

Predicting Online News Popularity: A Comparative Study of Machine Learning Models with Boruta Feature Selection and SHAP Interpretation

Yuyang Che

*School of Humanities, Central South University, Changsha, China
cheyuyang@csu.edu.cn*

Abstract. Predicting the popularity of online news articles matters for content recommendation platforms, digital marketing strategies, and information dissemination research. This study compares the performance of seven machine learning models for binary classification of online news popularity using the UCI Online News Popularity dataset. We apply Boruta feature selection to identify 41 statistically significant features from 58 candidates, then GridSearchCV-optimized training of Random Forest, XGBoost, LightGBM, Logistic Regression, Linear SVC, Decision Tree, and K-Nearest Neighbors (KNN) classifiers. Experimental results on 39,644 articles show XGBoost performing best at an AUC of 0.7304 and F1-score of 0.6996. We use SHAP (SHapley Additive exPlanations) to explain the predictions of the best-performing model, and the analysis shows that `kw_avg_avg`, which reflects keyword information, together with the publication time feature `is_weekend`, has the strongest influence on news popularity; these results can provide useful reference for content creators and platform designers when they adjust content distribution strategies.

Keywords: news popularity prediction, Boruta feature selection, XGBoost, SHAP, machine learning

1. Introduction

The rapid growth of online news platforms generates vast content volumes. Publishers and platforms struggle to predict which articles will reach wide audiences. Identifying what drives popularity matters for content recommendation algorithms, digital journalism, advertising revenue, and information ecosystem management [1]. Early work by Bandari et al. [2] showed that features extracted from article metadata and early traffic patterns can forecast sharing volumes with reasonable accuracy, establishing online news popularity prediction as an active research area spanning data mining [3], natural language processing, and social network analysis. Subsequent studies have explored more sophisticated approaches, from ensemble methods and gradient boosting to deep learning architectures. While content-based features carry predictive signal, the task remains challenging due to complex, non-linear relationships between article attributes and user sharing behavior [1]. Most prior work has focused on predictive accuracy, with less attention to model

interpretability and feature selection rigor. Recent advances in explainable AI, particularly SHAP-based methods [4], can decompose model predictions into interpretable feature-level contributions, yet their application to news popularity prediction remains underexplored.

In recent years, scholars have applied explainable machine learning to interdisciplinary domains ranging from healthcare [5] to social resilience analysis [6, 7], showing that SHAP-based interpretation connects predictive modeling to practical insight. Building on this work, this study makes three contributions. First, we apply Boruta feature selection [8]—a Random Forest-based wrapper method—to identify all relevant features, avoiding arbitrary importance thresholds. Second, we compare seven machine learning algorithms with hyperparameter optimization via GridSearchCV: tree-based ensembles (Random Forest, XGBoost, LightGBM), linear models (Logistic Regression, Linear SVC), and instance-based methods (KNN, Decision Tree). Third, we use SHAP values [4] to explain the best-performing model, linking predictions to practical guidance on content strategy and publication timing.

Section 2 describes the dataset and preprocessing. Section 3 covers the experimental setup—feature selection and model configuration. Section 4 reports experimental results and analysis. Section 5 provides SHAP-based model interpretation. Section 6 concludes with findings and future directions.

2. Dataset and preprocessing

The Online News Popularity dataset [9, 10] contains 39,644 articles with 58 numerical features and one target variable (shares). The features fall into five categories: (1) Content: word count (`n_tokens_content`), image and video counts (`num_imgs`, `num_videos`), hyperlinks (`num_hrefs`, `num_self_hrefs`); (2) Linguistic: subjectivity (`global_subjectivity`), sentiment polarity (`global_sentiment_polarity`), title characteristics (`title_subjectivity`, `title_sentiment_polarity`); (3) Keyword metadata: average, maximum, and minimum shares of referenced keywords (`kw_avg_avg`, `kw_max_avg`, `kw_min_avg`, `kw_avg_max`, etc.); (4) Temporal: day-of-week indicators (`weekday_is_*`), data channel categories (`data_channel_is_*`); (5) NLP: LDA topic distributions (`LDA_00`–`LDA_04`), global sentiment metrics. The target variable, shares, records the number of times each article was shared on social media, ranging from 1 to 843,300 (median = 1,400, mean = 3,395, SD = 11,526), with a heavily right-skewed distribution. We convert this to binary classification following established practice [9] by thresholding at the median: articles with shares $\geq 1,400$ as High Popularity (Class 1, $n = 21,154$, 53.4%), others as Low Popularity (Class 0, $n = 18,490$, 46.6%). The resulting dataset is balanced (53.4% High, 46.6% Low).

