Machine Learning Group Project - Popularity

Ethan Lu (14724235), Grace Chang 14820056, Chi Kit Tin 14711079

Objective

This project aims to evaluate the factors contributing to the popularity of online news articles by analyzing various elements such as their topics, number of words and their sentiment. Utilizing a dataset of 39,797 articles, the project has two primary objectives. First, it seeks to classify which articles are likely to be popular by defining an appropriate popularity threshold and comparing articles that meet or exceed this threshold with those that do not, to identify distinguishing characteristics. Second, it aims to group articles based on features such as words, topics, and general sentiment to examine whether these clusters align with article popularity.

Beyond these objectives, the project provides opportunities for deeper exploration, such as studying the sensitivity of models to variations in training set sizes and regularization strength to improve generalization. It also includes performing feature importance analysis to identify the most influential predictors of article popularity and applying unsupervised learning techniques, such as clustering and outlier detection, to uncover hidden patterns and trends in the data. Ultimately, the project's main goal is to investigate the relationship between article features and their popularity by leveraging both supervised and unsupervised learning approaches.

Data understanding

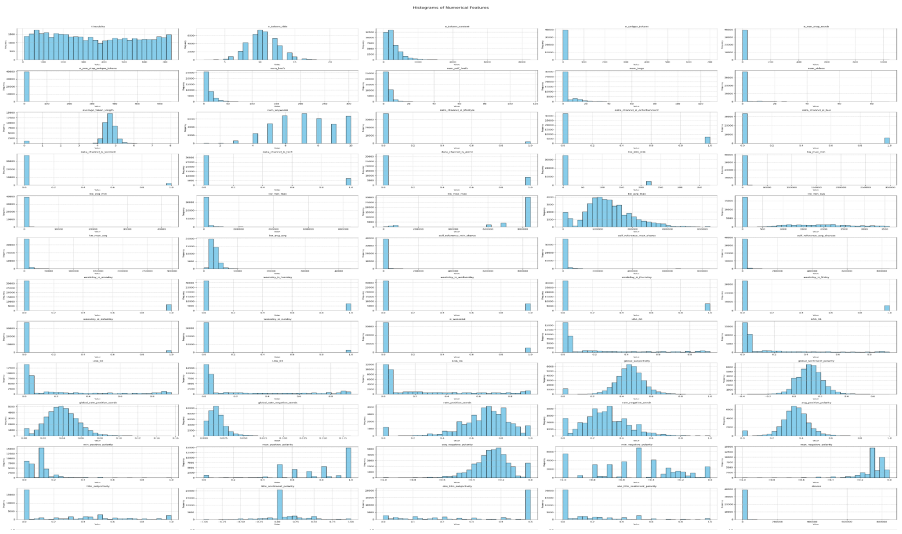
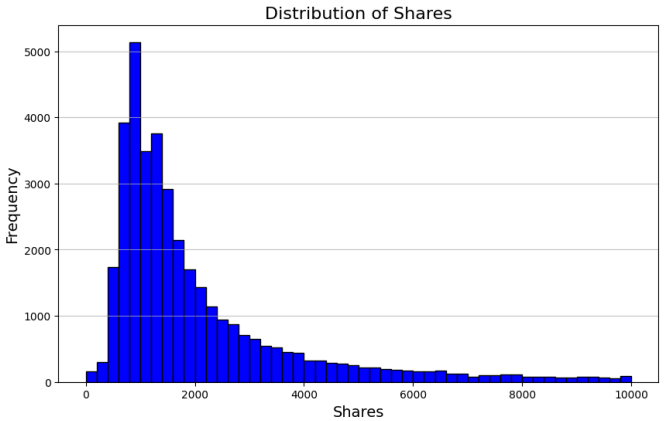


Figure 1: Distribution of Features in the Dataset

The dataset comprises 60 features, making it high-dimensional, and it contains no missing values, with all variables appropriately formatted in their correct data types. However, certain features exhibit significant skewness in their distributions. Using pd.info() and pd.describe() we saw that there are multiple features with near-zero variance.




```
Median Threshold: 1400.0
75th Percentile Threshold: 2800.0
pop_level
Popular        21154
Not Popular    18490
Name: count, dtype: int64
```

Figure 2: Distribution of the number of shares & its median

Comparing the distribution and the median of the number of shares, it is observed that using 1400 as the threshold of popularity allows less unbalanced two classes.

