PROJECT

PATTERN RECOGNITION

Team 5

2176045 KIM DOYOON

2176222 WOO MINHA

2491007 KIM YEEUN

2491025 LEE JIWON

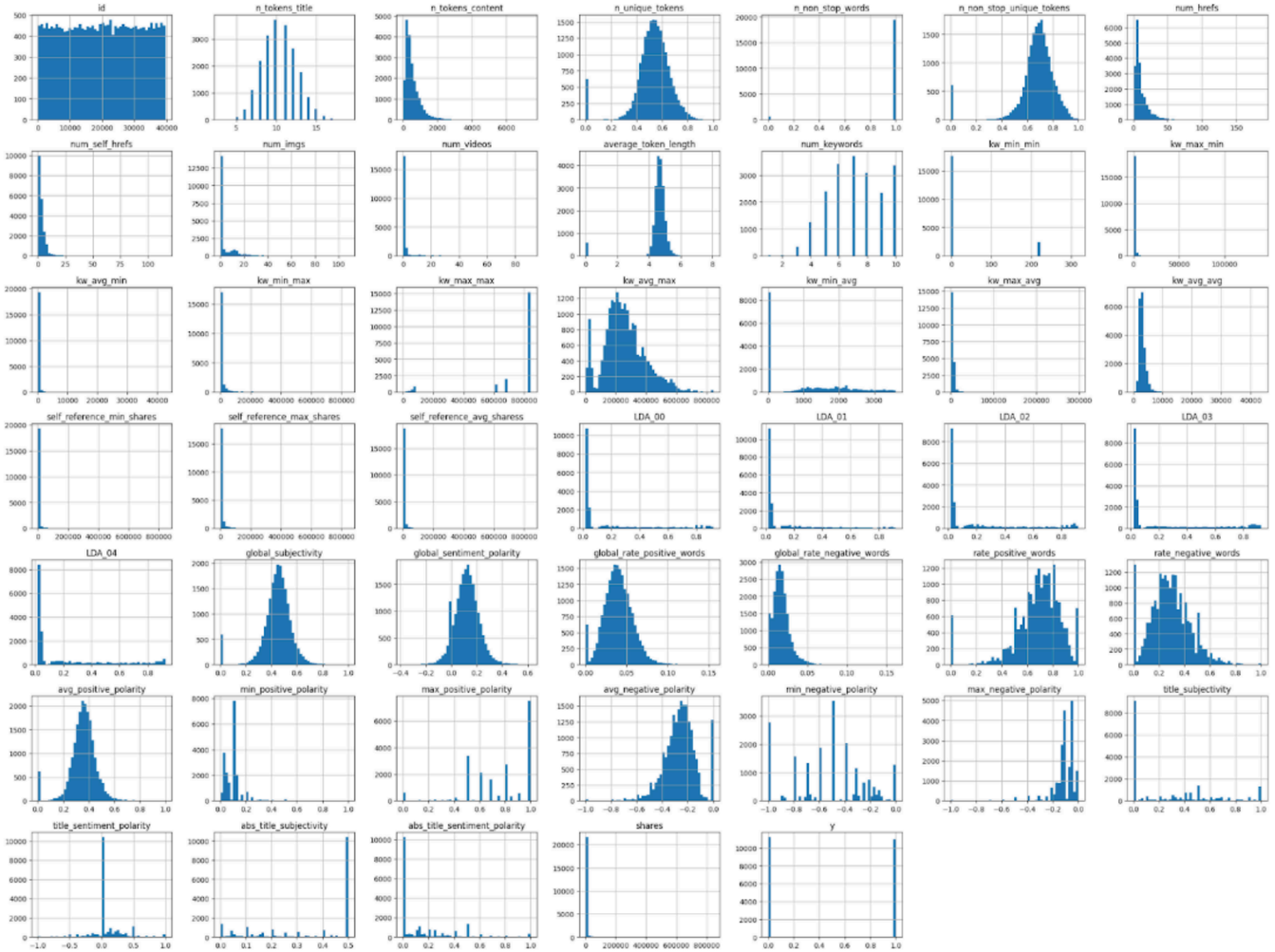
PROJECT

INDEX

DATA EXPLORATION

HISTOGRAM VISUALIZATION

DATA EXPLORATION



Histogram visualization

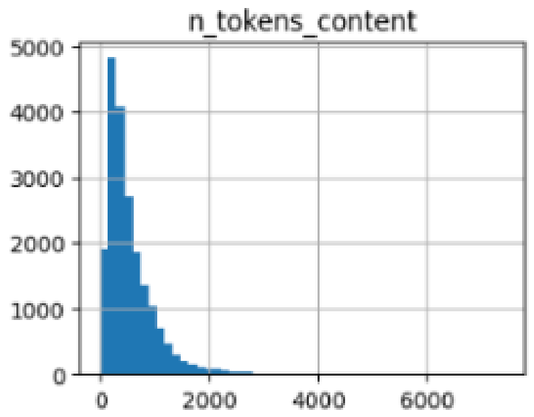
1. Long-tail distribution

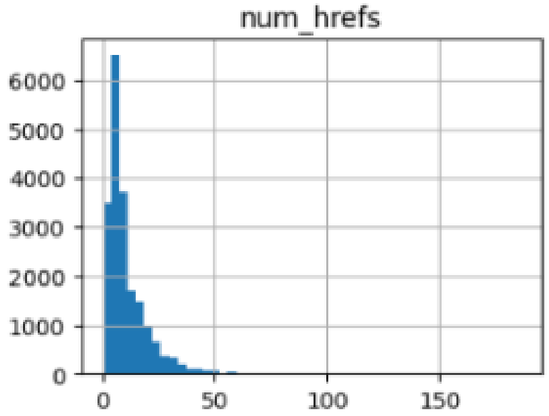
2. 0 spike + Outlier distribution

