news_data_for_rf functions were initially used to prepare data for Random Forest algorithm training. However, I later observed they could be used for the other two algorithms to ensure preprocessing consistency.

Decision Trees (decision_tree.py)

The code implements a complete training and testing pipeline for a decision tree classifier for two datasets: Heart Disease - binary classification (chd_risk) and News Popularity - multi-class classification (popularity_category).

Main Stages:

1. **Data Preparation:** Eliminates unimportant attributes, treats missing values and outliers, standardizes numerical columns, applies label encoding or one-hot encoding for categorical variables.
2. **Model Training:** Configures DecisionTreeClassifier with optimized hyperparameters, trains on training set, predicts on test set, calculates accuracy.
3. **Experiment Execution:** Runs entire pipeline and returns models and results.

Hyperparameters for Heart Disease:

- max_depth: 10 (limits tree levels to prevent overfitting)
- min_samples_split: 20 (split allowed only with ≥20 examples)
- min_samples_leaf: 10 (leaf must contain minimum 10 examples)
- criterion: 'gini' (uses Gini index to measure impurity)
- class_weight: 'balanced' (automatically adjusts class weights for imbalance)

Hyperparameters for News Popularity:

- max_depth: 15 (slightly deeper tree for more complex dataset)
- min_samples_split: 50 (reduces overfitting risk)
- min_samples_leaf: 20 (stabilizes final decisions)
- criterion: 'entropy' (uses Information Gain)
- class_weight: 'balanced'

Random Forests (random_forests.py)

An improved version of tree-based classification using the Random Forest algorithm for the same two datasets.

Main Stages:

1. **Data Preparation:** Similar preprocessing to decision tree
2. **Model Training:** Trains RandomForestClassifier with optimized hyperparameters, uses class weighting to counteract imbalance, evaluates performance on test set

3. **Experiment Execution:** Runs entire process for both sets, returns models, scores, and predictions

Hyperparameters for Heart Disease:

- max_depth: None (no limitation, trees grow to maximum necessary depth)
- min_samples_leaf: 1 (allows very fine granularity but may lead to overfitting)
- criterion: 'gini'
- class_weight: 'balanced'
- n_estimators: 1000 (very robust model with reduced variance)
- max_samples: 1500 (each tree trained on random sample of 1500 examples)
- max_features: 'sqrt' (default for classification, introduces diversity between trees)

Hyperparameters for News Popularity:

- max_depth: 30 (limits tree depth to control model complexity)
- min_samples_leaf: 2
- criterion: 'entropy' (preferable for multi-class problems)
- class_weight: 'balanced'
- n_estimators: 1000
- max_samples: 1500
- max_features: 'sqrt'

MLP (mlp.py)

Implements a complete pipeline for data preparation and training of an MLP (MLPClassifier) for both datasets.

Process Stages:

1. **Data Preparation:** Eliminates irrelevant columns, replaces missing values and outliers, standardizes numerical columns, applies one-hot encoding for categorical columns in news dataset
2. **MLP Model Training:** Splits data into features (x) and labels (y), configures and trains MLP classifier, calculates accuracy on test set
3. **Experiment Execution:** Runs complete pipeline for both sets, displays performance results, returns models, predictions, and scores

Network Architecture - Heart Disease:

- hidden_layer_sizes: (100, 50, 25) - 3 fully connected layers
- activation: 'relu' (chosen for fast learning and avoiding vanishing gradient problem)

Network Architecture - News Popularity:

- hidden_layer_sizes: (200, 100, 50) - larger network for increased data complexity
- activation: 'relu'

Optimizer and Learning Parameters (Both Models):

