[GROUP 6] REPORT

2171009 Kim Jiwoo, 2171016 Park Jihyeun, 2176392 Choi Hanuy, 2491036 Ju Taein

Contents

1. EDA

2. Data preprocessing

2.1. Feature Selection Enhancement

2.1.1.PCA

2.1.2. SHAP

2.2. Feature Refinement

2.2.1. top15

2.2.2. VIF

2.3. Final Feature Set Construction

2.3.1. FullAugmentedSet_ver1

2.3.2. FullAugmentedSet_ver2

3. Modeling

3.1 Selection of Baseline Model

3.1.1 Linear Model Performance Evaluation: Logistic Regression and SVM

3.1.2 Tree-Based Model Performance Evaluation: Random Forest and XGBoost

3.2 Ensemble model building and optimization

3.2.1 Comparing Ensemble Techniques: Voting vs. Stacking

3.2.2 Performance Analysis by Stacking Model Dataset

3.3 Building a new dataset based on derivatives

3.4 Stacking Model Hyperparameter Tuning

3.4.1 Full AugmentedSet_ver1 Dataset-Based Grid Search

3.4.2. Full AugmentedSet_ver2 Dataset-Based Grid Search

3.4.3. Final Dataset and Hyperparameter Selection

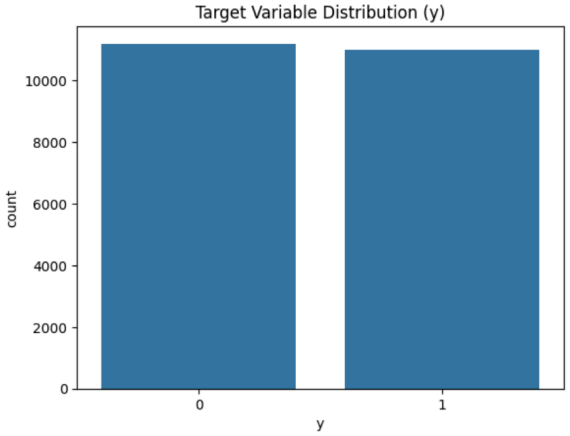
3.5 Final Model Selection and Performance Comparison

1. EDA

(1) Data exploration

i. y Distribution

The first thing to check in the classification problem is the class distribution of the target variable (y). This dataset is data on binary classification problems, and the y value is divided into two classes, 0 and 1. As a result of visualizing the frequencies of classes 0 and 1 in a bar graph, it was found that the two classes were relatively balanced. Therefore, the possibility of bias due to class imbalance is expected to be low when learning the model, and based on this, it aims to secure stable prediction performance.



y	proportion
0	0.504324
1	0.495676

ii. Feature Shape

A total of two categorical variables in the dataset were identified. One is channel, which indicates the publication channel of the article, and the other is week, which contains the day of the week in which the article was published. Since the two characteristics are different, we will proceed with the appropriate encoding. This encoding method contributes to enhancing the predictive power of the model by reflecting the unique characteristics of each variable.

data_channel	object
weekday	object

Other specific variables include id, share, and y. In addition to the target variable y to be predicted, id and share variables exist in the dataset. id is a unique identifier for each sample and is not used for analysis or modeling, but is used only when matching the final result.

On the other hand, shares are a numerical variable representing the actual number of article shares, and y, the target variable, is a binary variable derived based on these shares values. Therefore, shares are provided only for reference, and if used for learning, there is a risk of data leakage, so it must be excluded. In this project, shares were used only for target definition and then removed.

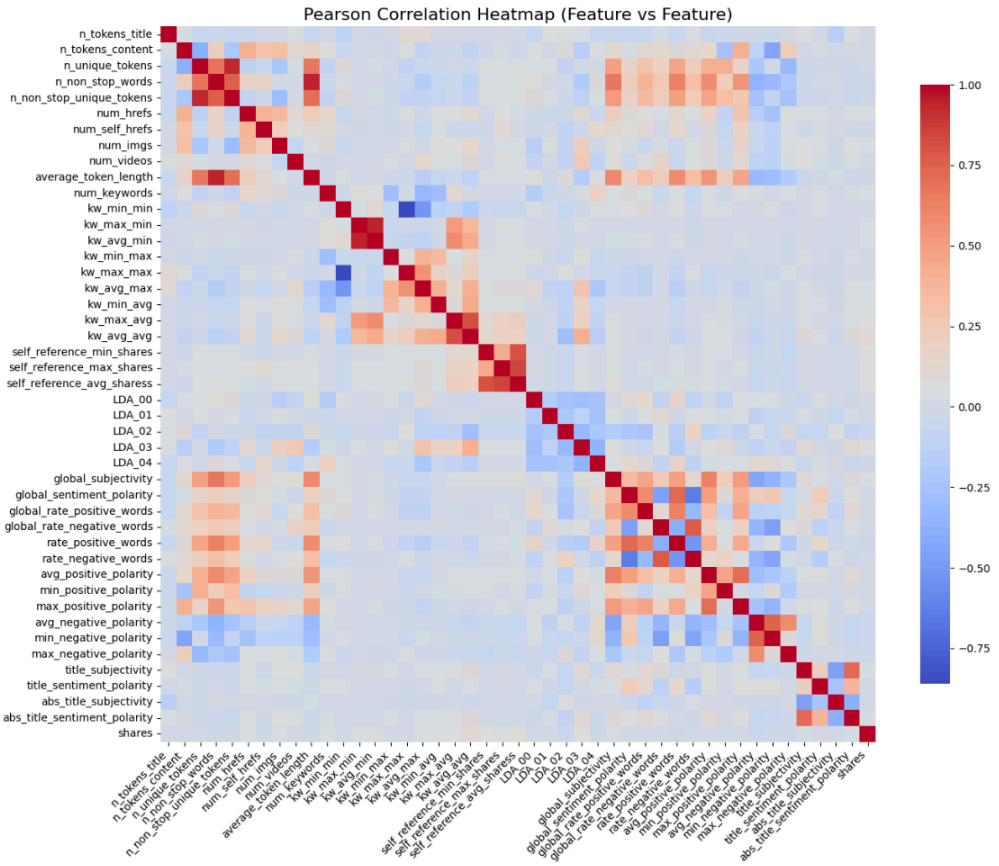
id	int64
shares	int64
y	int64

(2) Data Exploration

i. Heatmap

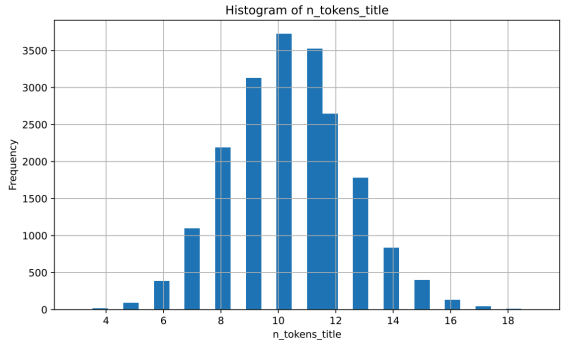
A heatmap visualizing the correlation between features after processing missing values with categorical encoding and mean shows that some variables show a high positive correlation (e.g., 0.8 or higher). In particular, there was a strong correlation in some keyword-related variables, including kw_avg_avg and kw_max_avg, and in the self_reference_min_shares and self_reference_max_shares variables.

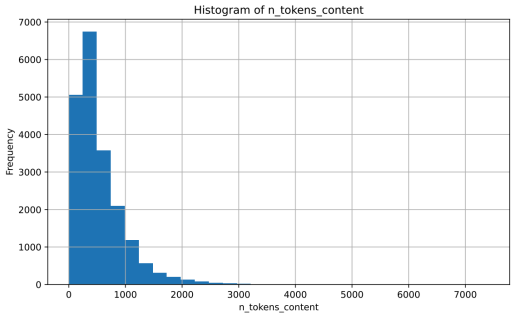
These highly correlated variables can cause a multicollinearity problem, so it is necessary to select appropriate variables or process such as dimension reduction before learning the model. In the future, the influence of these variables will be quantitatively evaluated through various indicators, and additional preprocessing will be performed to alleviate multicollinearity.

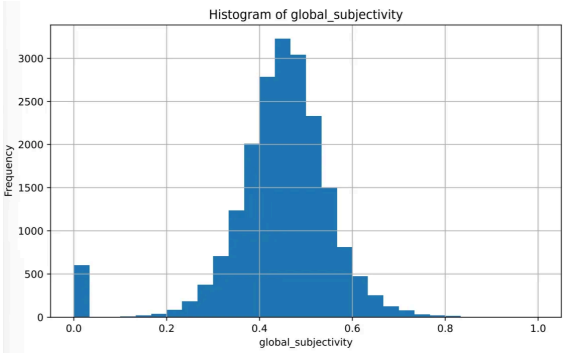


ii. Histogram

The distribution structure was reviewed by visualizing histograms for all numerical variables. Since the total number of features is large, we had a brief discussion about type identification and processing methods in the early stages. The following is a summary of each type of example and the preprocessing strategy applied accordingly, and the processing method tailored to the distribution characteristics can have an important influence on improving stability and performance when learning the model.







Normal-like	Most models are scale sensitive	Standardization
Right-skewed	Log transformation reduces long tail to close to normal	log conversion, Standardization
Multimodal	Information distortion is possible when treated as a single continuous variable	Review whether discrete values

2. Data preprocessing

2.1. Feature Selection Enhancement

In general, unnecessary variables are removed by applying the Feature Selection technique to improve model performance and reduce learning time. However, as a result of attempting Feature Selection in a simple and intuitive way, rather significant variables were excluded in advance, resulting in poor performance. In addition, there was a problem that the overall prediction performance was deteriorated because derivative information that could be obtained through combinations or nonlinear relationships between variables was removed together.