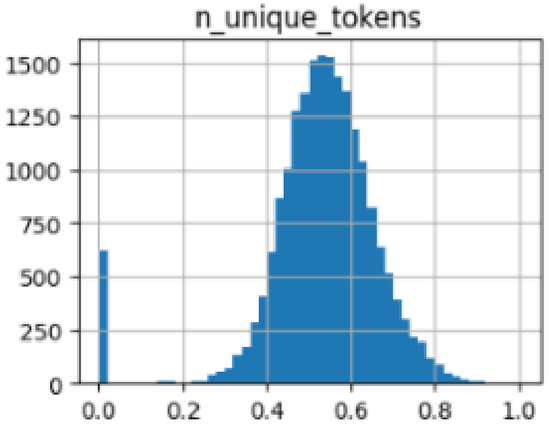
3. Discrete value spike

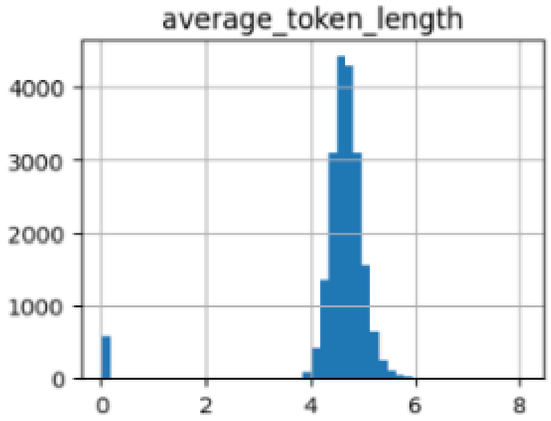
4. 0 spike + Outlier distribution

HISTOGRAM









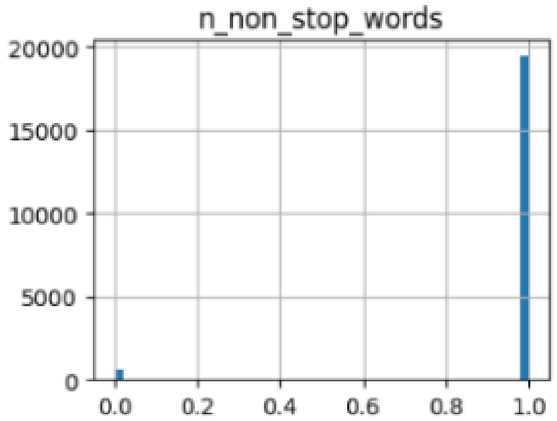
1. Long-tail distribution

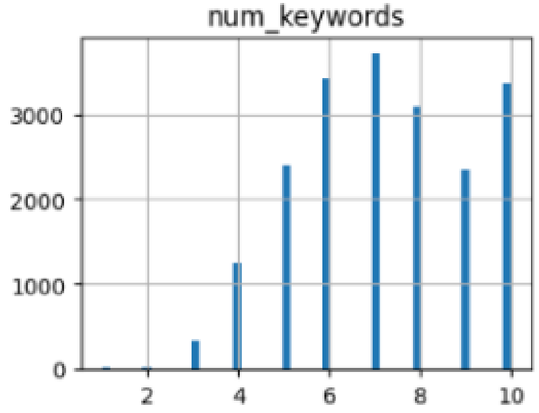
- n_tokens_content, num_hrefs, num_imgs, kw_* series, etc.
- Most values are clustered near 0, with a long tail extending towards larger values.