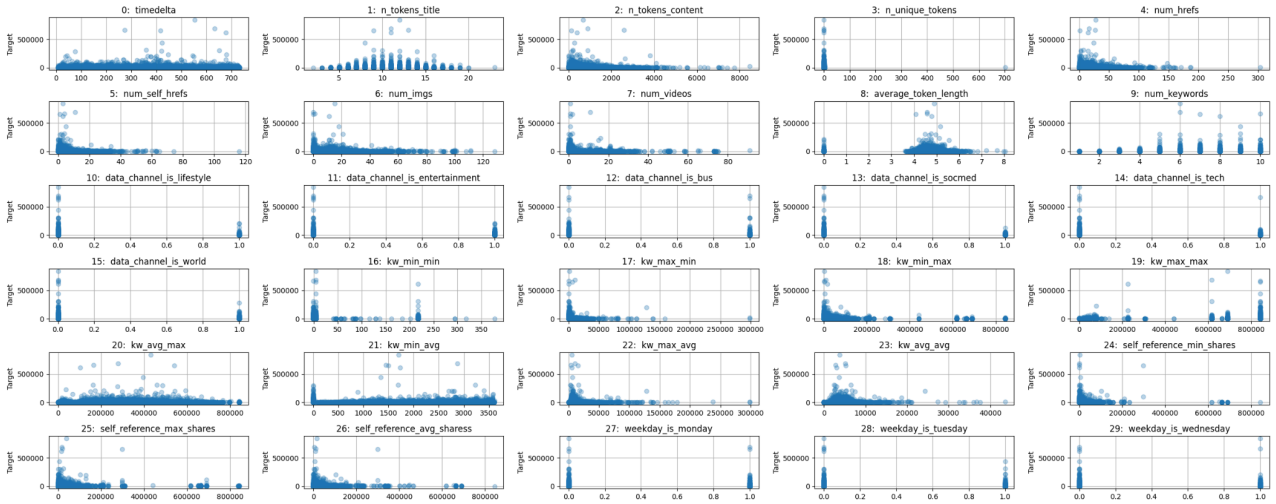


Figure 3: Scatter Plots of Features Against the Target Variable (Shares in numerical form)

The majority of the features exhibit nonlinear relationships with the target variable, suggesting that linear models may not effectively capture the complexities in the data.

```
selected_features = target_corr[abs(target_corr) > 0.1].index
print("Selected Features:", selected_features)

✓ 0.0s
Selected Features: Index([' kw_avg_avg', ' shares'], dtype='object')
```

Figure 4: Correlation between features and Shares

The relationships between X and Y are mostly nonlinear given that correlations are mostly below 0.1, suggesting correlation-based feature selection is an inappropriate approach to reducing unnecessary features and dimension. Thus, other methods will be tried instead.

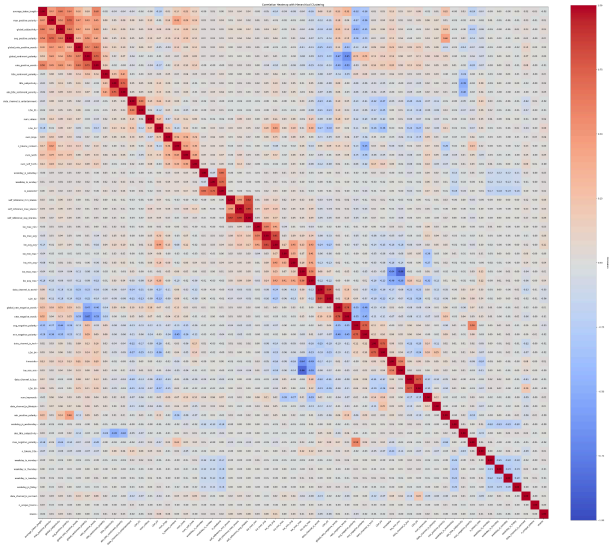


Figure 5: Heatmap of Feature Correlations

Figure 5. illustrates that several features exhibit high correlations, indicating they may share similar information with the target variable. This suggests the potential necessity for further dimensionality reduction through feature selection to eliminate redundancy and improve model efficiency.