Based on these results, an approach was attempted from two directions.

The first was to switch the dimension reduction method itself, applying **PCA (main component analysis)** that can reduce duplicate information while minimizing information loss. PCA is a method that can reduce the dimension while preserving interpretable information as much as possible by linearly combining original variables and converting them into newly defined main components. Through this, we tried to consider both the efficiency and predictive performance of the model while maintaining important information.

The second was to supplement the existing Selection method, using the **SHAP value**. Feature Selection was intended to be applied by selecting meaningful variables by reflecting the correlation between variables or model-based importance. In particular, in this process, we tried to select variables based

on more reliable criteria than simple importance indicators by visualizing and interpreting the contribution of each variable using SHAP values.

2.1.1. PCA

(1) Overview

The principal component analysis (PCA) method is a method of summarizing high-dimensional data to fewer dimensions while preserving the most important information of the data as much as possible. In the figure above, when the original data is spread on the X and Y axes, the PCA finds the direction in which the data is most spread (PC1). This is the axis with the largest distribution and the largest amount of information, and it can be said that it contains most structural information. Then, the axis PC2 is orthogonal to PC1 and expresses less important information. Then, this makes it possible to reduce the dimension by leaving only PC1 and discarding PC2. In this way, PCA is an efficient and information-oriented data preprocessing by reconstructing the original coordinates.

Additionally, the dimension is reduced by converting high-dimensional data into independent principal components so that multicollinearity or noise between variables does not impede the performance of the model. The PCA mainly uses cumulative explanatory variance to select the number of principal components whose variance value satisfies more than 95%, which contributes to model performance improvement through dimensional reduction while maintaining more than 95% of the total information.

(2) Experiments and Results

i. Accumulated Description Variance

When applying PCA, one of the commonly used criteria is to select the number of main components so that the cumulative explanatory variance is above a certain percentage. In the first experiment, we adopted this and selected 32 main components that preserve more than 95% of the total information. However, we thought that cumulative explanatory variance alone could be different from the information of the y-value required by the predictive model. Since it can include high-dispersion components or noise irrelevant to the y-value, cross-validation was performed to adjust the number of main components based on the actual predictive performance.

ii. Experimental Method : cross-validation based principal component optimization

Model: logistic regression (max_iter=1000)

Verification: Cross-validation score (cv = 5, score = "accuracy")

Variables: Number of PCA dimensions

In addition, in the experiment, we applied PCA only to numerical variables, and categorical variables were preserved separately through One-hot encoding. This allows us to consider both the linear transformation characteristics of PCA and categorical information loss at the same time.

iii. Analysis of Results

Number of PCA dimensions (n)	10	20	30	35
Accuracy (%)	54.63	55.86	57.60	57.32

As the number of dimensions of PCA exceeded 10, it could be confirmed that the performance gradually improved, and the highest performance was recorded with an average accuracy of 57.60% at n=30. After that, if the number of dimensions was increased further, the accuracy decreased, and rather, there is a possibility that overfitting or unnecessary information could be added, so n=30 was judged as the optimal dimension. In the case of the logistic regression model, a linear structure is premised, so it can be confirmed that linear organization of high-dimensional numerical data with PCA contributed to the performance improvement. This can be interpreted as a result of confirming the generalization effect of stably maintaining the predictive performance of the test data without being overfitted with the training data.

2.1.2. SHAP

(1) Overview

Initially, variable selection was conducted based simply on importance rankings or correlation measures; however, this approach had inherent limitations, such as ignoring interactions among variables or excessively excluding important features. Consequently, the predictive performance of the model was constrained, and the complex relationships present in the actual data were not adequately captured.

SHAP (SHapley Additive exPlanations) is a method that interprets the contribution of each feature to predictions at the individual data point level. It quantifies not only the independent effect of a single variable on the prediction but also the complex influences arising from interactions with other variables.

Owing to these characteristics, SHAP provides more precise and granular explanatory power compared to traditional feature importance techniques. In particular, by computing feature contributions for each individual data instance, it significantly enhances the transparency and interpretability of the model's decision-making process.

Based on these results, a preprocessing and variable selection strategy was established to exclude redundant or overestimated features and thereby maximize model performance.

(2) Data Processing

i. Categorical Encoding

To preserve the characteristics of two categorical features, different encoding methods were applied to each.

feature	type	encoding
channel	nominal	one-hot encoding
weekday	ordinal	label-encodig

ii. Correlation

To alleviate multicollinearity, variable pairs with a very high correlation coefficient of 0.8 or above were identified. For each such pair, redundant information was reduced by retaining only one variable and removing the other.

	Feature1	Feature2	Correlation
2	n_non_stop_unique_tokens	n_unique_tokens	0.938879
3	average_token_length	n_non_stop_words	0.942168
6	kw_avg_min	kw_max_min	0.943714
7	kw_max_max	kw_min_min	0.859416
9	kw_avg_avg	kw_max_avg	0.824573
12	self_reference_avg_sharess	self_reference_min_shares	0.805721
13	self_reference_avg_sharess	self_reference_max_shares	0.851193

```
drop_features = [ 'n_non_stop_unique_tokens', 'n_non_stop_words',
   'kw_max_min', 'kw_min_min','kw_max_avg',
   'self_reference_min_shares','self_reference_max_shares']
```

iii. Missing Value Imputation

Missing value imputation methods were applied differently according to the distribution characteristics of each variable. The decisions were based on histogram analysis.

For variables with skewed distributions, missing values were replaced using the median, whereas for variables exhibiting approximately normal-like distributions, the mean was used for imputation.

Example Feature	Distribution Summary	Imputation Method
n_tokens_title	Centered around 7–15 words, narrow tails	Median
num_self_hrefs	Concentrated at 0, range 0–120, mild tail	Median
average_token_length	4–5 characters, narrow distribution	Mean
num_keywords	Centered around 4–10, presence of outliers	Median
kw_min_avg	Range 0–3,000, concentration at 0, long tail	Median
abs_title_sentiment_polarity	Range 0–1, steep slope near 0, multimodal	Mean

Missing value imputation methods were applied differently according to the distribution characteristics of each variable. The decisions were based on histogram analysis.

For variables with skewed distributions, missing values were replaced using the median, whereas for variables exhibiting approximately normal-like distributions, the mean was used for imputation.

Missing value imputation methods were applied differently according to the distribution characteristics of each variable. The decisions were based on histogram analysis.

For variables with skewed distributions, missing values were replaced using the median, whereas for variables exhibiting approximately normal-like distributions, the mean was used for imputation.

Missing value imputation methods were applied differently according to the distribution characteristics of each variable. The decisions were based on histogram analysis.

For variables with skewed distributions, missing values were replaced using the median, whereas for variables exhibiting approximately normal-like distributions, the mean was used for imputation.

Missing value imputation methods were applied differently according to the distribution characteristics of each variable. The decisions were based on histogram analysis.

For variables with skewed distributions, missing values were replaced using the median, whereas for variables exhibiting approximately normal-like distributions, the mean was used for imputation.

(2) Feature Selection

i. Model-Based Importance

The predictive contributions of each variable were evaluated using Random Forest and XGBoost models. Both models employed the same train-validation data split and were trained with 100 estimators each. After training, feature importance scores were computed to quantitatively assess the relative impact of each variable on model performance.

These results served as a reference for understanding the relative importance of variables prior to conducting SHAP analysis.

Importances (XGBoost):

	Feature	Importance
37	channel_Entertainment	0.152158
41	channel_Tech	0.077541
40	channel_Social Media	0.055085
14	kw_avg_avg	0.037721
11	kw_max_max	0.037144
43	weekday_encoded	0.030582
15	self_reference_avg_sharess	0.023509
13	kw_min_avg	0.023075
42	channel_World	0.019466
38	channel_Lifestyle	0.019295
18	LDA_02	0.018976
16	LDA_00	0.018965
2	n_unique_tokens	0.018888
28	min_positive_polarity	0.018458

Importances (RandomForest):

	Feature	Importance
14	kw_avg_avg	0.052223
15	self_reference_avg_sharess	0.040379
18	LDA_02	0.040354
17	LDA_01	0.035770
9	kw_avg_min	0.035717
12	kw_avg_max	0.034990
20	LDA_04	0.033987
2	n_unique_tokens	0.033441
16	LDA_00	0.033167
7	average_token_length	0.032072
19	LDA_03	0.031446
21	global_subjectivity	0.031011
1	n_tokens_content	0.030808
27	avg_positive_polarity	0.030340
22	global_sentiment_polarity	0.030194
23	global_rate_positive_words	0.030175

ii. SHAP Importance

A comparative analysis of SHAP values derived from the XGBoost and Random Forest models revealed that a substantial portion of the highly important features overlapped between the two models.