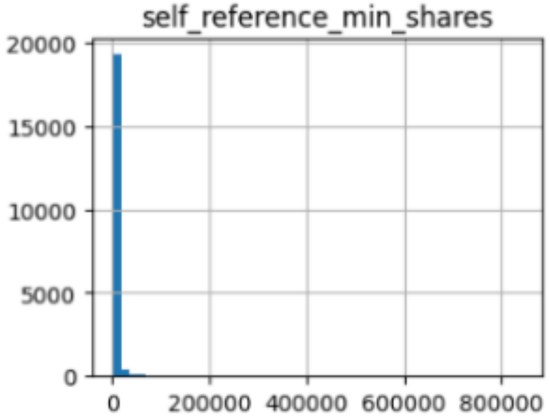
2. 0 spike + Outlier distribution

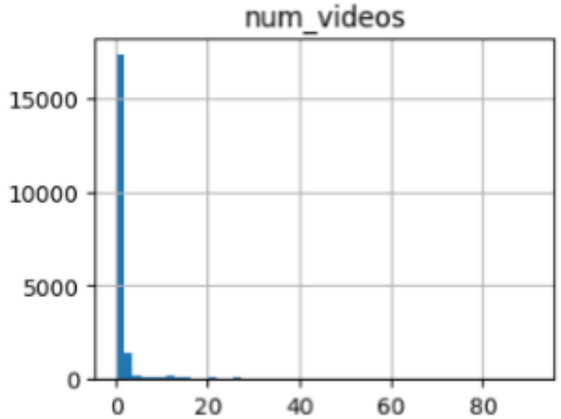
- n_unique_tokens, average_token_length, etc.
- These show a bell-shaped normal distribution centered in the middle.

HISTOGRAM









3. Discrete value spike

- n_non_stop_words, num_keywords, etc.
- distribution concentrated at specific values