Preprocessing proceeds in three stages. First, we remove two non-predictive identifier columns (`url`, `timedelta`), leaving 58 features. The binary target is constructed via median thresholding. We retain all 39,644 articles. Second, we apply median imputation (scikit-learn's `SimpleImputer`) followed by `StandardScaler` normalization to zero mean and unit variance. Standardization matters for distance-based methods (KNN) and regularized linear models (Logistic Regression, Linear SVC), where features with larger numerical ranges would otherwise dominate. The dataset is split into training (80%, $n = 31,715$) and testing (20%, $n = 7,929$) sets using stratified sampling to maintain class balance. Third, we apply Boruta feature selection on the training set only, preventing data leakage from the test set. The Boruta analysis (Section 3) reduces the feature space from 58 to 41 features, a 29% reduction while preserving all predictive features.

3. Methodology

3.1. Feature selection

Boruta is an all-relevant feature selection algorithm based on Random Forest importance measures [8]. Unlike methods that select a minimal-optimal feature subset for a specific model, Boruta identifies all features with statistically significant associations to the target. The algorithm creates shadow features by randomly permuting the originals, then iteratively compares each real feature's importance against the maximum importance among all shadow features across multiple Random Forest runs. In each iteration, features significantly outperforming the best shadow feature are confirmed as important; those consistently underperforming are rejected; and borderline features remain tentative for further evaluation. We configure the Boruta selector with 300 Random Forest estimators ($n_estimators=300$), a maximum tree depth of 7 to prevent overfitting, up to 150 iterations ($max_iter=150$), and a 90th percentile ($perc=90$) comparison threshold. The analysis is conducted on the full training set comprising 31715 samples after preprocessing, using 5-fold cross-validation within the Random Forest to ensure robust importance estimates. The Boruta process converges after classifying all 58 features: 41 confirmed as statistically relevant, 17 rejected as noise, and zero tentative, separating informative from noise features. The confirmed features span all five original categories: keyword metadata (8 confirmed), NLP features (5 LDA topics, `global_sentiment_polarity`), content properties (`n_tokens_content`, `n_unique_tokens`, `num_hrefs`, `num_self_hrefs`), temporal indicators (`is_weekend`, weekday indicators), and linguistic features (`global_subjectivity`, `rate_positive_words`). Features related to video and image counts (`num_videos`, `num_imgs`) are rejected, indicating that multimedia content volume alone does not reliably predict sharing behavior.

3.2. Machine learning models

We evaluate seven machine learning algorithms selected to span complementary modeling paradigms:

Random Forest (RF) [11] is a bagging ensemble of decision trees trained on bootstrap samples with random feature subsets at each split. We tune $n_estimators \in \{200, 300\}$, $max_depth \in \{7, None\}$, and $min_samples_leaf \in \{1, 3\}$ via `GridSearchCV`. XGBoost [12] implements regularized gradient boosting [13] with depth-wise tree growth, L1/L2 regularization, and native missing value handling; the grid search covers $n_estimators \in \{200, 300\}$, $max_depth \in \{4, 6\}$, $learning_rate \in \{0.05, 0.1\}$, and $subsample \in \{0.8, 1.0\}$. LightGBM [14] is a gradient boosting variant optimized for efficiency through histogram-based binning and leaf-wise tree growth; we pre-optimize it with $n_estimators=200$, $max_depth=5$, $learning_rate=0.05$ after empirical tuning.

Logistic Regression provides a well-calibrated probabilistic baseline with L2 regularization; the grid search explores $C \in \{0.1, 0.5, 1.0, 5.0, 10.0\}$ with $max_iter=2000$. Linear SVC, augmented with Platt probability calibration via `CalibratedClassifierCV`, is pre-optimized with $C=0.1$. Decision Tree offers full interpretability as a single-tree model, tuned over $max_depth \in \{5, 8, None\}$, $min_samples_leaf \in \{1, 5\}$, and $criterion \in \{gini, entropy\}$. K-Nearest Neighbors (KNN) is an instance-based classifier with $n_neighbors \in \{7, 15, 25\}$ and $weights \in \{uniform, distance\}$.