XGBoost SHAP Importance:

	Feature	SHAP Importance
14	kw_avg_avg	0.327341
15	self_reference_avg_sharess	0.209629
43	weekday_encoded	0.195671
13	kw_min_avg	0.156585
16	LDA_00	0.150362
2	n_unique_tokens	0.119120
18	LDA_02	0.109061
12	kw_avg_max	0.108892
17	LDA_01	0.103815
4	num_self_hrefs	0.094167
20	LDA_04	0.093233
11	kw_max_max	0.092693
24	global_rate_negative_words	0.090932

RandomForest SHAP Importance:

	Feature	SHAP Importance
14	kw_avg_avg	0.040434
15	self_reference_avg_sharess	0.031861
18	LDA_02	0.018584
13	kw_min_avg	0.017747
9	kw_avg_min	0.013102
17	LDA_01	0.012876
20	LDA_04	0.012047
43	weekday_encoded	0.011765
16	LDA_00	0.011091
2	n_unique_tokens	0.010967
12	kw_avg_max	0.010497
37	channel_Entertainment	0.010339

Feature selection was conducted focusing on the intersection of the top *n* features.

Based on empirical knowledge, it is common to select approximately 30% of the total features during selection. This proportion strikes a balance between minimizing information loss by not overly reducing the number of variables, and simultaneously removing redundant features to prevent overfitting and enhance interpretability.

Given that the dataset used contains 46 features, selecting around 10 to 14 features was deemed appropriate, leading to candidate values of n = 15 and 20. When n was set to 15, 11 variables were selected; when n was 20, 15 variables were selected. Ultimately, n = 15 was chosen as the optimal value.

This approach focuses on variables that consistently exhibit high importance across both models (XGBoost and Random Forest), aiming to reduce variability between models and the risk of overfitting. Additionally, by selecting a relatively appropriate number of features relative to the total, the model complexity is lowered, and unnecessary noise variables are excluded to improve generalization performance.

Consequently, a final set of 11 reliable core features was selected.

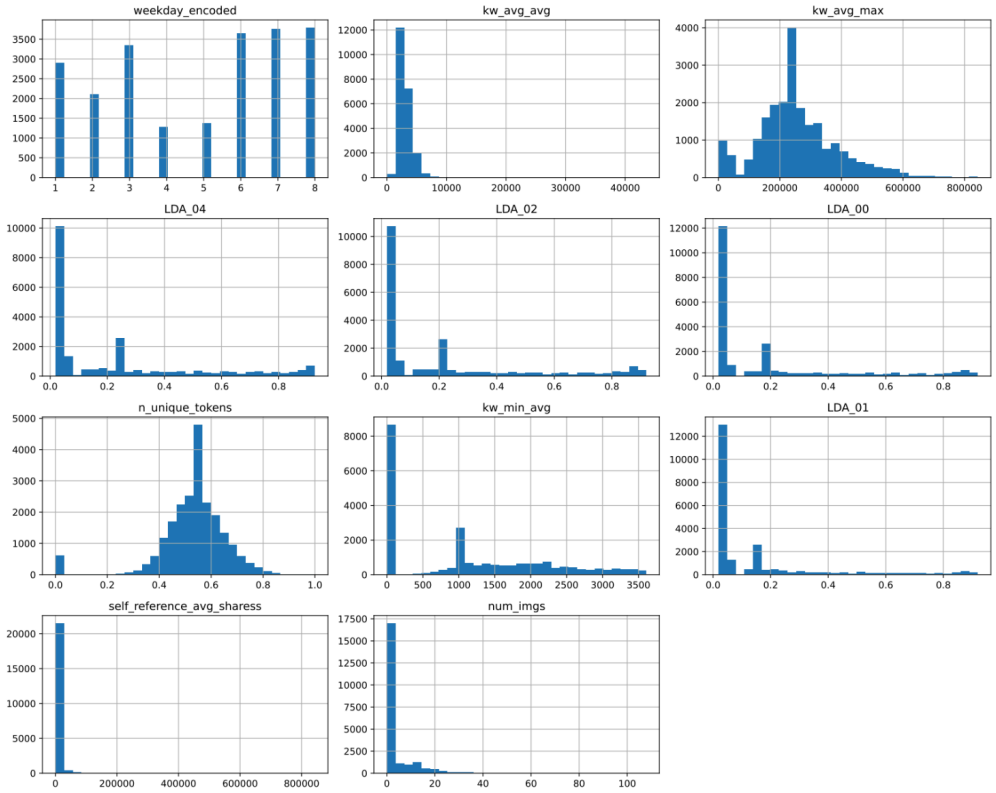
```
top 15:
  {'weekday_encoded', 'kw_avg_avg', 'kw_avg_max',
  'LDA_04', 'LDA_02', 'LDA_00', 'n_unique_tokens', 'kw_min_avg',
  'LDA_01', 'self_reference_avg_sharess', 'num_imgs'}

top 20:
   {'weekday_encoded', 'kw_avg_avg', 'LDA_04', 'kw_avg_max',
   'LDA_03', 'num_imgs', 'LDA_02', 'LDA_00', 'global_subjectivity',
   'kw_avg_min', 'n_unique_tokens', 'kw_min_avg', 'LDA_01',
   'self_reference_avg_sharess', 'channel_Social Media'}
```

(3) Additional Preprocessing

Each variable underwent tailored preprocessing steps according to its distributional characteristics and data properties. Similar to the process prior to feature selection, histograms were utilized, and the following criteria were applied.

Distribution Type	Processing Method	Description
Normal-like	Interquartile Range (IQR) method, Standardization (StandardScaler)	Improves model performance by limiting outlier ranges and applying standardization.
Right-skewed	Log transformation (log1p), clipping at 99th quantile, Standardization (StandardScaler)	Transforms distribution closer to normal and clips extreme values to enhance model stability.
Multimodal	Binary split for zero-value group, log transformation, and clipping	Minimizes information loss and preprocesses variables to fit model requirements based on distribution characteristics.
Categorical & Uniform	Categorical variables: no additional preprocessing; Uniform variables: Standardization (StandardScaler)	Categorical variables are used as-is, while uniform distribution variables are normalized via standardization.



2.2. Feature Refinement

As a result of comparing the performance between the original data and the preprocessed data by adopting the stacking model to be described later, the case where the original data was still used showed better performance. Accordingly, the preprocessing was planned to minimize only variables that could negatively affect the performance of the model while maintaining the structure of the original data as much as possible.

Specifically, a built-in model such as XGBoost was used to calculate the importance of variables and based on this, the feature selection of Top15 was conducted to select the upper features. At the same time, in order to solve the collinearity problem between numerical variables, we intend to try a method of mainly removing variables with serious multicollinearity by using the VIF (Variance Inflation Factor).

2.2.1. top15 + 3 derivatives

In this experiment, the importance of each variable was quantitatively evaluated using XGBoost, a representative tree-based model, and the upper variables were selected based on the results.

(1) Overview

After learning is completed, XGBoost automatically calculates the importance of variables based on the degree to which each feature contributed to the model's predictive performance. Importance can be evaluated by one of three criteria: gain, weight, and cover, and appropriate criteria are selected according to the purpose of the experiment. The gain is calculated by measuring how much the model's performance has improved based on the loss function when the data is divided based on a specific variable. The feature was selected based on gain because it directly reflects the degree to which it contributed to the improvement of the actual predictive performance rather than the simple frequency or range. A high gain means that when the tree branches to the feature, the degree of

impurity decreased the most, that is, the feature is a variable that could be classified much more clearly than other features.

(2) Feature Selection : Criteria for selecting the number of units

As a result of arranging the feature importance (gain basis) of the model in descending order, the top 15 variables correspond to about 30% of all variables. Above all, a boundary point where the contribution to improving predictive performance was rapidly reduced was observed around the 15th place. Therefore, it was determined that the model could learn the core information of the data with only these top 15 variables.

```
top15_features = ['kw_avg_avg', 'data_channel_Entertainment', 'kw_max_avg',
   'data_channel_Tech', 'self_reference_min_shares', 'weekday_Saturday',
   'self_reference_avg_sharess','weekday_Sunday', 'data_channel_Social Media',
   'kw_avg_max','LDA_00', 'LDA_02', 'kw_min_avg', 'kw_avg_min', 'num_hrefs']
```

(3) Experiments and Results

i. Add 3 Derivative Variables

In the case of using only the existing Features, there was a limitation in that interactions, nonlinear relationships, and complex meanings between variables were not revealed. In other words, it was judged that it was difficult to sufficiently reflect the potentially information with only a single variable-centered selection method. Therefore, to compensate for this, a derivative variable containing new information was used through ratio, multiplication, and selection between variables. Unlike simple variance-based dimensionality reduction (PCA) or importance-based selection methods, derivative variables are features directly constructed based on actual domain interpretation and the relationship between variables.

1. keyword_strength_ratio = kw_max_avg + kw_avg_avg
2. img_token_ratio = num_imgs + n_unique_tokens
3. subjectivity_sentiment_mix = global_subjectivity + global_rate_positive_words

Keyword_strength_ratio represents the overall average keyword strength compared to the strongest keyword strength. It can measure the influence of the key keyword of the article. Img_token_ratio can determine whether the article is centered on visual data by the number of unique words compared to the number of images, and finally, subjectivity_sentiment_mix represents how much subjectivity and positive expression appear together.

ii. Experimental Method

Model : XGBoost (n_estimators = 20, max_depth = 2, tree_method = 'hist')

Verification : cross validation score (cv = 3, scoring= “accuracy”)

Variables: Top 10, Top 15, All Variables (pre-selected based on importance)

iii. Analysis of Results

	All variables	Top10 +Derivative variable	Top15 +Derivative variable
Accuracy (3-Fold basis)	0.6354	0.6226	0.6398 (Best Performance)

Based on the gain value calculated through XGBoost, 15 variables that had the most positive effect on model performance were selected. The amount of information increased as it expanded from Top10 to Top15, and by adding three meaningful derivative variables, the accuracy was close to 0.64, it shows higher performance than baseline and 0.63, which used all variables. Therefore, it was possible to quantify the qualitative characteristics of the article by quantifying meaningful characteristics such as text/image density and emotional tone through derivative variables based on the relationship between variables, away from the simple variable selection method.