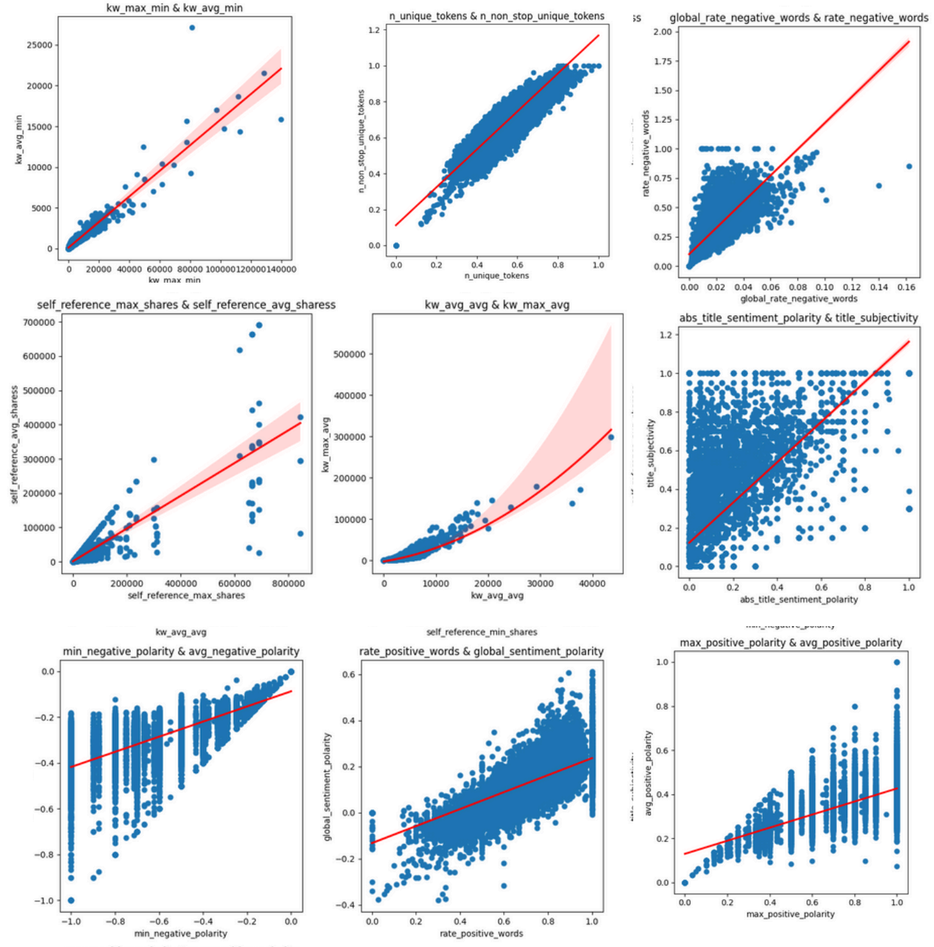
4. 0 spike + Outlier distribution

- self_reference_*, num_videos, etc.
- Most of the dataset consists of 0s, with only a few large outlier values.

PREPROCESSING

missing values & outliers

MISSING VALUES



Correlation Analysis

pairs of variables with absolute Pearson correlation coefficients greater than 0.8

→ linear or curved relationships.

↓

Regression

MISSING VALUES

```
# 1) 둘 다 결측치 없는 행으로 regression 학습용 데이터 준비
  mask_both_train = train[var_avg].notna() & train[var_max].notna()
  train_reg = train.loc[mask_both_train, [var_max, var_avg]]

# 모델1: var_max → var_avg
  model_avg = LinearRegression()
  model_avg.fit(train_reg[[var_max]], train_reg[var_avg])

# 모델2: var_avg → var_max
  model_max = LinearRegression()
  model_max.fit(train_reg[[var_avg]], train_reg[var_max])
```

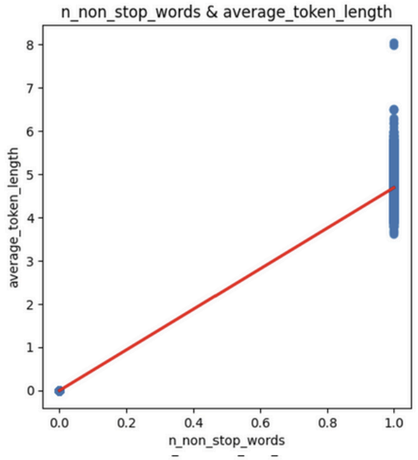
```
# test에도 동일하게 적용
  mask_both_test = test[var_avg].notna() & test[var_max].notna()

# 학습은 train 데이터만 사용
  mask_avg_miss_test = test[var_avg].isna() & test[var_max].notna()
  if mask_avg_miss_test.any():
    Xt_pred_avg = test.loc[mask_avg_miss_test, [var_max]]
    test.loc[mask_avg_miss_test, var_avg] = model_avg.predict(Xt_pred_avg)

  mask_max_miss_test = test[var_max].isna() & test[var_avg].notna()
  if mask_max_miss_test.any():
    Xt_pred_max = test.loc[mask_max_miss_test, [var_avg]]
    test.loc[mask_max_miss_test, var_max] = model_max.predict(Xt_pred_max)
```