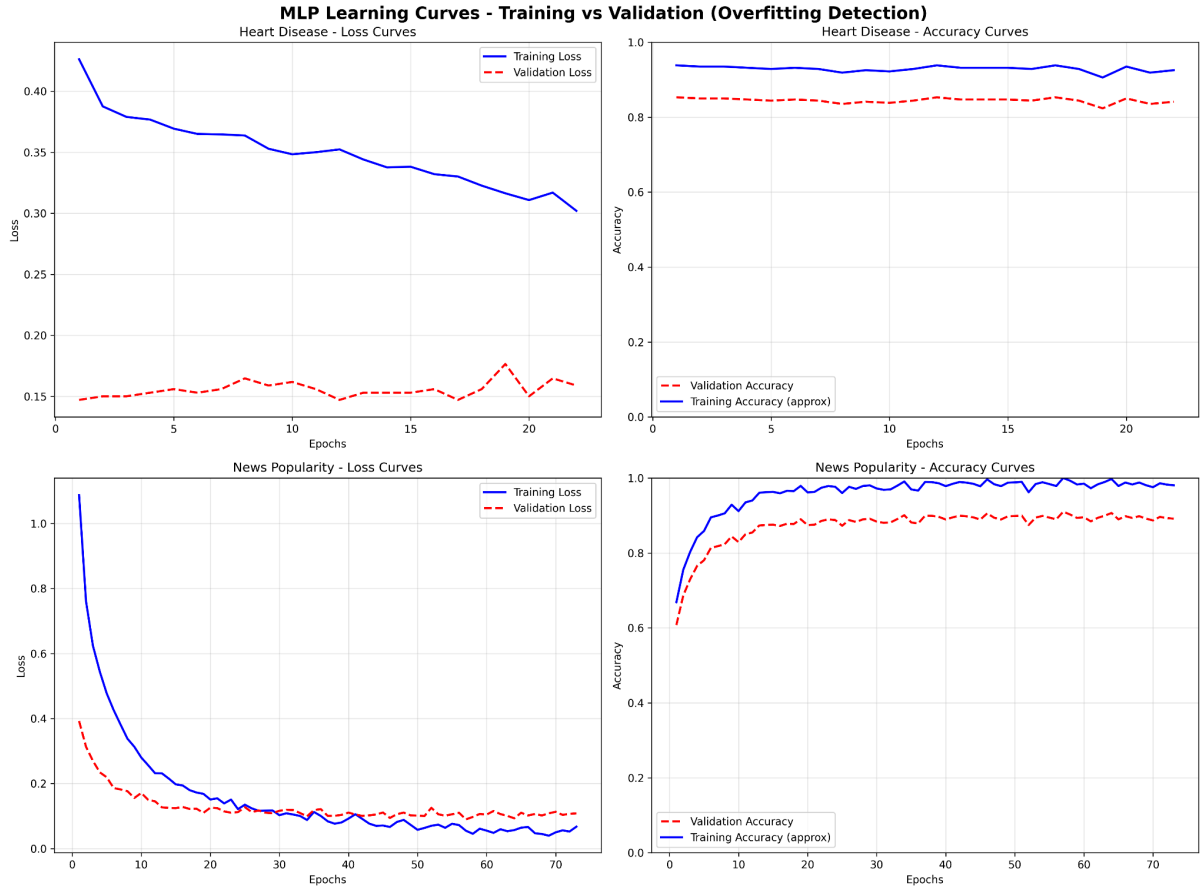
- solver: 'adam' (adaptive momentum-based algorithm, very efficient)
- learning_rate_init: 0.01 for Heart, 0.001 for News
- learning_rate: 'adaptive' (automatically decreases if no performance improvement)
- max_iter: 500 for Heart, 300 for News
- Batch size: Default (automatic mini-batch learning)

Regularization and Overfitting Prevention:

- alpha: 0.001 for Heart, 0.0001 for News (L2 regularization/ridge penalty)
- early_stopping: True (training stops if validation performance doesn't improve)
- validation_fraction: 0.1 (10% of training data reserved for internal validation)
- n_iter_no_change: 20 epochs for Heart, 15 for News

MLP Learning Curves Visualization:

The plot_mlp_learning_curves function visualizes the MLP learning process for both datasets. It generates a figure with 4 subplots showing loss curves, accuracy curves, training and validation metrics. The function automatically checks for overfitting and saves the graph as PNG.