2.2.2. VIF

(1) Overview

Various preprocessing and feature engineering approaches were previously attempted; however, excessive transformation of the original data resulted in information loss and degradation of model performance. In particular, performing feature selection without adequately considering inter-feature correlations posed risks to model stability and interpretability due to multicollinearity issues. Therefore, an approach was adopted to preserve the inherent characteristics of the original data as much as possible, while selectively removing only the problematic points identified through statistical analysis of the variables.

Special emphasis was placed on diagnosing and resolving multicollinearity among variables. To this end, the Variance Inflation Factor (VIF), a representative diagnostic metric for multicollinearity, was used as a key indicator to identify and eliminate variables negatively impacting the predictive model. VIF quantitatively measures the extent to which each independent variable is linearly correlated with other independent variables in regression analysis.

Additionally, to analyze associations among categorical variables beyond multicollinearity, statistical tests such as the Chi-Square test and Cramér's V were employed. The Chi-Square test assesses the difference between observed and expected frequencies to determine whether a statistically significant association exists between two categorical variables. Cramér's V, calculated based on the Chi-Square test results, quantifies not only the presence but also the strength of the association.

(2) Feature Selection

i. Numerical Variables

First, the Pearson correlation coefficient was calculated to quantitatively assess linear relationships among numerical variables. By examining the correlation matrix, the presence of strong correlations between variables was identified as a preliminary step to explore potential multicollinearity.

```
pearson correlation (abs):
                      feature_1                    feature_2  correlation
462                  kw_max_min                  kw_avg_min     0.943714
134           n_non_stop_words        average_token_length     0.942168
88             n_unique_tokens     n_non_stop_unique_tokens     0.938879
432                  kw_min_min                  kw_max_max    -0.859416
714   self_reference_max_shares   self_reference_avg_sharess     0.851193
639                  kw_max_avg                  kw_avg_avg     0.824573
691   self_reference_min_shares   self_reference_avg_sharess     0.805721
```

High VIF values indicate that a variable has a strong linear relationship with other variables. Typically, a VIF exceeding 10 is considered indicative of severe multicollinearity, and variables with VIF values above this threshold were considered candidates for removal. Some LDA variables exhibited extremely high VIF values, approaching infinity.

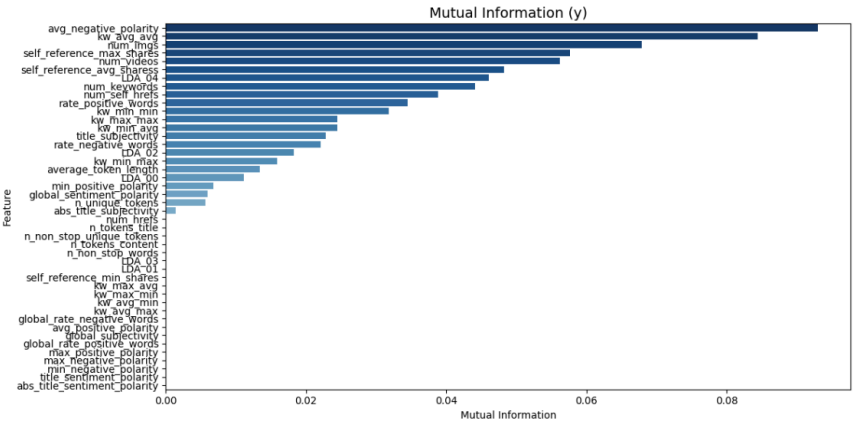
```
VIF = inf : ['LDA_01', 'LDA_04', 'LDA_00', 'LDA_03', 'LDA_02']

VIF top 30 (inf removed):
                         feature           VIF
33           rate_positive_words  9.007199e+15
34           rate_negative_words  9.007199e+15
4              n_non_stop_words  9.007199e+15
23    self_reference_avg_sharess  3.799800e+01
14                   kw_avg_min  3.545584e+01
3              n_unique_tokens  3.453385e+01
13                   kw_max_min  3.400238e+01
22     self_reference_max_shares  2.327864e+01
5       n_non_stop_unique_tokens  2.077674e+01
20                   kw_avg_avg  1.747950e+01
10         average_token_length  1.133808e+01
```

Variables sharing similar information were grouped, and within each group, variables exhibiting strong correlations with a Pearson correlation coefficient of 0.8 or higher were identified as candidates for removal. The priority for variable removal within each group was determined by referencing variable importance measured via Mutual Information.

Variable Group	Variables Removed	Variables Retained
LDA	LDA_00, LDA_01, LDA_02, LDA_03	LDA_04
rate_words	rate_negative_words	rate_positive_words
n_non_stop_tokens	n_non_stop_words	n_non_stop_unique_words
self_reference_shares	self_reference_min_shares	self_reference_max_shares
kw	kw_max_min, kw_avg_avg	kw_avg_min, kw_avg_avg

```
drop_features = ['LDA_00', 'LDA_01', 'LDA_02', 'LDA_03',
   'rate_negative_words','n_unique_tokens', 'kw_max_min', 'kw_avg_avg',
   'n_non_stop_words', 'self_reference_avg_sharess']
```



The subsequently recalculated VIF results showed that all variables had VIF values below 10, confirming that the multicollinearity issue was effectively mitigated.

ii. Categorical Variables

For each pair of categorical variables, a crosstabulation table was created, and the Chi-Square statistic and p-value were calculated. A smaller p-value indicates that the two variables are not independent and suggests a statistically significant association.

Cramér's V is a measure derived from the Chi-Square statistic that expresses the strength of association between two nominal variables as a value between 0 and 1.

```
categorical (Chi-square + Cramér's V):
      var1          var2  p_value  cramers_v
0  weekday  data_channel      0.0     0.0432
```

The p-value was extremely low, effectively resulting in 0.0, indicating a statistically significant association between the two variables. However, the Cramér's V value was 0.0432, which is low, suggesting a weak strength of association; therefore, it was decided to retain both categorical variables.

(3) Additional Preprocessing

After removing the variables selected in (2)-i, additional preprocessing was performed using the same method as applied in section 2.1.1. The following are some examples of this process.

Example Feature	Distribution Summary	Missing Value Imputation Method
n_tokens_content	Right-skewed, long tail	log1p transformation + clipping + StandardScaler
n_non_stop_unique_tokens	Bimodal with zero spike	Binary split for zero/non-zero + StandardScaler
num_hrefs	Right-skewed, long tail	log1p transformation + clipping + StandardScaler

2.3 Final Feature Set Construction

The results from section 2.2 were positive, with the greatest improvement in model performance observed when derived variables were added. It was confirmed that predictive accuracy was highest when derived variables, reflecting relationships among variables, were considered alongside the top-ranked variables, rather than using only the highest importance variables. This approach effectively addressed the limitations inherent in single-variable-focused methods.

Accordingly, the final dataset was constructed by removing variables based on VIF analysis, including all Top 15 variables, and adding 11 newly created derived variables.

2.3.1. FullAugmentedSet_ver1

(1) Overview

Eleven new derived variables were introduced, designed to reflect meaningful relationships rather than simple combinations of existing variables. These variables capture quantitative characteristics such as lexical diversity relative to content length, keyword concentration, differences between sentiments, and levels of emotional emphasis. Additionally, they incorporate aspects of expressiveness and information density. Notably, by introducing ratios, differences, and interactive relationships not directly evident in the original features, these derived variables contributed to enhancing predictive performance. The key added derived variables are as follows.

Variable Name	Description
token_density	Density of keywords relative to content length

sentiment_ratio	Ratio of positive to negative sentiment
emotion_contrast	Polarity difference between positive and negative sentiment
lda_entropy	Entropy of LDA topic distribution (topic uncertainty)
kw_maxmax_per_keyword	Ratio of maximum intensity to number of keywords
unique_token_ratio	Indicator of lexical diversity
ref_share_ratio	Ratio of maximum to average share counts
share_per_token	Average shares relative to content length
kw_variety_ratio	Keyword diversity ratio (maximum/minimum)
positive_minus_negative	Sentiment difference
sentiment_weighted_subjectivity	Emotional emphasis indicator combining subjectivity and sentiment intensity

(2) Feature Selection

Variance Inflation Factor (VIF) analysis was applied to the entire dataset, including derived variables, to eliminate multicollinearity. Initially, one variable was removed from each pair exhibiting a correlation coefficient greater than 0.9. Subsequently, variables with VIF values exceeding 10 were further removed to reduce the risk of overfitting and enhance interpretability. This selection process eliminated redundant information and variables with strong linear correlations while preserving the core information represented by the derived variables. Consequently, the final dataset represents a balanced composition of original variables, importance-based variables, and derived variables.

(3) Additional Preprocessing

During the derived variable creation process, a small constant of 1e-5 was added to denominators where there was a risk of zero values to ensure numerical stability. For the generated variables, infinite (inf) and NaN values were handled by imputing missing values with the mean of all numerical variables.

The same derived variable creation and feature selection procedures were applied to the test dataset. Missing value imputation in the test data was performed using the mean values calculated from the training set.

2.3.2. FullAugmentedSet_ver2

(1) Overview

Although the results from FullAugmentedSet_ver1 achieved the best performance, concerns about overfitting led to the creation of a revised dataset. Rather than merging all three datasets and addressing multicollinearity via VIF, it was decided to base the dataset on a single set and supplement it with missing information. Subsequently, 11 derived variables based on the original features created thus far were added, followed by post-processing to complete the dataset.