All models are optimized via `GridSearchCV` with 5-fold stratified cross-validation on the training set, using ROC-AUC as the primary metric. AUC is preferred over accuracy because it is insensitive to class imbalance and evaluates ranking quality across all classification thresholds [15]. We

evaluate on the held-out test set using five metrics—Accuracy, Precision, Recall, F1-score, and ROC-AUC—with AUC and F1-score serving as the primary criteria for model comparison.

All experiments use scikit-learn 1.4+ [16], XGBoost 2.0+ [12], and LightGBM 4.0+ [14] on a standard workstation. Random states are fixed (random_state=42) for reproducibility, and the complete preprocessing pipeline—imputation, scaling, and feature selection—is saved as serialized joblib objects. Per-model computational cost, including GridSearchCV with 5-fold validation, ranges from under 1 second (Logistic Regression) to approximately 7 seconds (Random Forest with 6 hyperparameter combinations).

4. Experimental results

4.1. Overall performance comparison

Table 1 presents the performance metrics for all seven models on the held-out test set of 7,929 samples. Gradient boosting methods (XGBoost, LightGBM) achieve the highest overall performance, followed by Random Forest, with linear models and instance-based methods trailing behind.

Table 1. Model performance comparison on test set

Model	AUC	F1	Accuracy	Precision	Recall
XGBoost	0.7304	0.6996	0.6701	0.6804	0.7199
LightGBM	0.7283	0.6994	0.6701	0.6806	0.7192
RandomForest	0.7226	0.6991	0.6645	0.6704	0.7303
LogisticRegression	0.7060	0.6832	0.6542	0.6683	0.6987
LinearSVC	0.7048	0.6859	0.6542	0.6655	0.7076
KNN	0.6932	0.6609	0.6418	0.6679	0.6540
DecisionTree	0.6859	0.6781	0.6344	0.6395	0.7216

XGBoost obtains the best performance, with an AUC of 0.7304, an F1-score of 0.6996, and an accuracy of 0.6701, while LightGBM performs very closely, reaching an AUC of 0.7283, an F1-score of 0.6994, and the same accuracy of 0.6701. Random Forest also performs well and ranks third, with an AUC of 0.7226 and an F1-score of 0.6991. Compared with the best linear model, Logistic Regression (AUC: 0.7060), the top three ensemble models improve the AUC by about 0.024 and the F1-score by around 0.016, which indicates that non-linear relationships among features can improve prediction performance, although the improvement is not very large. These ensemble models mainly show advantages in ranking ability instead of overall accuracy. The accuracy of all seven models falls within a relatively small range, from 63.4% to 67.0%, which further reflects that predicting news popularity remains a challenging task [9]. Among all models, the Decision Tree achieves the highest Recall (0.7216), meaning that it identifies more truly popular news articles, but its Precision is the lowest at 0.6395. In contrast, KNN has the lowest Recall (0.6540) and a moderate Precision of 0.6679, showing a relatively conservative prediction pattern. These differences suggest that different models may be suitable for different application needs. For example, a content recommendation system that focuses more on recall may prefer ensemble

models, while tasks requiring higher precision can further improve performance by adjusting the decision threshold.

Figure 1 provides a visual comparison of model performance. The left panel displays multi-metric scores across all seven models, showing consistent patterns across Accuracy, Precision, Recall, and F1-score. The right panel shows ROC curves with AUC values; models with higher AUC exhibit curves more strongly bowed toward the upper-left corner, indicating superior discrimination between high and low popularity classes.

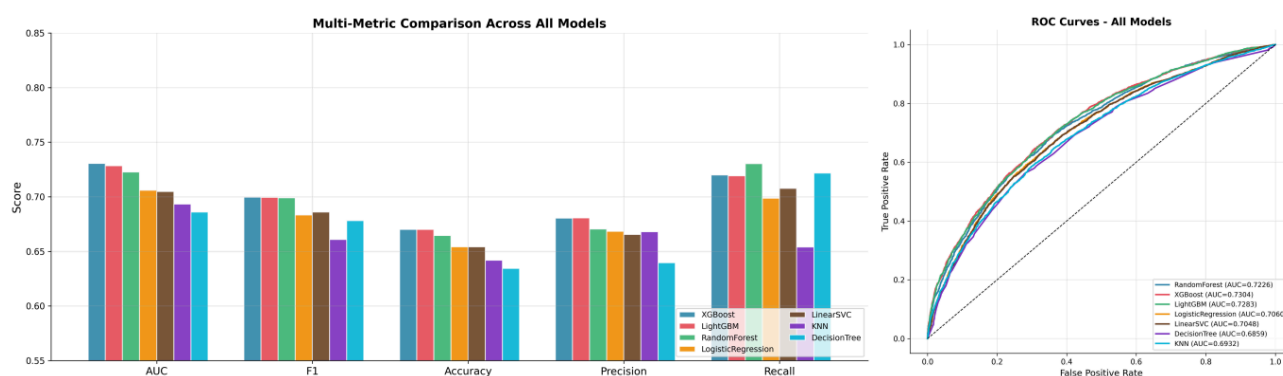


Figure 1. Multi-metric comparison (left) and ROC curves (right) across all seven models