Interpretation:

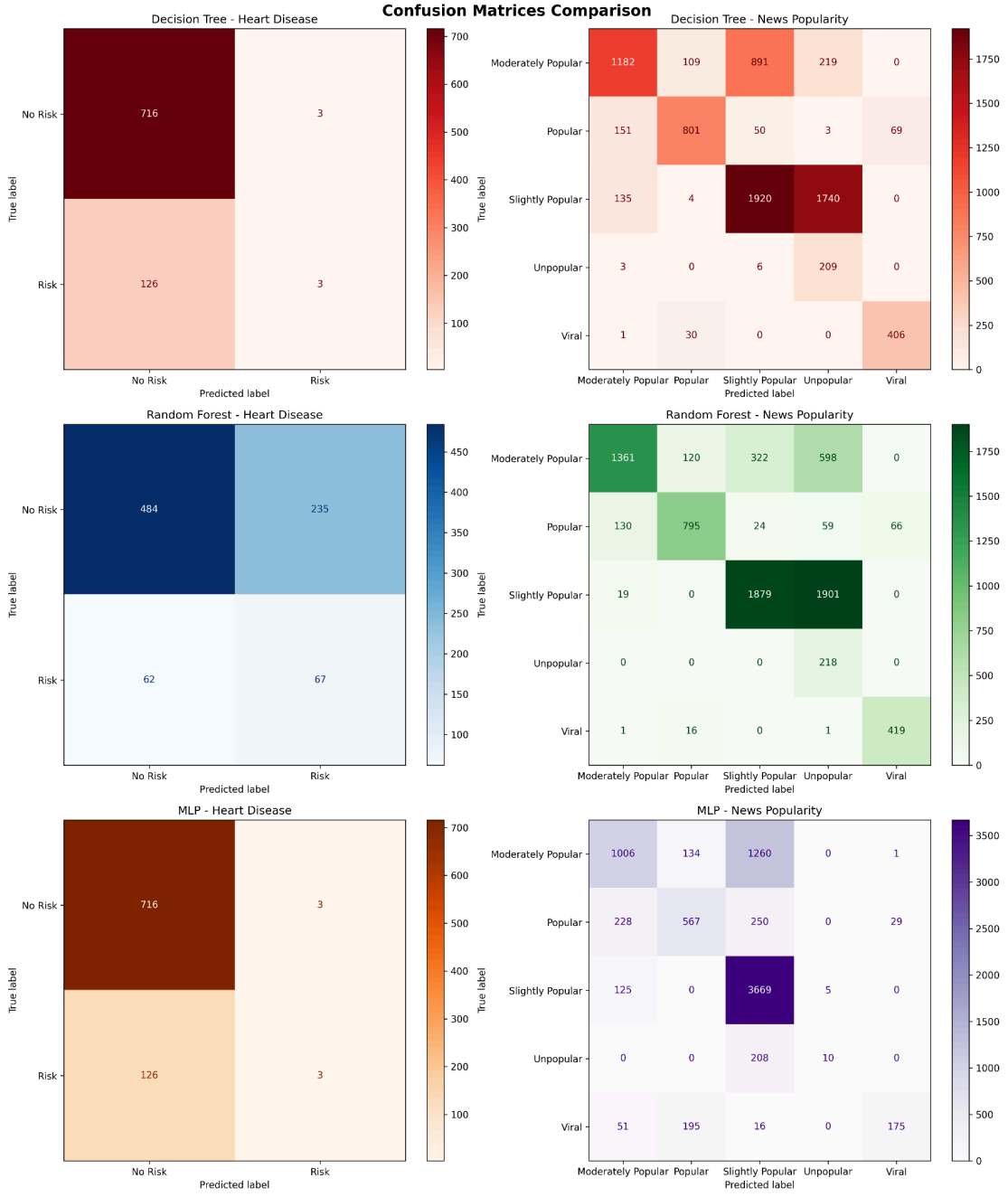
Heart Disease:

- Loss: Validation loss is constant and low (<0.2), while training loss is higher. This may indicate a stable model (slight underfitting) or simplified validation set.
- Accuracy: Validation accuracy is stable (~0.85), slightly lower than training (~0.90), with no clear signs of overfitting.