(2) Feature Selection

i. Base Dataset

Rather than applying identical preprocessing uniformly, it was determined that making decisions based on histogram analysis would be more effective for model training. Therefore, the VIF-based dataset was selected as the reference, and all variables from the Top 15 were included, excluding duplicates.

ii. Derivative Variables

After merging the datasets, 11 derived variables based on the original features created thus far were added. Using VIF and correlation analysis, multicollinearity issues were addressed, resulting in an enhanced dataset that builds upon section 2.3.1.

iii. Addressing Multicollinearity

To address multicollinearity among variables during model training, analyses were conducted based on the Variance Inflation Factor (VIF) and Pearson correlation coefficient. Variables with VIF values exceeding 10 or correlation coefficients of 0.9 or higher were considered candidates for removal. Variables exhibiting perfect multicollinearity(”완전다중공선성”) showed VIF values approaching infinity.

After removing one variable from each correlated group, the remaining variables exceeding the VIF threshold were kw_max_max and emotion_contrast. However, as the removal of positive_minus_negative resolved the multicollinearity issue involving emotion_contrast, it was preliminarily determined that emotion_contrast did not require removal.

```
corr ≥ 0.9:
emotion_contrast          positive_minus_negative           1.000000
global_sentiment_polarity  sentiment_weighted_subjectivity    0.913691

VIF > 10
['kw_max_max', 'global_sentiment_polarity', 'emotion_contrast', 'positive_minus_negative']
완전다중공선성: ['emotion_contrast', 'positive_minus_negative']
```

```
drop_features = ['kw_max_max', 'positive_minus_negative',
   'global_sentiment_polarity']
```

After variable removal, VIF and correlation were re-evaluated, and the derived variable emotion_contrast was found to still exceed a VIF value of 10; therefore, it was removed.

```
VIF > 10
['emotion_contrast']
```

```
corr ≥ 0.9
Series([], dtype: float64)
```

(3) Additional Preprocessing

Preprocessing based on histograms was performed using the same approach as applied in the VIF analysis. The following are some examples of this process.

Example Feature	Distribution Summary	Imputation Method
kw_avg_avg	Right-skewed	log1p transformation, clipping, scaling
self_reference_avg_sharess	Right-skewed	log1p transformation, clipping, scaling
LDA	Right-skewed (extreme)	Not applied (log transform risks excessive clustering)

3.1 Baseline Model Selection

3.1.1 Linear Model Performance Evaluation: Logistic Regression and SVM

(1) Experimental Background and Evaluation Method

In the initial modeling phase, experiments were conducted using representative linear classification models, Logistic Regression and Support Vector Machine (SVM), to ascertain the fundamental classify-ability of the data.
The evaluation was performed on both the SHAP-based dataset and a dataset dimensionally reduced via PCA, each using a 5-fold cross-validation method.

(2) Logistic Regression Model Results

On the SHAP dataset, the model achieved a 5-fold mean accuracy of 0.5398 (standard deviation 0.0434).
On the PCA dataset, the mean accuracy was 0.5439 (standard deviation 0.0487).

(3) SVM Model Results

On the SHAP dataset, the mean accuracy was 0.5031 (standard deviation 0.0045).
On the PCA dataset, the mean accuracy was 0.5030 (standard deviation 0.0044).

(4) Analysis

The experimental results showed that while SVM appeared to make stable predictions due to a very small standard deviation in performance across cross-validation folds, its mean accuracy remained at approximately 0.503, indicating performance no better than random guessing.
Logistic Regression exhibited slightly higher accuracy compared to SVM, but still demonstrated low performance around 0.54, and on some datasets, it also showed relatively large variance between folds.

(5) Conclusion

As both linear models generally showed low predictive accuracy, it was concluded that these models are insufficient for effectively learning the complex characteristics of the given dataset.

3.1.2 Performance Evaluation of Tree-Based Models: RandomForest and XGBoost

[Performance Comparison of XGBoost and RandomForest Models Based on PCA/SHAP/Original Datasets]

(1) Experimental Background and Evaluation Method

In the previous SVM and logistic regression experiments, the accuracy was only in the low 0.5 range, indicating poor performance. This suggests that the characteristics of the dataset are difficult to distinguish effectively using linear classifiers, leading to the conclusion that a model capable of learning more nonlinear and complex decision boundaries is needed.

Therefore, we decided to establish a new baseline using representative nonlinear models: the tree-based model RandomForest and the gradient boosting model XGBoost. RandomForest combines multiple decision trees, which helps prevent overfitting while allowing for high predictive performance. XGBoost, based on gradient descent boosting, is expected to capture interactions between variables and nonlinearities more effectively than previous models, thus making it suitable as an additional baseline.

(2) Model Results (PCA, SHAP, Origin)

Accordingly, the three datasets (SHAP-based dataset, PCA-based dataset, and the original dataset with simple missing value treatment and one-hot encoding) were each applied to RandomForest and XGBoost, and 5-fold accuracy was measured. As shown in the table below, both models achieved the highest performance on the original dataset, indicating that further data preprocessing and feature extraction are necessary.

(3) Model Results (Top15, VIF)

Considering that the existing preprocessed datasets did not show a significant improvement in performance compared to the original dataset, additional feature engineering was conducted to create new datasets: a VIF-based dataset and a top 15 feature dataset based on XGB gain. These datasets were then applied to RandomForest and XGBoost using the same evaluation method. The experimental results showed that the newly constructed datasets performed similarly to the original dataset overall, but better than the SHAP and PCA-based datasets, confirming that the new datasets can have a positive impact on model performance.

<5-fold Avg Acc(std)>

	SHAP	PCA	top15+var3	VIF	origin
RandomForest	0.6308 (0.0049)	0.6314(0.0056)	0.6440 (0.0055)	0.6424 (0.0075)	0.6501 (0.0093)
XGBoost	0.6267(0.0072)	0.6128(0.0097)	0.6377 (0.0087)	0.6256 (0.0076)	0.6372 (0.0057)

(4) Analysis

Overall, the highest performance was still observed with the original (origin) dataset. This suggests that excessive preprocessing or dimensionality reduction may have hindered model training, and that the raw information and interactions between variables required by the model were better preserved in the original data.

In contrast, the feature sets based on Top15 and VIF showed significantly better performance than those based on SHAP and PCA, confirming that more refined feature selection strategies can play an important role in improving model performance. In particular, the Top15 feature set, which includes engineered features, demonstrated stable performance with both XGBoost and RandomForest, providing empirical evidence of the effectiveness of meaning-based derived features over individual raw features.

(5) Conclusion

Through this experiment, tree-based models (RandomForest, XGBoost) were shown to overcome the limitations of linear models and generally achieve superior classification performance. The fact that the original dataset recorded the highest accuracy suggests that excessive preprocessing or feature reduction can lead to the loss of important information. Moreover, the Top15 and VIF-based datasets outperformed the SHAP and PCA-based datasets, and the addition of derived features positively contributed to model performance. These results confirm that feature construction that prioritizes information preservation and the expansion of meaningful variables play a crucial role in enhancing model performance.

3.2 Ensemble Model Construction and Optimization

3.2.1 Comparison of Ensemble Techniques: Voting vs. Stacking

(1) Experimental Background and Evaluation Method

In the previous experiments, individual models (RandomForest, XGBoost) were applied to each dataset (VIF, Top15, Origin), and the average accuracy generally converged to around 0.64 (or lower). This suggests that despite tree-based models effectively capturing interactions between variables and nonlinearity, a single model has limitations in improving predictive performance.

Accordingly, in the next stage, we applied ensemble techniques that can improve performance by combining the prediction results of multiple models. In this experiment, we used two representative ensemble methods: VotingClassifier and StackingClassifier, each configured as follows:

- VotingClassifier: Combines the predictions of RandomForest, XGBoost, LightGBM, and CatBoost using hard voting (majority rule)
- StackingClassifier: Uses the same set of base models, with their predictions fed into a meta-model (Logistic Regression) to make the final prediction

Model performance was evaluated using 5-Fold Cross Validation on three datasets (VIF, Top15 + 3 derived features, Origin).

(2) Model Results (Voting)

As a result of applying the Voting method, we observed improved average accuracy across all datasets compared to the individual models. In particular, the Voting-based average accuracy for the Origin dataset was the highest at 0.6577, and the Top15 + derived3 and VIF datasets also showed stable performance with accuracies of 0.6519 and 0.6511, respectively. This demonstrates that the majority voting structure offset the bias of individual models to some extent and contributed to improved prediction consistency.

(3) Model Results (Stacking)

The Stacking method achieved higher overall accuracy and lower standard deviation than Voting. All three datasets outperformed Voting in terms of performance, with the Origin dataset showing the best results at an average accuracy of 0.6614 (standard deviation 0.0066). The Top15 + derived3 and VIF datasets recorded accuracies of 0.6542 and 0.6568, respectively, confirming that Stacking outperformed Voting in both performance improvement and prediction stability across all configurations.

(4) Analysis

From this experiment, the following findings can be drawn. First, when ensemble techniques were applied, performance improved overall across all datasets compared to the previous individual models. This implies that by combining the predictions of individual tree models, their respective blind spots were complemented, enabling more consistent predictions. Second, while both the Voting and Stacking methods showed meaningful improvements in performance, Stacking generally recorded higher accuracy and lower standard deviation, confirming its more stable and predictive structure. Third, the Origin dataset, which previously had the best performance, continued to deliver excellent results, and the other datasets also showed similar performance when ensemble methods were applied, confirming the potential for performance improvement.