Improvement Experiments using PCA & Outlier detection

Due to the correlated features, the skewness of feature distributions, and the wide varieties of feature variances, PCA (dimension reduction) is used to filter certain linear relationships of features before modeling. This allows us to facilitate model computation by compromising merely a slight amount of variance of the feature data.

```
Explained variance ratio for the components: [0.08236348 0.07028717 0.0620355  0.05002959 0.04758211 0.04324228
 0.04239547 0.03873209 0.0361785  0.03492986 0.03363256 0.03144707
 0.02802906 0.02308833 0.02285155 0.02082632 0.02050289 0.02029982
 0.02011905 0.01963944]
```

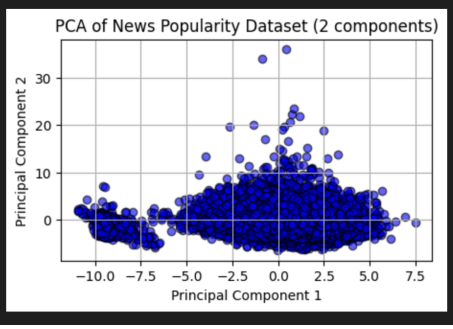


Figure 6 : The first 20 components perceive 75% of the data with only keeping ⅓ of the dataset

On the other hand, outlier detection eliminates those less relevant influential features to the target (popularity). This may allow us to achieve a more precise classification of the popularity of an article given certain features. We mainly use OneClassSVM and Isolation Forest.

Models fitted with Original Data

```
Best Parameters: {'Classifier__C': 1.0, 'Scaler': StandardScaler()}
Best Estimator: Pipeline(steps=[('Scaler', StandardScaler()),
                ('Classifier', LinearSVC(dual=False, max_iter=5000))])
Training Score (accuracy) of Best Estimator):0.6538
Test Score (accuracy) of Best Estimator :0.6525

=== Evaluation: LinearSVC with Original Data ===
Confusion matrix:
[[2281 1493]
 [1262 2893]]
Classification report:
              precision    recall  f1-score   support

 Not Popular       0.64      0.60      0.62      3774
     Popular       0.66      0.70      0.68      4155

    accuracy                           0.65      7929
   macro avg       0.65      0.65      0.65      7929
weighted avg       0.65      0.65      0.65      7929

Micro-average Precision: 0.6525
Micro-average Recall: 0.6525
```

```
=== Logistic Regression with Original Data ===
Best Parameters: {'logisticregression__C': 1, 'standardscaler': StandardScaler()}
Best Cross-Validation Score: 0.654106889484471

=== Evaluation: Logistic Regression with Original Data ===
Confusion Matrix:
 [[2261 1513]
 [1245 2910]]
Classification Report:
              precision    recall  f1-score   support

 Not Popular       0.64      0.60      0.62      3774
     Popular       0.66      0.70      0.68      4155

    accuracy                           0.65      7929
   macro avg       0.65      0.65      0.65      7929
weighted avg       0.65      0.65      0.65      7929

Micro-average Precision: 0.6522
Micro-average Recall: 0.6522
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
Best Parameters: {'classifier__max_depth': 10, 'classifier__max_features': 'sqrt', 'classifier__n_estimators': 100}
Best Cross-Validation Accuracy: 0.6685480056755478
Test Accuracy: 0.6674233825198638

=== Evaluation: Random Forest with Original Data ===
Confusion matrix:
[[2190 1584]
 [1053 3102]]

=== Classification Report ===
              precision    recall  f1-score   support

 Not Popular       0.68      0.58      0.62      3774
     Popular       0.66      0.75      0.70      4155

    accuracy                           0.67      7929
   macro avg       0.67      0.66      0.66      7929
weighted avg       0.67      0.67      0.66      7929

Micro-average Precision: 0.6674
Micro-average Recall: 0.6674
```

The three models show moderate performance, likely underfitting as indicated by similar accuracy scores. All models struggle with lower recall for the 'Not Popular' class, failing to capture truly unpopular samples, though micro and macro scores highlight general prediction power. LinearSVC and Logistic Regression achieve similar test accuracies (~65%) with balanced metrics, while Random Forest outperforms them with 66.74% accuracy and stronger precision, recall, and F1-scores (67%), excelling in predicting the 'Popular' class. Overall, Random Forest is the best-performing model.

Models fitted with Original Data Preprocessed by PCA

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits
Best Parameters: {'max_depth': 10, 'max_features': 'sqrt', 'n_estimators': 100}
Best Cross-Validation Accuracy: 0.6465394923537758

=== Evaluation: Random Forest with PCA Data ===
Test Accuracy: 0.6437129524530205
Confusion matrix:
[[2036 1738]
 [1087 3068]]

=== Classification Report ===
              precision    recall  f1-score   support

 Not Popular       0.65      0.54      0.59      3774
     Popular       0.64      0.74      0.68      4155

    accuracy                           0.64      7929
   macro avg       0.65      0.64      0.64      7929
weighted avg       0.64      0.64      0.64      7929

Micro-average Precision: 0.6437
Micro-average Recall: 0.6437
```

The random forest classifier performs best in the previous section with the original data. We only experimented on this classifier to fit with the data processed by PCA mentioned above. The below evaluation suggests the PCA preprocessing likely removed important feature information, negatively impacting the classifier's ability to distinguish between classes. This may suggest that dimensionality reduction using PCA can harm performance when the original features carry critical predictive information.

Models fitted with data without outliers by OneClassSVM

OneClassSVM requires manual selection of the proportion of the outliers. Using \nu = 0.2 almost has no impact on the performance of the linear classifications while improving the random forest classifier. The tuning in parameter \nu in OneClassSVM can affect its effect. However, it is challenging for us to commit to the tuning within our knowledge. Thus, we decided to experiment on Isolation Forest outlier detection. Since it performs well in high-dimensional datasets and is robust to skewed features, it is chosen to be our second experimental outlier detector.