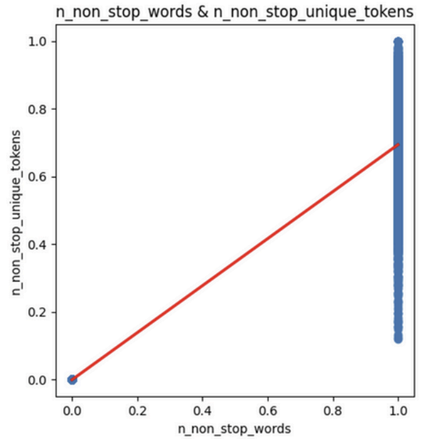
Regression

- **only rows without missing values** for both A and B were initially selected.
- **Linear Regression** OR **Polynomial Regression**
- For **rows where both variables were missing**, missing values were subsequently filled using the mean or median **based on the variable's skewness.**

MISSING VALUES

Relationships Among Three Variables



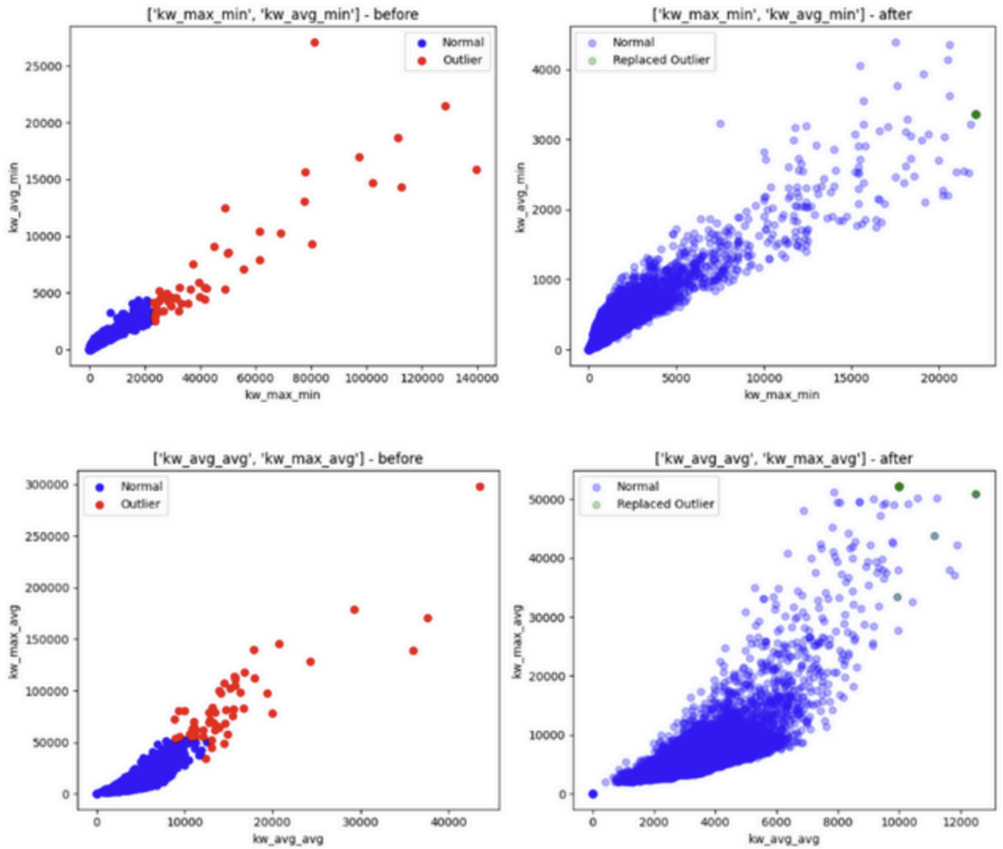


The following conditional logic can be derived:

- n_non_stop_words = 0 ⟶ the other two variables = 0
- average_token_length > 0
  OR
  n_non_stop_unique_tokens > 0
  ⟶ n_non_stop_words = 0

OUTLIERS

DBSCAN



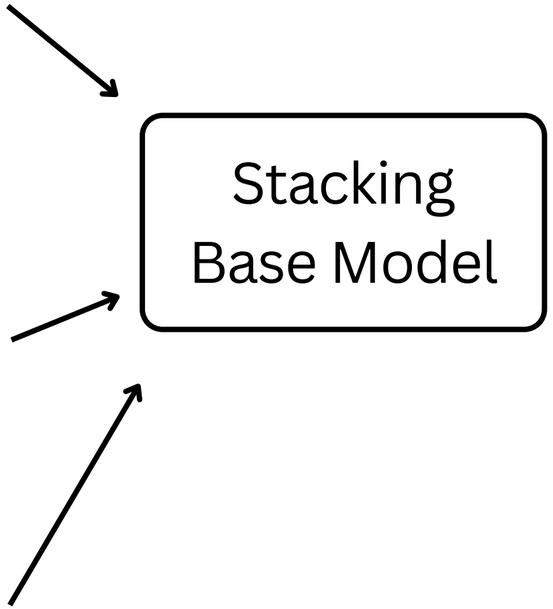
- Replacing outliers with the nearest normal point
- Four variable pairs (eight variables) are handled by DBSCAN
- The others are handled by clipping at the lower 5% and upper 95%

MODELING

STACKING

BASE MODEL

Model	Accuracy	F1 Score	ROC AUC	Mean Score
Logistic Regression	0.6423	0.628	0.6892	0.6532
Neural Network	0.6067	0.6016	0.6562	0.6215
Random Forest	0.667	0.6699	0.7215	0.6862
Gradient Boosting	0.6589	0.6579	0.7264	0.6811
LightGBM	0.6615	0.6606	0.7222	0.6814
XGBoost	0.6651	0.6644	0.7266	0.6854
Decision Tree	0.6206	0.6338	0.6601	0.6382
Catboost	0.6692	0.6674	0.7304	0.689



META MODEL TUNING

PARAMETER TUNING

```
# 4. 메타 모델 정의
meta_model = LogisticRegression(random_state=42)
```

- simplicity
- interpretability of output probabilities
- prevent overfitting

```
# 1. 튜닝 대상 조합 설정 (penalty, solver, l1_ratio, C)
param_combinations = [
    {'penalty': 'l2', 'solver': 'lbfgs', 'l1_ratio': None, 'C': 0.01},
    {'penalty': 'l2', 'solver': 'lbfgs', 'l1_ratio': None, 'C': 0.1},
    {'penalty': 'l1', 'solver': 'liblinear', 'l1_ratio': None, 'C': 0.01},
    {'penalty': 'l1', 'solver': 'liblinear', 'l1_ratio': None, 'C': 0.1},
    {'penalty': 'elasticnet', 'solver': 'saga', 'l1_ratio': 0.5, 'C': 0.01},
    {'penalty': 'elasticnet', 'solver': 'saga', 'l1_ratio': 0.5, 'C': 0.1}
]
```

META MODEL TUNING

THRESHOLD TUNING

Threshold tuning

0.40 ~ 0.60 in 0.01 increments

highest score
in **0.44**

Final Result

Accuracy: 0.6614928098607624

F1 Score: 0.6917480773228019

ROC AUC : 0.7333949573876959

Mean Score: 0.6955452815237534

FUTURE DIRECTION

FUTUTE DIRECTION

01 Feature Engineering

02 Diversification and Addition of Stacking Base Models

THANK YOU