(5) Conclusion

Through this experiment, we demonstrated that ensemble techniques are more effective for the

online news article classification problem than single tree models. In particular, the Stacking method, which combines various tree-based models, showed the highest performance and stability, proving its suitability as the final classifier. Therefore, for this task, the Stacking-based ensemble structure was judged to be a better choice than the Voting method.

<5-fold Avg Acc(std)>

	top15+var3	VIF	origin
Voting	0.6519 (0.0091)	0.6511 (0.0078)	0.6577 (0.0081)
Stacking	0.6542 (0.0079)	0.6568 (0.0082)	0.6614 (0.0066)

3.2.2 Performance Analysis of Stacking Models by Dataset

(1) Experimental Purpose and Background

Previous experiments confirmed the possibility that Stacking ensemble models could exhibit better performance than individual models. However, the performance of a Stacking model heavily depends on the types and hyperparameters of the base models used, as well as the configuration of the meta-model that performs the final prediction. Furthermore, the model's performance can vary based on the characteristics of the input dataset.

Therefore, the objectives of this experiment are as follows:

- **Hyperparameter Optimization**

  To perform a Grid Search for the main hyperparameters of the Stacking model to find the optimal combination. Due to time constraints, hyperparameters were first explored based on the Original dataset, which was considered to contain the richest information.

- **Performance Comparison by Dataset**

  To apply the Stacking model with fixed, optimized hyperparameters to various datasets that have undergone different preprocessing and feature engineering techniques. This aims to compare and evaluate which dataset configuration yields the best performance, thereby validating the effectiveness of data preprocessing and feature selection strategies.

- **Analysis of Normalization and Regularization Impact**

  To analyze the impact of data normalization (StandardScaler) application and L2 Regularization on model performance and stability.

(2) Experimental Design

i. **Reference Dataset (for hyperparameter tuning)**

Original dataset (dataset with missing values treated as "unknown" and categorical variables one-hot encoded).

ii. Applied Datasets (for performance comparison)

- **Original**: Missing values filled with "unknown," and categorical variables one-hot encoded.
- **Basic**: Missing values replaced with the median, and categorical variables one-hot encoded.
- **top15+var3**: Dataset using the top 15 important variables and 3 derived variables.
- **top15+var7**: Dataset using the top 15 important variables and 7 derived variables.
- **VIF**: Dataset with variables exhibiting high multicollinearity removed based on VIF (Variance Inflation Factor).

- **VIF+var4**: VIF-based dataset with 4 derived variables added.
- **original+VIF+top15**: Dataset combining information from Original, VIF, and top15 datasets.
- **interactions**: Dataset considering interaction terms between variables.

iii. Model

StackingClassifier

Base Models: RandomForest, XGBoost, LightGBM, CatBoost

Meta Model: Logistic Regression)

iv. Hyperparameters (Grid Search Results)

- catb_depth: 4
- final_estimator_C: 0.1 (Related to L2 regularization of Logistic Regression; smaller values imply stronger regularization)
- lgbm_num_leaves: 31
- passthrough: False
- rf_max_depth: 10
- xgb_learning_rate: 0.05

v. Evaluation Metrics

- 5-Fold cross-validation mean accuracy and standard deviation (Std).
- Accuracy, F1 Score, ROC AUC Score on the training dataset.

vi. Comparison Conditions

- Application of Normalization (using StandardScaler).
- Application of L2 Regularization (xgb_l2=1.0, lgbm_l2=0.1, catb_l2=3.0).

(3) Experimental Results and Analysis

i. **Hyperparameter Optimization and Baseline Performance (Original Dataset)**

As a result of conducting a Grid Search using the Original dataset, the hyperparameter combination mentioned above was determined to be optimal. The 5-Fold cross-validation results for the Stacking model applying these hyperparameters were as follows

- **Without Normalization:** Mean Accuracy 0.6610, Standard Deviation 0.0049.
- **With Normalization:** Mean Accuracy 0.6624, Standard Deviation 0.0070.

Applying normalization slightly increased the mean accuracy but also tended to increase the standard deviation somewhat.

The training performance on the entire training data without normalization and regularization showed an Accuracy of 0.7484, F1 Score of 0.7445, and ROC AUC of 0.8302.

After applying L2 Regularization, the Accuracy slightly decreased to 0.7400, F1 Score to 0.7370, and ROC AUC to 0.8226, suggesting that for this dataset, regularization did not significantly impact overfitting prevention or might have even caused a slight performance degradation.

ii. Comparison of Stacking Model Performance on Various Datasets (Without Normalization)

The 5-Fold cross-validation mean accuracies for the Stacking model (using optimized hyperparameters and without normalization) applied to each dataset are as follows:

Dataset	Mean Accuracy (5-Fold CV)	Std Dev (5-Fold CV)	Overall Training Accuracy	Overall Training F1	Overall Training ROC AUC
1. Original	0.6610	0.0049	0.7484	0.7445	0.8302
2. Basic	**0.6813**	0.0057	0.7483	0.7448	0.8305
3. top15+var3	0.6548	0.0062	0.7241	0.7213	0.8029
4. top15+var7	0.6561	0.0075	0.7311	0.7277	0.8129
5. VIF	0.6579	0.0063	0.7459	0.7429	0.8254
6. VIF+var4	0.6545	0.0112	0.7505	0.7476	0.8303
7. original+VIF+top15	0.6634	0.0049	0.7518	0.7515	0.8343
8. interactions	0.6634	0.0049	0.7545	0.7515	0.8343

Key Observations:

- **Superiority of the Basic Dataset**

  The Basic dataset, where missing values were replaced by the median, showed the highest performance with a 5-Fold cross-validation mean accuracy of 0.6813. This suggests that using specific statistical values for missing value imputation might be more suitable for this model and data characteristics than filling with "unknown."

- **Limitations of Feature Engineering**

  The top15-related datasets (top15+var3, top15+var7) showed somewhat lower cross-validation performance compared to the Original or Basic datasets. This could mean that some useful information was lost during variable selection, or the added derived variables did not significantly contribute to model performance. However, it should be considered that these hyperparameters were tuned for the Original dataset and might not be optimal for other datasets.

- **Effect of Data Combination**

  The dataset combining Original, VIF, and top15 information, and the interactions dataset, showed higher cross-validation performance (0.6634) than the Original dataset, indicating that integrating various information sources can be positive for performance improvement.

- **Impact of Normalization**

  For most datasets, the difference in cross-validation accuracy with or without normalization was not significant, and no consistent trend was observed (e.g., Original, Basic, top15+var3, VIF, etc.). This suggests that data scaling might have a relatively small impact due to the nature of Stacking ensembles, which are predominantly composed of tree-based models.

- **Impact of Regularization**

  Looking at the performance on the entire training data, applying L2 Regularization mostly resulted in minimal performance degradation or almost no change. This implies that either overfitting is not severe with the current model configuration and data, or the applied regularization strength might not be optimal.

(4) Conclusions and Recommendations

Through this experiment, applying Stacking model hyperparameters optimized for the Original dataset to various datasets, the highest cross-validation accuracy (0.6813) was achieved with the Basic dataset, where missing values were imputed with the median. This demonstrates that the method of handling missing values can significantly impact model performance.
Furthermore, the <original+VIF+top15> dataset and the <interactions>dataset, which combined various information sources, also showed respectable performance, confirming the potential of data integration strategies. Conversely, datasets with variables reduced based on specific criteria (top15 related datasets) showed relatively lower performance.

(5) Limitations

- The hyperparameters in this experiment were optimized only for the Original dataset. Therefore, to fully leverage the characteristics of each dataset, hyperparameter tuning for individual datasets is necessary.
- Overfitting control and performance improvement could be pursued through a more detailed Grid Search for regularization strength.

3.3 Construction of a New Dataset Based on Derived Features

In-Depth Dataset Analysis and the Need for Constructing a New Dataset Based on Derived Features

In previous experiments, when fixed hyperparameters optimized for the original dataset were applied to various derived datasets, the 'Basic' dataset—created by replacing missing values with the median—showed the best performance with a cross-validation accuracy of 0.6813. This indicates that data preprocessing, particularly the method of handling missing values, significantly impacts model performance. However, other datasets based on feature engineering did not exhibit a clear performance advantage over the 'Basic' dataset under the same fixed hyperparameters.

This implies that existing feature selection or simple preprocessing alone has limitations in improving model performance, and highlights the need to more actively leverage the potential information within the data. Therefore, we decided to enhance the quality of the dataset itself by discovering new insights that can boost model predictive power and generating more meaningful derived features through domain knowledge or data analysis. As a result of this effort, we constructed new datasets such as the fullAugmentedSet, which includes new features created through combinations or transformations of existing variables.

[Performance Analysis of Models Based on fullAugmentedSet]

In the following experiment, we applied ensemble techniques once again to the fullAugmentedSet dataset, which was constructed by combining the previously used origin, VIF, and top15 datasets, adding domain-based derived features, and applying feature selection based on correlation and VIF criteria. This dataset was refined to minimize information loss and include only variables meaningful for

prediction; details of this process are described in the data section. Based on this dataset, both VotingClassifier and StackingClassifier were evaluated using the same method through 5-fold cross-validation.