```
: Parameters: {'Classifier__C': 0.01, 'Scaler': StandardScaler()}
: Estimator: Pipeline(steps=[('Scaler', StandardScaler()),
             ('Classifier', LinearSVC(C=0.01, dual=False, max_iter=5000
ining Score (accuracy) of Best Estimator: 0.6577
: Score (accuracy) of Best Estimator: 0.6528

Evaluation: LinearSVC without outliers by OneClassSVM===
usion matrix:
016 1148]
60 2136]]
sification report:
              precision    recall  f1-score   support

: Popular       0.66      0.64      0.65      3164
 Popular       0.65      0.67      0.66      3196

 accuracy                           0.65      6360
acro avg       0.65      0.65      0.65      6360
hted avg       0.65      0.65      0.65      6360

o-average Precision: 0.6528
o-average Recall: 0.6528
```

```
Best Parameters: {'Classifier__C': 0.01, 'Scaler': StandardScaler()}
Best Estimator: Pipeline(steps=[('Scaler', StandardScaler()),
                ('Classifier', LinearSVC(C=0.01, dual=False, max_iter=5000))]
Training Score (accuracy) of Best Estimator: 0.6577
Test Score (accuracy) of Best Estimator: 0.6528

=== Evaluation: LogisticRegressor without outliers by OneClassSVM ===
Confusion matrix:
 [[1997 1167]
 [1048 2148]]
Classification report:
              precision    recall  f1-score   support

 Not Popular       0.66      0.63      0.64      3164
     Popular       0.65      0.67      0.66      3196

    accuracy                           0.65      6360
   macro avg       0.65      0.65      0.65      6360
weighted avg       0.65      0.65      0.65      6360

Micro-average Precision: 0.6517
Micro-average Recall: 0.6517
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits

=== RandomForestClassifier Without Outliers (Cleaned Data) ===
Best Parameters: {'classifier__max_depth': 10, 'classifier__max_features': 'log2', 'classifier__n_estimators': 100}
Best Cross-Validation Score: 0.6684138474712191
Test Accuracy (After Isolation Forest): 0.6681

=== Evaluation: RandomForestClassifier Without Outliers by OneClassSVM ===
Confusion Matrix:
 [[1953 1211]
 [ 900 2296]]
Classification Report:
              precision    recall  f1-score   support

 Not Popular       0.68      0.62      0.65      3164
     Popular       0.65      0.72      0.69      3196

    accuracy                           0.67      6360
   macro avg       0.67      0.67      0.67      6360
weighted avg       0.67      0.67      0.67      6360

Micro-average Precision: 0.6681
Micro-average Recall: 0.6681
```

After removing outliers using One-Class SVM, Random Forest achieved the best performance with a test accuracy of 66.81% and balanced metrics (67% precision, recall, and F1-score), excelling in predicting the "Popular" class. LinearSVC and Logistic Regression showed minimal changes, maintaining test accuracies around 65%.

LinearSVC's marginal improvement after outlier removal can be attributed to its margin-based loss, which inherently makes it less sensitive to outliers. Logistic Regression, being robust to mild outliers due to its convex log-loss optimization, also shows limited improvement when outliers are eliminated.

Random Forest improves more significantly because it relies on recursive partitioning of the data, and outliers can distort the splits and tree structure. Removing outliers allows Random Forest to create more accurate splits, leading to better performance.

Models fitted with data without outliers by Isolation Forest

```
Result of Isolation Forest (Outlier Detection):
Number of inliers in training data: 25372
Number of outliers removed from training data: 6343
Training Data: 25372 inliers, 6343 outliers removed.
Test Data: 6316 inliers, 1613 outliers removed.
```

With a similar proportion of outliers detected, Isolation Forest improves classifiers more average on the three models. While the improvement on random forests is similar to the effect of OneClassSVM, the classifier still performs better than the others.