4.2. Model behavior analysis

Gradient boosting models, including XGBoost and LightGBM, perform slightly better than the bagging model Random Forest, with an average AUC improvement of about 0.008. This result is related to the boosting process, where each new tree is built to correct the prediction errors made by the previous ones. However, the difference is relatively small, indicating that Random Forest is still a competitive choice, especially in situations where model stability and interpretability are important. Logistic Regression (AUC = 0.7060) and Linear SVC (AUC = 0.7048) achieve about 96.5%–96.7% of the AUC obtained by the best ensemble model, showing that linear models are able to capture a large part of the useful prediction information. The remaining improvement of about 3.5% achieved by the non-linear models suggests that some feature interactions, especially those between temporal features and keyword-related information, are difficult for linear models to learn.

KNN (AUC: 0.6932) performs worse than the tree-based models, which reflects the limitations of distance-based methods when dealing with 41-dimensional data. Although the features are standardized, Euclidean distance becomes less effective in a high-dimensional feature space. The Decision Tree (AUC: 0.6859) shows the common trade-off between bias and variance for a single-tree model. It achieves relatively high Recall, but its performance is less stable than that of the ensemble models. Looking at all seven models together, the AUC values range from 0.686 to 0.730, indicating that content-related features alone are not sufficient to accurately predict online news popularity. Other factors, such as social network interactions, platform recommendation mechanisms, news events, and unexpected viral spread, are also likely to influence the prediction results and explain part of the remaining variation. This observation is consistent with previous studies [11] and also suggests that future work can further improve prediction performance by introducing multi-modal information and temporal features.

5. SHAP-based model interpretation

5.1. Motivation and methodology

In addition to predicting performance, it is also important to understand the reasons why the model produces certain predictions, because it helps to build trust and extract practical insights [17]. In this study, SHAP (SHapley Additive exPlanations) [4, 18] is used for the best-performing XGBoost model to break down the prediction into the contribution of the feature level. The SHAP framework is based on cooperative game theory, in which each feature is assigned a value to reflect its marginal contribution to the final prediction, which is calculated through different feature combinations; this process follows the principles of local accuracy, missingness, and consistency. Using TreeExplainer, the SHAP value is calculated on a subset of 3,000 randomly selected test samples, in order to keep the calculation manageable while maintaining a reasonable coverage of the data. Therefore, a matrix with a size of 3000×41 is obtained, which can not only conduct overall analysis across samples, but also explain individual prediction cases.

5.2. Global feature importance

Figure 2 shows three parts of the SHAP results. The left panel is a summary diagram of the bee colony, which shows the SHAP value distribution of the top 20 characteristics in all 3,000 samples. Each point corresponds to one prediction; the horizontal axis shows the SHAP value, where positive values indicate a push toward the High Popularity class and negative values indicate a push toward the Low Popularity class, while the color represents the feature value (red indicates high values and blue indicates low values). The middle panel is a bar chart, which sorts the features according to its average absolute SHAP value. The right panel is a waterfall diagram, which decomposes individual predictions and shows how each feature helps to move the result out of the baseline value.