(1) Experimental Background and Evaluation Method

In previous experiments, each individual feature set (origin, VIF, top15) was used separately for modeling, but the results showed limitations in achieving further performance improvements. Therefore, in this experiment, to minimize information loss and maximize the complementary synergy between variables, we constructed a final integrated feature set (fullAugmentedSet) by merging the three feature sets (origin, VIF, top15), adding 11 domain knowledge-based derived features, and applying feature selection based on correlation coefficients and VIF criteria. Based on this dataset, VotingClassifier and StackingClassifier were applied in the same manner and evaluated using 5-Fold Cross Validation.

(2) Model Results (Voting Method)

In the case of the Voting method, the experiment using the fullAugmentedSet showed an average accuracy of 0.6586 (standard deviation 0.0059), slightly exceeding the previous performance based on the origin dataset (0.6577), and thus recording the highest accuracy so far.

(3) Model Results (Stacking Method)

For the Stacking method, a more noticeable performance improvement was observed, with an average accuracy of 0.6630 (standard deviation 0.0074), which turned out to be the best result among all experiments conducted.

(4) Analysis

With this dataset, for the first time, performance surpassed that of the origin dataset. Especially with the Stacking method, an average accuracy of 0.6630 was recorded, marking the best result across all experiments. This can be interpreted as the outcome of effectively integrating useful information from various datasets, adding domain-based derived features, and then eliminating unnecessary variables to minimize information loss while capturing only the key variables that contribute most to prediction.

(5) Conclusion

Through this, we were able to achieve performance improvements that were difficult to obtain with single models or simple preprocessing alone, by combining sophisticated feature construction with ensemble strategies. In particular, since the Stacking method consistently outperformed the Voting method, it was selected as the final model, and the next step was to proceed with tuning the Stacking model.

<Comparison of Experimental Results for fullAugmentedSet>

	top15+derived3	VIF	origin	fullAugmentedSet
Voting	0.6519 (0.0091)	0.6511 (0.0078)	0.6577 (0.0081)	0.6586 (0.0059)
Stacking	0.6542 (0.0079)	0.6568 (0.0082)	0.6614 (0.0066)	0.6630 (0.0074)

3.4 Hyperparameter Tuning for the Stacking Model

[F1-Score Improvement Experiment Based on Threshold Adjustment]

(1) Experimental Background and Evaluation Method

Although the previously constructed fullAugmentedSet—created by merging three datasets—achieved high performance by encompassing diverse information, it also raised the possibility of overfitting due to redundant information. Therefore, we constructed a final feature selection dataset (train_fin1) by removing unnecessary redundant variables and retaining only those meaningful for prediction. This dataset was refined with a focus on improving the generalization performance of the model, based on correlations and VIF among all variables.

In this experiment, based on the final dataset, we used not only accuracy but also F1-score, which considers the balance between precision and recall, as well as AUC, which reflects the overall classification performance. During the interim evaluation, it was confirmed that the team's classification model (merged dataset + basic stacking) achieved a relatively low F1-score of approximately 0.63, indicating that the model was insufficiently capturing the positive class. To address this issue, we attempted to find the optimal threshold for the stacking model by averaging F1, accuracy, and AUC as a composite performance metric.

The method used was a StackingClassifier based on soft voting. Unlike hard voting, soft voting averages the class probability outputs from base models to make the final prediction, allowing for flexible threshold adjustment to reflect various trade-offs. This provides a more adaptable decision boundary than hard voting, which classifies strictly based on a 0.5 cutoff, and is especially effective in imbalanced datasets or when improvement in F1-score is desired.

In the experiment, the threshold value was varied from 0.35 to 0.49 in increments of 0.02. For each threshold, 5-fold cross-validation was performed, and F1-score, accuracy, and AUC were measured for each fold. The average of these values was calculated to evaluate performance. Finally, the average of F1, accuracy, and AUC was defined as the mean_score, and the threshold value with the best performance based on this score was selected.

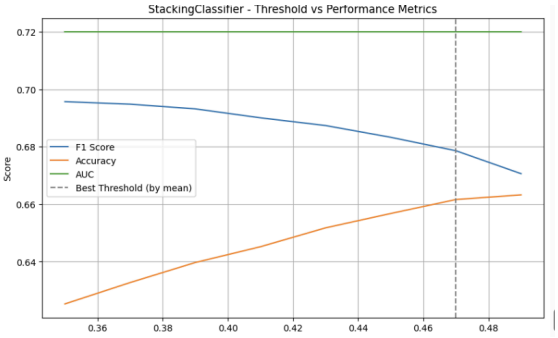
(2) Model Results

As a result of the experiment, the highest mean_score was recorded at threshold = 0.47. At this threshold, the F1-score was 0.6786, accuracy was 0.6616, and AUC was 0.7201, showing relatively stable performance across all three metrics.

<Threshold Experiment – Based on Average of AUC, F1, and Accuracy>

StackingClassifier 성능 요약 (Top 5 by Mean Score):

	threshold	f1	accuracy	auc	mean_score
6	0.47	0.678620	0.661622	0.720091	0.686778
5	0.45	0.683299	0.656757	0.720091	0.686716
4	0.43	0.687390	0.651757	0.720091	0.686413
3	0.41	0.690083	0.645180	0.720091	0.685118
7	0.49	0.670632	0.663243	0.720091	0.684655



(3) Analysis

This suggests that rather than fixing the threshold at 0.5, adjusting it based on the characteristics of the data and model can lead to slight performance improvements. Additionally, visualizations

revealed that F1-score and accuracy moved in opposite directions depending on the threshold, indicating a trade-off relationship, while AUC remained nearly constant.

(4) Conclusion

This experiment demonstrated that selecting the threshold based on a composite metric—the average of three indicators (F1, accuracy, AUC)—rather than a single metric like accuracy is a valid approach. It also showed that threshold adjustment is an effective post-processing technique for more finely tuning model performance.

[Performance Comparison of Final Estimators]

(1) Experimental Background and Evaluation Method

The previous experiment identified 0.47 as the optimal threshold for the StackingClassifier and confirmed that using this threshold led to stable performance improvements. Based on this result, the next step was to experiment with the final estimator, one of the components that most significantly affect final prediction performance.

In a Stacking structure, the final estimator (meta model) takes the prediction outputs of the base models as input and makes the final decision. It is a key component that determines overall performance. Therefore, we fixed the threshold at 0.47 and evaluated performance using various candidate models as the final estimator under the same conditions (5-fold CV, soft voting).

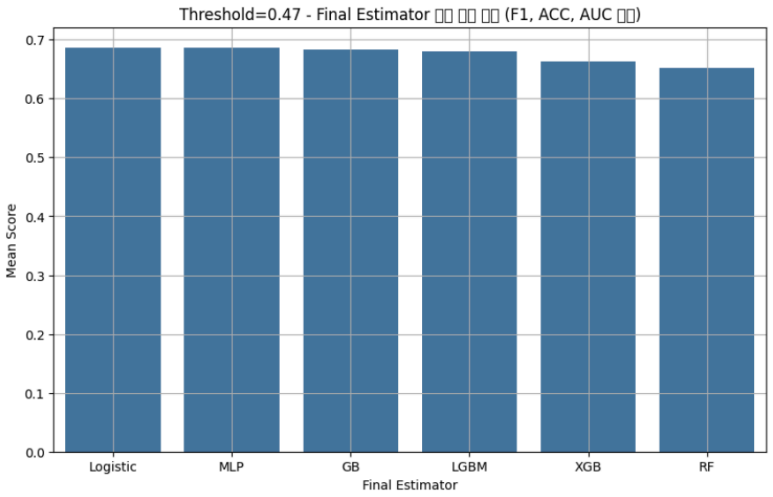
The candidate final estimators included: Logistic Regression, MLP (Multi-Layer Perceptron), Gradient Boosting, LGBM, XGBoost, and Random Forest. All candidates used the same set of base models (RandomForest, XGBoost, LGBM, CatBoost), and binary classification was performed using the soft voting approach based on prediction probabilities with the threshold set to 0.47.

(2) Model Results

Based on the average performance over 5 folds for each model, the final rankings were determined according to the mean_score metric. The experimental results are as follows:

<Results of Final Estimator Experiments>

	final_estimator	f1	accuracy	auc	mean_score
0	Logistic	0.678620	0.661622	0.720091	0.686778
2	MLP	0.675648	0.661081	0.719970	0.685566
1	GB	0.672930	0.660180	0.718670	0.683927
4	LGBM	0.666774	0.657117	0.714242	0.679378
5	XGB	0.653312	0.641712	0.694409	0.663144
3	RF	0.645331	0.629550	0.680471	0.651784



Thus, it was confirmed that the previously used Logistic Regression model achieved the highest performance.

(3) Analysis

This suggests that rather than a complex model, a simple linear classifier can more effectively aggregate and generalize the prediction patterns of the base models. Accordingly, the Logistic Regression model was selected as the final estimator for the final stacking structure.

(4) Conclusion

As a result, the final model configuration for this project was determined as follows:

- **Base Models**: RandomForest, XGBoost, LGBM, CatBoost
- **Final Estimator**: Logistic Regression
- **Prediction Strategy**: Soft Voting with Threshold = 0.47
- **Performance Metric Basis**: Average of F1, Accuracy, and AUC (mean_score)

To maximize the potential of the newly constructed datasets, hyperparameter optimization for the Stacking model was conducted via Grid Search, following the recommendations from previous experiments. Although this process is time-consuming due to the diverse components and extensive search space of the Stacking model, it was deemed essential for achieving optimal performance.

3.4.1 Grid Search based on FullAugmentedSet_ver1 Dataset

In order to fully harness the potential of the newly constructed dataset, we performed hyperparameter optimization for a Stacking model specifically tailored to this dataset using a Grid Search approach, as recommended by prior experiments. Although the Stacking model comprises diverse components and its search space is extensive—requiring a considerable amount of time—we judged this step to be indispensable for securing optimal performance.