```
=== Evaluation: LinearSVC without outliers by Isolation Forest ===
Best Parameters: {'linearsvc__C': 1, 'standardscaler': MinMaxScaler()}
Best Cross-Validation Score: 0.6535948552555187
Test Accuracy (After Isolation Forest): 0.6533
Confusion Matrix:
 [[2030 1122]
 [1068 2096]]
Classification Report:
              precision    recall  f1-score   support

 Not Popular       0.66      0.64      0.65      3152
     Popular       0.65      0.66      0.66      3164

    accuracy                           0.65      6316
   macro avg       0.65      0.65      0.65      6316
weighted avg       0.65      0.65      0.65      6316

Micro-average Precision: 0.6533
Micro-average Recall: 0.6533
```

After removing outliers using Isolation Forest, LinearSVC and Logistic Regression showed similar performance with a test accuracy of 65.33%, maintaining balanced precision, recall, and F1-scores (65%) for both classes. Random Forest achieved the highest test accuracy (66.50%) and improved metrics, with better precision (67%), recall (66%), and F1-scores (66%), especially for the "Popular" class. Overall, removing outliers led to marginal improvements, with Random Forest continuing to outperform the other models.

```
=== Evaluation: Logistic Regression without outliers by Isolation Forest ===
Best Parameters: {'logisticregression__C': 0.01, 'standardscaler': StandardScaler()}
Best Cross-Validation Score: 0.6541466725953426
Test Accuracy (After Isolation Forest): 0.6533
Confusion Matrix:
 [[2021 1131]
 [1067 2097]]
Classification Report:
              precision    recall  f1-score   support

 Not Popular       0.65      0.64      0.65      3152
     Popular       0.65      0.66      0.66      3164

    accuracy                           0.65      6316
   macro avg       0.65      0.65      0.65      6316
weighted avg       0.65      0.65      0.65      6316

Micro-average Precision: 0.6520
Micro-average Recall: 0.6520
```

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits

=== RandomForestClassifier Without Outliers (Cleaned Data) ===

=== Evaluation: RandomForestClassifier Without Outliers by Isolation Forest ===
Best Parameters: {'classifier__max_depth': 10, 'classifier__max_features': 'log2', 'classifier__n_estimators': 100}
Best Cross-Validation Accuracy: 0.6649062097702767
Test Accuracy (After Isolation Forest): 0.6650
Confusion Matrix:
 [[1980 1172]
 [ 944 2220]]
Classification Report:
              precision    recall  f1-score   support

 Not Popular       0.68      0.63      0.65      3152
     Popular       0.65      0.70      0.68      3164

    accuracy                           0.66      6316
   macro avg       0.67      0.66      0.66      6316
weighted avg       0.67      0.66      0.66      6316

Micro-average Precision (Cleaned Test Data): 0.6650
Micro-average Recall (Cleaned Test Data): 0.6650
```

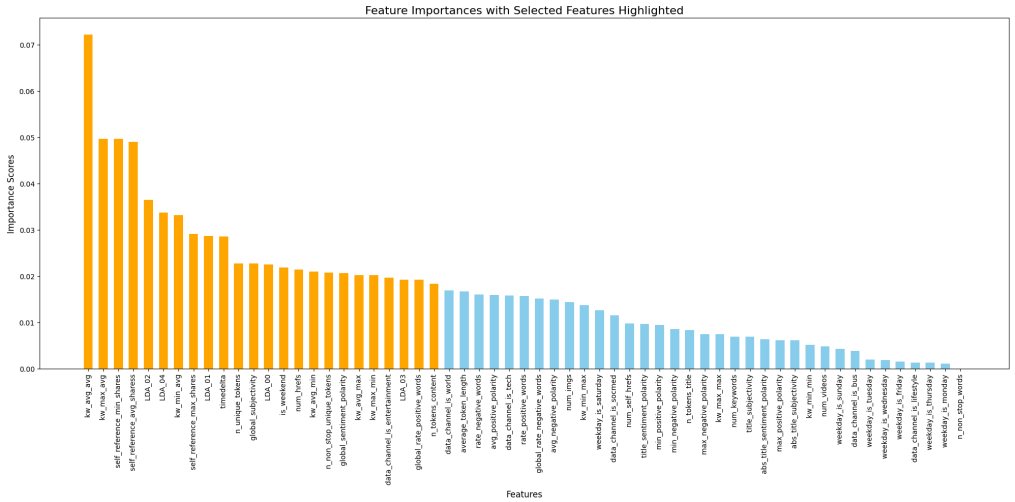
Deeper Exploration of the Best Model: Random Forest Classifier Without Outliers by Isolation Forest