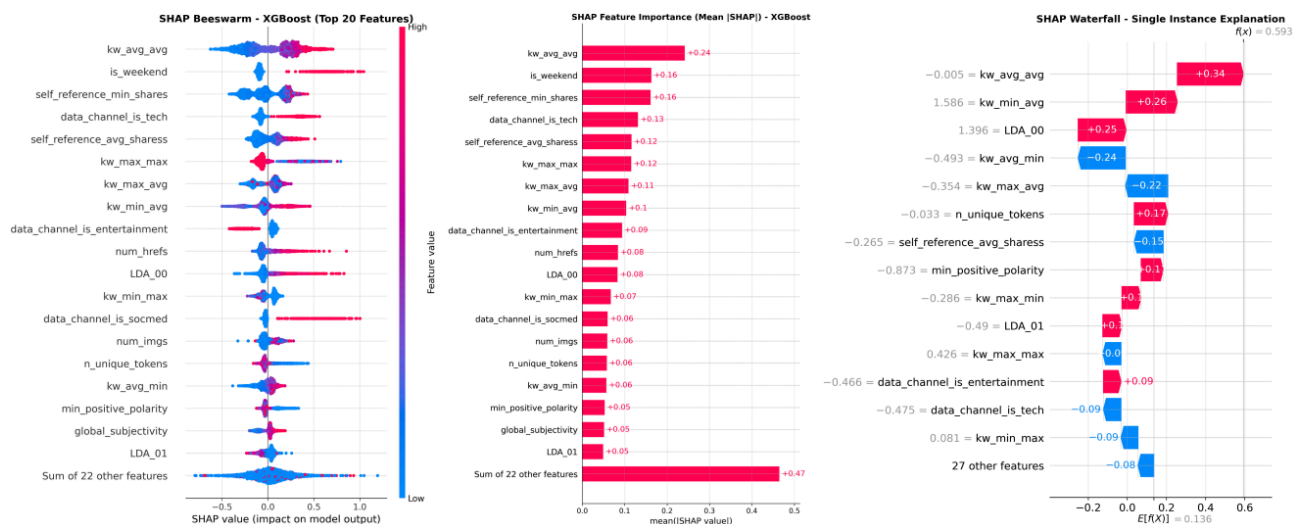


Figure 2. SHAP analysis results. Left: Beeswarm summary plot showing feature impact distribution (top 20 features). Center: Mean |SHAP| bar chart. Right: Waterfall plot decomposing a single prediction

5.3. Key predictive features

Several patterns can be observed from the SHAP results. Feature `kw_avg_avg` represents the average number of keyword shares, which is the most important factor. The higher value of this feature usually turns the prediction to a high popularity level. This shows that topic selection plays an important role in content coverage; articles that use historically engaging keywords tend to get more sharing. In contrast, factors such as writing length, style characteristics or multimedia-related variables seem to have a weaker impact on popularity.

The binary feature `is_weekend` ranks second in terms of importance. Overall, its effect is positive, which means that articles published on weekends are more likely to be popular. This may be related to user behavior, because readers usually have more free time on weekends, which increases content consumption and sharing activities. Self-reference feature (`self_reference_avg_shares` and `self_reference_min_shares`) is also very important. These variables reflect the shared history of the cited Mashable article, and they capture a domain-level popularity signal. Generally speaking, articles that link to previous resources related to successful posts tend to perform better, which may indicate the network effect within the platform.

Other prominent features include `LDA_00`, `global_subjectivity` and `n_unique_tokens`. The importance of `LDA_00` shows that in addition to keyword-level signals, the potential theme structure is still important to popularity. The `global_subjectivity` feature shows an overall negative correlation with popularity, and the effect size is relatively small (average $|\text{SHAP}|$ about 0.05), indicating that more subjective writing is unlikely to be widely shared. The `n_unique_tokens` feature shows heterogeneous effects across samples. In the swarm chart, it usually shows a negative trend, which means that articles with very high vocabulary diversity perform poorly on average. However, in some individual cases, it can still make a positive contribution (for example, +0.17 in the waterfall example), depending on the combination of other features. This change shows that the global average cannot fully describe the characteristic effect, and local explanation is still necessary.

At the instance level, the waterfall diagram on the right side of Figure 2 explains a single prediction step by step. In selected cases, `kw_avg_avg` contributed the largest upward shift (+0.34), followed by `kw_min_avg` (+0.26) and `LDA_00` (+0.25). These features increase the prediction probability higher than the baseline. In contrast, `kw_avg_min` (-0.24) and `kw_max_avg` (-0.22) produce the strongest decline effect. This decomposition provides interpretable and actionable insights into individual predictions. If such a system is deployed, in practice, it can help content creators adjust article features before publication.

6. Conclusion and future work

We investigated online news popularity prediction using the UCI Online News Popularity dataset. We applied Boruta feature selection, trained seven machine learning models with GridSearchCV, and employed SHAP-based interpretation. Our conclusions follow.

Feature selection and model comparison produced two findings. First, Boruta identified 41 out of 58 features as statistically relevant (confirmed = 41, rejected = 17), reducing dimensionality by 29% while preserving all informative features. The rejected features, mainly multimedia counts and certain NLP-related metrics, showed no statistically significant relationship with sharing behavior beyond what the confirmed features capture. Second, tree-based ensemble methods show better performance than both linear models and instance-based methods. Among them, XGBoost performs the best, reaching an AUC of 0.7304 and an F1-score of 0.6996. The difference of about 0.024 in

AUC between ensemble models and linear models indicates that there are non-linear patterns in the feature space, although the gain is not very large.