News Popularity:

- Loss: Training loss drops rapidly to very low (~0.05), while validation stagnates at ~0.1–0.15.
- Accuracy: Training accuracy reaches nearly 100%, but validation accuracy stagnates below 90%. This suggests overfitting: the model learned training data very well but doesn't generalize as well to new data.

Confusion Matrices Analysis



Heart Disease

Decision Tree and MLP are very poor at detecting "Risk" class:

- Both: only 3 correct classifications out of 129 real risk cases (recall of only 2.3%)
- Both miss massively people with risk (126 false negatives), incorrectly classifying them as "No Risk"

Random Forest is much better at risk detection:

- 67 correct classifications out of 129 real risk cases (recall of 52%)
- Only 62 false negatives

News Popularity

MLP is the weakest algorithm for most classes:

- "Moderately Popular": only 1006 correct classifications out of ~2401 cases (recall of ~42%)
- "Popular": only 567 correct out of ~1074 cases (recall of ~53%)
- "Unpopular": only 10 correct out of ~218 cases (recall of ~5%)
- "Viral": only 175 correct out of ~437 cases (recall of ~40%)

Decision Tree and Random Forest are much more balanced:

- Random Forest excels at "Moderately Popular" with 1361 correct classifications and is perfect at "Unpopular" with 218/218
- Decision Tree is very good at "Viral" with 406/437 correct and at "Unpopular" with 209/218

Comparative Performance Table

Dataset,Algorithm,Accuracy,Precision,Recall,F1-Score
Heart Disease,Decision Tree,0.8479,0.7971,0.8479,0.7846
Heart Disease,Random Forest,0.6498,0.7853,0.6498,0.6961
Heart Disease,MLP,0.8479,0.7971,0.8479,0.7846
News Popularity,Decision Tree,0.5698,0.7287,0.5698,0.6222
News Popularity,Random Forest,0.5892,0.8428,0.5892,0.6711
News Popularity,MLP,0.6844,0.6925,0.6844,0.6523

Analysis

Heart Disease: Decision Tree and MLP display identical values for all metrics (Accuracy=0.8479, F1=0.7846), confirming their similar behavior observed in confusion matrices - both algorithms classify nearly everything as "No Risk", achieving high accuracy through simple numerical dominance of this class. However, this performance is misleading because they massively miss

detecting people with cardiac risk (only 3 out of 129 cases detected correctly). Random Forest, despite having lower global accuracy (0.6498), offers a much more balanced and clinically useful approach, effectively detecting people with risk.

News Popularity: MLP appears to have the best global accuracy (0.6844), but Random Forest excels at precision (0.8428), suggesting that when Random Forest makes a prediction, it's trustworthy. However, aggregated metrics don't reflect specific class challenges - for example, all algorithms struggle with detecting "Unpopular" or "Viral" articles, which are minority but important classes.

Algorithm Performance Analysis

From a global accuracy perspective, Decision Tree and MLP dominate for Heart Disease (84.79%), while MLP leads for News Popularity (68.44%). This apparent performance is due to a conservative classification strategy - algorithms learn to exploit class imbalance, classifying most cases in the majority class. For Heart Disease, they achieve high accuracy by classifying nearly all patients as "No Risk", which is statistically correct but clinically dangerous.

From a practical utility perspective, Random Forest proves superior through its balance in detecting all classes. For Heart Disease, Random Forest sacrifices global accuracy to effectively detect people with cardiac risk (recall of 52% vs 2.3% for others), which is essential in a medical context. For News Popularity, Random Forest achieves the highest precision (84.28%), indicating its predictions are trustworthy.

Random Forest excels by aggregating multiple decision trees trained on different data subsets, successfully capturing data variability and complexity, avoiding overfitting and bias toward the majority class. This robustness makes it more suitable for practical applications where correct detection of all classes is crucial.

Tools Used:

- Claude (https://claude.ai/new) and ChatGPT (https://chatgpt.com/) for graph generation, complex analysis, and function usage guidance
- Links provided in the documentation