1. Balance the two classes by class_weight

```
Fitting 5 folds for each of 24 candidates, totalling 120 fits

=== RandomForestClassifier Without Outliers (Cleaned Data) ===

=== Evaluation: RandomForestClassifier Without Outliers by Isolation Forest After Balance===
Best Parameters: {'classifier__max_depth': 10, 'classifier__max_features': 'log2', 'classifier__n_estimators': 100}
Best Cross-Validation Accuracy: 0.664630130230228
Test Accuracy (After Isolation Forest): 0.6713
Confusion Matrix:
 [[2071 1081]
 [ 995 2169]]
Classification Report:
              precision    recall  f1-score   support

 Not Popular       0.68      0.66      0.67      3152
     Popular       0.67      0.69      0.68      3164

    accuracy                           0.67      6316
   macro avg       0.67      0.67      0.67      6316
weighted avg       0.67      0.67      0.67      6316

Micro-average Precision (Cleaned Test Data): 0.6713
Micro-average Recall (Cleaned Test Data): 0.6713
```

Balancing classes with class_weight improved test accuracy from 66.50% to 67.13%, increasing recall for "Not Popular" from 63% to 66% while maintaining strong performance for "Popular." The balanced model better handles class imbalance with more equitable predictions.

2. Feature Importance Analysis & Feature Selection by SelectFromModel



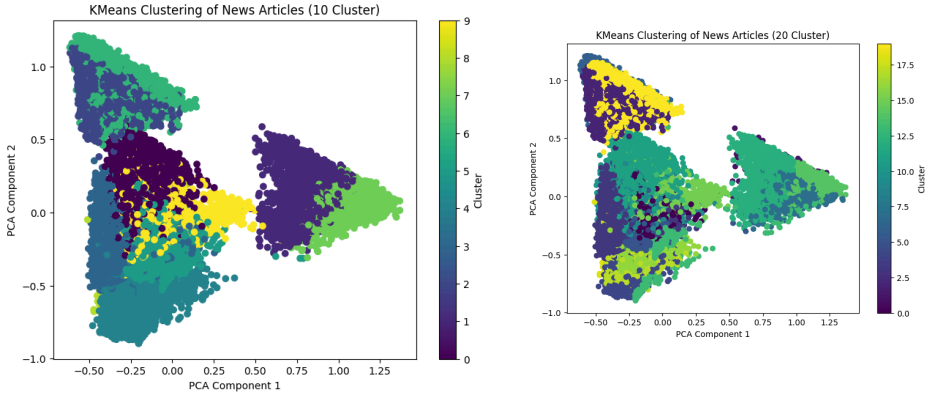
The feature importances are all relatively low. Using the median importance threshold, SelectFromModel filtered the most influential features to train a new Random Forest classifier.

3. Fit the random forest classifier using the selected features by SelectFromModel

```
=== Evaluation After SelectFromModel ===
Test Accuracy (Selected Features): 0.6643
Confusion Matrix:
 [[2047 1105]
 [1015 2149]]
Classification Report:
              precision    recall  f1-score   support

 Not Popular       0.67      0.65      0.66      3152
     Popular       0.66      0.68      0.67      3164

    accuracy                           0.66      6316
   macro avg       0.66      0.66      0.66      6316
weighted avg       0.66      0.66      0.66      6316

Micro-average Precision (Selected Features): 0.6643
Micro-average Recall (Selected Features): 0.6643
```

The balanced Random Forest classifier, after feature selection, shows slightly worse performance than before selection. This suggests that the feature selection process might have removed some features that, while having low individual importance, collectively contributed to the model's performance. The marginal decline indicates that the eliminated features could provide complementary information to the selected ones, highlighting the trade-off between dimensionality reduction and retaining predictive power.

K-means clustering (on manually selected features)