The SHAP analysis of the best-performing model shows that the most important predictors can be roughly grouped into two main types. The first group is related to keyword and metadata information. Features such as `kw_avg_avg`, `kw_max_avg`, and `self_reference_avg_shares` have relatively large contributions to the model output. This suggests that topic choice and keyword strategy matter for news reach; in particular, selecting keywords that have shown higher engagement in the past seems more helpful for increasing article popularity than changes in writing length, style, or multimedia-related features. The second group is related to timing. The feature `is_weekend` ranks second in SHAP importance, which indicates that publication time has an effect on audience attention. Articles published on weekends tend to receive more exposure, which can provide some reference for scheduling decisions.

However, even though these patterns are observed, the overall model performance remains only moderate, with AUC values ranging from 0.686 to 0.730 across all models. This suggests that predicting news popularity using only content-based features is still difficult. A large part of the unexplained variation may come from external factors such as social network interactions, platform recommendation mechanisms, and sudden viral events that are hard to capture in the dataset. Future work may extend this study in several directions. First, multi-modal features can be introduced by using pre-trained vision models such as CLIP or ViT to capture image-related signals that are not included in text features. Second, temporal sequence models like LSTM or Temporal Fusion Transformers could be applied to better model changes in news dynamics over time. Third, external signals from other platforms, such as Twitter/X engagement, Reddit discussions, and Google Trends, may be added to reflect cross-platform attention diffusion. Fourth, causal inference methods could be used to better estimate the causal impact of specific content features on popularity. Finally, practical applications such as real-time SHAP-based dashboards could be developed to provide feedback to editors during content creation.

References

- [1] Tatar, A., de Amorim, M. D., Fdida, S., & Antoniadis, P. (2014). A survey on predicting the popularity of web content. *Journal of Internet Services and Applications*, 5(1), 1–20.
- [2] Bandari, R., Asur, S., & Huberman, B. A. (2012). The pulse of news in social media: Forecasting popularity. In *Proceedings of the International AAAI Conference on Weblogs and Social Media (ICWSM 2012)* (pp. 26–33).
- [3] Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data mining: Practical machine learning tools and techniques* (4th ed.). Morgan Kaufmann.
- [4] Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS 2017)* (pp. 4765–4774).
- [5] Jiang, W., Zhang, Y., Weng, J., Song, L., Liu, S., Li, X., & Zheng, R. (2025). Explainable machine learning model for predicting persistent sepsis-associated acute kidney injury: Development and validation study. *Journal of Medical Internet Research*, 27, e62932.
- [6] Zhang, S., Zhang, L., Weng, J., Gasevic, D., Wei, Y., Chen, Z., & Jian, W. (2025). Evaluating community resilience through social media during China's first post-COVID-19 reopening: Insights from machine learning. *Journal of Global Health*, 15, 04315.
- [7] Zhang, S., Zhang, L., Liu, S., Weng, J., Jian, W., & Guo, J. (2025). The similarities and differences of multiple chronic diseases risk factors across depressive symptoms trajectories among middle-aged and older Chinese adults: A 10-year longitudinal cohort study. *Journal of Affective Disorders*, 120, 395.
- [8] Kursa, M. B., & Rudnicki, W. R. (2010). Feature selection with the Boruta package. *Journal of Statistical Software*, 36(11), 1–13.
- [9] Fernandes, K., Vinagre, P., & Cortez, P. (2015). A proactive intelligent decision support system for predicting the popularity of online news. In *Proceedings of EPIA 2015 (LNCS Volume 9273)* (pp. 535–546). Springer.

- [10] Dua, D., & Graff, C. (2017). UCI machine learning repository. *University of California, Irvine, School of Information and Computer Sciences*. <http://archive.ics.uci.edu/ml>
- [11] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32.
- [12] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)* (pp. 785–794).
- [13] Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5), 1189–1232.
- [14] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Proceedings of NeurIPS 2017*, 3146–3154.
- [15] Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. MIT Press.
- [16] Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- [17] Molnar, C. (2022). *Interpretable machine learning: A guide for making black box models explainable* (2nd ed.). <https://christophm.github.io/interpretable-ml-book/>
- [18] Lundberg, S., et al. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1), 56–67.