(1) Tuning Objectives

To identify the optimal hyperparameter combination for the StackingClassifier using the FullAugmentedSet_ver1 dataset.

(2) Optimal Hyperparameters

- catb__depth: 4
- catb__iterations: 100
- final_estimator__C: 0.1 (L2 Regularization Strength of Logistic Regression)
- lgbm__n_estimators: 100
- lgbm__num_leaves: 31
- passthrough: False
- rf__max_depth: 10
- rf__n_estimators: 100
- xgb__learning_rate: 0.05
- xgb__n_estimators: 100

(3) Final Performance Evaluation of the Optimal Model (Threshold = 0.47, Soft Voting, 5-Fold CV)

The Stacking model, configured with the optimal hyperparameters derived from Grid Search, was evaluated using 5-Fold Cross-Validation with a threshold of 0.47 and Soft Voting, which proved effective in prior experiments. The results are as follows:

Fold	Accuracy	F1-Score	AUC
1	0.7543	0.7633	0.8363
2	0.7514	0.7619	0.8406
3	0.7349	0.7466	0.8237
4	0.7565	0.7654	0.8428
5	0.7511	0.7596	0.8336
Mean	**0.7496**	**0.7593**	**0.8354**

Final Mean Score (Average of Accuracy, F1-Score, and AUC): 0.7815

(4) Conclusion

The hyperparameter tuning for the FullAugmentedSet_ver1 dataset resulted in a mean accuracy of 0.7496, an F1-score of 0.7593, and an AUC of 0.8354. The final mean score, averaging these three metrics, was 0.7815. This performance significantly surpassed that of previous experiments (e.g., the Basic dataset's cross-validation accuracy of 0.6813).

3.4.2. Grid Search based on FullAugmentedSet_ver2 Dataset

Following the evaluation of FullAugmentedSet_ver1, Grid Search for Stacking model hyperparameter optimization was similarly performed on another key candidate, the FullAugmentedSet_ver2 dataset.

(1) Tuning Objective

To identify the optimal hyperparameter combination for the StackingClassifier using the FullAugmentedSet_ver2 dataset.

(2) Optimal Hyperparameters

- catb__depth: 4
- catb__iterations: 100
- final_estimator__C: 0.1 (L2 Regularization Strength of Logistic Regression)
- lgbm__n_estimators: 100
- lgbm__num_leaves: 31
- passthrough: False
- rf__max_depth: 10
- rf__n_estimators: 100
- xgb__learning_rate: 0.05
- xgb__n_estimators: 100

(3)Final Performance Evaluation of the Optimal Model (Threshold = 0.47, Soft Voting, 5-Fold CV)

The final performance of the Stacking model, using the optimal hyperparameters derived from Grid Search for the FullAugmentedSet_ver2 dataset, is detailed below. Evaluation was conducted via 5-Fold Cross-Validation with a threshold of 0.47 and Soft Voting.

Fold	Accuracy	F1-Score	AUC
1	0.7518	0.7623	0.8367
2	0.7514	0.7611	0.8386
3	0.7417	0.7522	0.8246
4	0.7579	0.7676	0.8411
5	0.7552	0.7633	0.8356
Mean	**0.7516**	**0.7613**	**0.8353**

Final Mean Score (Average of Accuracy, F1-Score, and AUC): 0.7827

(4) Conclusion

The hyperparameter tuning on the FullAugmentedSet_ver2 dataset yielded a mean accuracy of 0.7516, an F1-score of 0.7613, and an AUC of 0.8353. The final mean score, averaging these three metrics, was 0.7827.

3.4.3. Final Dataset and Hyperparameter Selection

(1) Comparison of Results for Both Datasets

The performance of the Stacking model, derived from Grid Search on FullAugmentedSet_ver1 and FullAugmentedSet_ver2 respectively, is compared below:

Dataset	Mean Accuracy	Mean F1-Score	Mean AUC	Final Mean Score (Avg. of Acc, F1, AUC)
FullAugmentedSet_ver1	0.7496	0.7593	0.8354	0.7815
FullAugmentedSet_ver2	**0.7516**	**0.7613**	0.8353	**0.7827**

(2) Performance Comparison

Both datasets demonstrated excellent performance. However, based on the Final Mean Score, FullAugmentedSet_ver2 (0.7827) slightly outperformed FullAugmentedSet_ver1 (0.7815). The individual metrics of Mean Accuracy and Mean F1-Score were also higher for the FullAugmentedSet_ver2 dataset.

(3) Final Dataset Determination

Therefore, FullAugmentedSet_ver2 is selected as the final dataset for building the project's model. The hyperparameters to be applied are those derived from the Grid Search on FullAugmentedSet_ver2: {'catb__depth': 4, 'catb__iterations': 100, 'final_estimator__C': 0.1, 'lgbm__n_estimators': 100, 'lgbm__num_leaves': 31, 'passthrough': False, 'rf__max_depth': 10, 'rf__n_estimators': 100, 'xgb__learning_rate': 0.05, 'xgb__n_estimators': 100}. The final model will be trained using this dataset and hyperparameter configuration for subsequent analysis.

(4) Anticipated Reasons for the Good Performance of the Final Selected Dataset

Despite applying various preprocessing techniques and feature selection strategies, the highest predictive performance was consistently observed when utilizing the original dataset from the early stages of the experiment. Observing these results, it was determined that information loss occurred during some preprocessing steps. Therefore, based on this intuition, the final feature set was constructed by merging datasets created based on different criteria (origin, VIF, top15), adding 11 meaning-based derived variables on top, and performing additional preprocessing. It was anticipated that this approach yielded the highest performance for the following reasons:

i. **Information Complementation and Diversity Assurance**

The three types of datasets were designed according to different criteria, thus containing information from different perspectives (importance-focused vs. multicollinearity-removed vs. original). By merging them without unifying, the model can learn from a richer and more diverse input space. In particular, by considering variables deemed highly important (top15), VIF-selected variables to prevent overfitting, and the information from the original data altogether, stable learning without signal loss became possible.

1. **Understanding Inter-Variable Relationships with Derived Variables**

   It was confirmed through experimental results that derived variables reflecting inter-variable relationships and understandable interactions, rather than simple variables, positively influenced predictive performance. For example, variables such as lexical diversity (unique_token_ratio), emotional polarity difference (emotion_contrast), and sentiment-weighted subjectivity, which quantify content characteristics difficult to capture with single variables, are judged to have positively impacted performance improvement.

2. **Feature Selection for Overfitting Prevention**

   Not merely stopping at merging and adding data, additional preprocessing (correlation removal, VIF-based multicollinearity removal) was performed to balance the amount of information in the data against overfitting. Therefore, it can be said that the model was able to comprehensively learn various patterns without overfitting to a specific feature subset.

To summarize, the key factor for performance improvement is thought to be the maintenance of a refined structure that ensures information diversity while preventing overfitting. Rather than simple feature removal and preprocessing based on a single criterion, organically combining information from multiple perspectives and adding domain knowledge-reflecting derived variables based on this, provided a data structure enabling the model to learn more complex and practical patterns, which was the decisive factor leading to performance improvement.

3.5 Final Model Selection and Performance Evaluation

Through the preceding steps, the FullAugmentedSet_ver2 dataset and the Stacking model hyperparameters optimized via Grid Search for this dataset were confirmed as the final configuration (mean cross-validation Score 0.7827: Accuracy 0.7516, F1-Score 0.7613, AUC 0.8353). Now, using this confirmed configuration, the model was trained on the entire FullAugmentedSet_ver2.csv (here in after FullAugmentedSet_ver2 dataset), and its performance was evaluated to prepare the final model for external assessment.

3.5.1. Final Stacking Model Training (Applying Selected Hyperparameters)

The StackingClassifier model was trained on the entire FullAugmentedSet_ver2 training dataset using the selected optimal hyperparameters. After training completion, the performance metrics on this training dataset are as follows:

- **Training Data Accuracy**: 0.7528
- **Training Data F1 Score**: 0.7519
- **Training Data ROC AUC Score**: 0.8351

These figures indicate how well the model explains the training data. The training accuracy being at a similar level to the mean cross-validation accuracy (0.7516) suggests that the model was trained relatively stably.

3.5.2. Final Model Improvement through L2 Regularization Application

In an additional attempt to further enhance the model's generalization performance and prevent overfitting, the model was retrained by applying L2 regularization to the base learners of the Stacking model while maintaining the previously selected optimal hyperparameter structure.
The results of training the StackingClassifier with L2 regularization applied on the entire FullAugmentedSet_ver2 training dataset are as follows:

- **Training Data Accuracy (L2 Model)**: 0.7650
- **Training Data F1 Score (L2 Model)**: 0.7642

- **Training Data ROC AUC Score (L2 Model)**: 0.8475

The model with L2 regularization applied showed improved results across all training performance metrics (Accuracy, F1 Score, ROC AUC Score) compared to the model without regularization. This indicates that L2 regularization had a positive impact on the learning process, potentially leading to better generalization performance.

3.5.3. Final Model Confirmation and Recommendations

The Stacking model applying L2 regularization with the hyperparameters optimized for the FullAugmentedSet_ver2 dataset demonstrated the best training performance (Accuracy: 0.7650, F1 Score: 0.7642, ROC AUC: 0.8475). Therefore, this L2 regularized Stacking model is confirmed as the final model for this project.