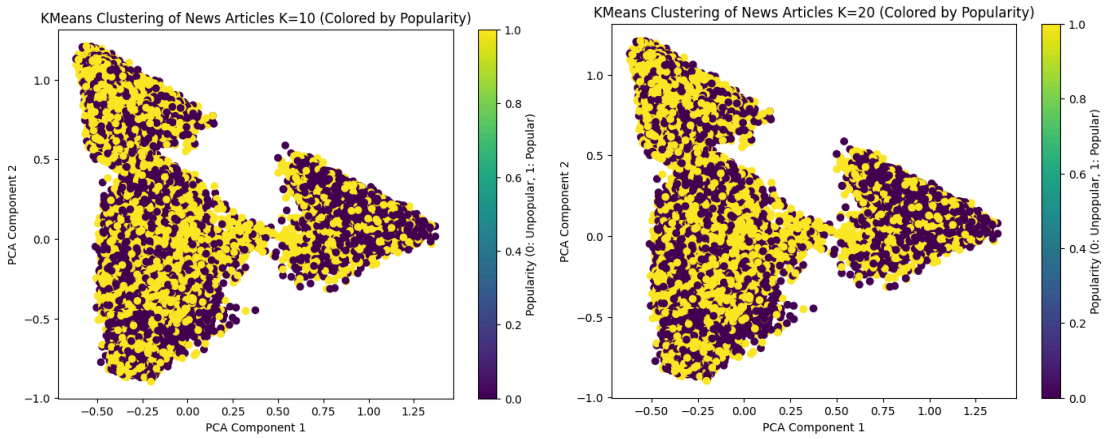
We performed K-means clustering with k=10 and k=20 to assess whether increasing the number of clusters improves performance. A "popularity" column was added, labeling articles with shares > 1400 as 1 (popular) and those < 1400 as 0 (unpopular). Clustering was based on features such as word usage, topics, and sentiment. The visualizations reveal significant overlap between clusters, particularly in densely packed regions, indicating that increasing k does not resolve the issue of overlapping clusters.

```
Comparison of Clusters with Popularity:
cluster
0    0.571292
1    0.421870
2    0.634241
3    0.484155
4    0.371546
5    0.613657
6    0.559551
7    0.307946
8    0.500899
9    0.715974
Cluster Purity (Distribution of Popularity within Each Cluster):
cluster  pop_label
0        1            1194
         0             896
1        0            1713
         1            1250
2        1            2119
         0            1222
3        0            3223
         1            3025
4        0            4139
         1            2447
5        1            3415
         0            2150
6        1            2241
         0            1764
7        0            3771
         1            1678
8        1             557
         0             555
9        1            1636
         0             649
```

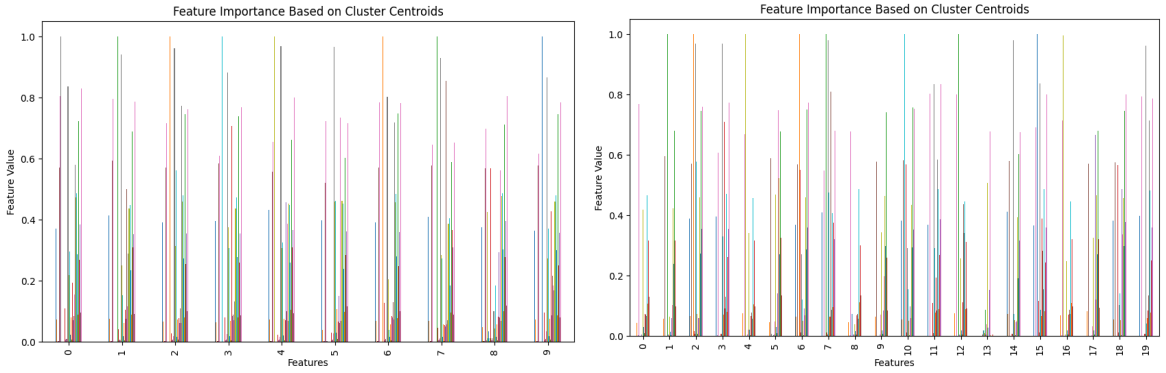
```
Comparison of Clusters with Popularity:
cluster
0     0.589744
1     0.439252
2     0.640417
3     0.481570
4     0.392584
5     0.647541
6     0.643892
7     0.319210
8     0.554608
9     0.730881
10    0.497537
11    0.566764
12    0.435414
13    0.557013
14    0.293799
15    0.704214
16    0.347024
17    0.366157
18    0.421384
19    0.534545
```

```
Cluster Purity (Distribution of Popularity within Each Cluster):
cluster  pop_label
0        1            1403
         0             976
1        0             300
         1             235
2        1            1968
         0            1105
3        0            2813
         1            2613
4        0            1507
         1             974
5        1            1659
         0             903
6        1             622
         0             344
7        0            1896
         1             889
8        1             325
         0             261
9        1             755
         0             278
10       0             408
         1             404
11       1            1163
         0             889
12       0            1167
         1             900
13       1             552
         0             439
14       0            2084
         1             867
15       1             869
         0             365
16       0            1415
         1             752
17       0            1120
         1             647
18       0             276
         1             201
19       1            1764
         0            1536
```

These provide detailed information about each cluster. For instance, in the case of k=10,, the highest popularity proportion within a cluster is 0.7159, indicating that approximately 72% of the data points in this cluster are classified as popular. The corresponding graph below shows that this cluster has a size of 2288, comprising 1636 popular and 649 unpopular points. However, most clusters contain a significant proportion of both "popular" and "unpopular" points, highlighting that the clustering does not achieve strong separation based on popularity. This demonstrates that the clustering algorithm struggles to effectively distinguish between "popular" and "unpopular" points, as no cluster achieves 100% purity in either category.

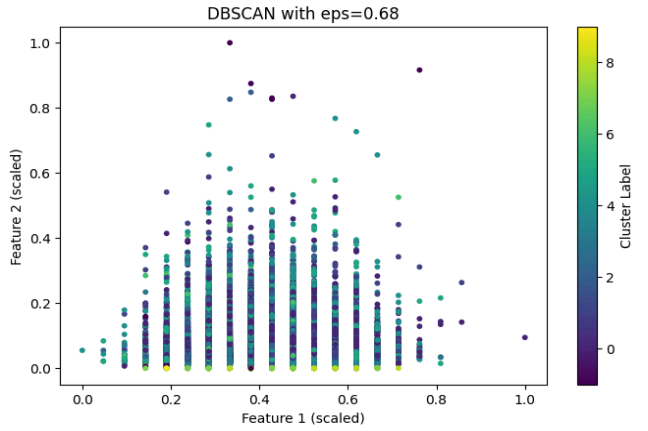


These show that yellow and purple data points, representing popular and unpopular articles respectively, are scattered throughout the clusters without clear distinction. This suggests that the selected features and clustering approach do not effectively capture the differentiation between popular and unpopular articles. While increasing kk introduces additional complexity to the clustering, it does not result in better alignment with the target variable (popularity). This may indicate that the features used for clustering have a weak correlation with popularity or that the data lacks a well-defined cluster structure related to this target.



The analysis of feature importance across clusters reveals that each cluster predominantly relies on a small subset of features with high importance, while the majority of features contribute minimally. This indicates that the clustering algorithm is overly dependent on specific dimensions, suggesting a lack of balanced representation across features within the dataset. Such dependency on a limited number of features may result in redundancy or overfitting, where clusters become sensitive to variations in these dominant features while neglecting potentially meaningful patterns in the remaining features. Consequently, this unbalanced reliance undermines the model's ability to generalize and effectively separate the data, leading to poor clustering performance, particularly in relation to the target variable, such as popularity.

DBSCAN clustering




```
Number of clusters: 10
Number of noise points: 921
Cluster sizes:
 0    8135
 1    7279
 4    6781
 5    6191
 3    5390
 2    2030
 6    1876
-1     921
 8     633
 7     251
 9     157
Name: count, dtype: int64
```

The DBSCAN clustering with eps = 0.68 and min_samples = 100 effectively groups the data into 10 clusters, capturing distinct patterns with a balanced trade-off between noise and clusters.

Larger clusters (like 0, 1, and 4) represent broad trends with common topics and sentiments, while smaller clusters (7,8,9) capture specific or unique articles with specialized content. Noise is minimal, indicating that most articles are meaningfully grouped.

```
Adjusted Rand Index (ARI): 0.0155
Normalized Mutual Information (NMI): 0.0212
```

Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI)
The ARI indicates a very limited level of agreement between the predicted label and the ground truth label. The result is just slightly better than a random result. The NMI score also indicates nearly zero mutual information between predicted and the ground truth label.

The result shows the popularity of an article is not directly related to topics, sentiments etc.

```
Comparison of Clusters with Popularity
clusters_db
-1    0.626493
 0    0.342225
 1    0.592664
 2    0.564532
 3    0.610390
 4    0.372364
 5    0.483767
 6    0.732942
 7    0.498008
 8    0.584518
 9    0.388535
Name: pop_label, dtype: float64

Cluster Purity (Distribution of Popularity within Each Cluster):
clusters_db  pop_label
-1           1            577
             0            344
 0           0           5351
             1           2784
 1           1           4314
             0           2965
 2           1           1146
             0            884
 3           1           3290
             0           2100
 4           0           4256
             1           2525
 5           0           3196
             1           2995
 6           1           1375
             0            501
 7           0            126
             1            125
 8           1            370
             0            263
 9           0             96
             1             61
Name: count, dtype: int64
```

Comparison of Clusters with Popularity Cluster

The average popularity score represents the level of engagement received from an audience.
Large cluster size with low score: The result of clusters like 0, 4 indicated clusters containing broad articles but not attractive to readers. It can be explained as a repetitive article that lacks originality.
Small cluster size with high score: Clusters like 7, 8 represent specific and unique topics that attract audiences. These articles contain unique keywords and have a targeted audience.
The high score of noise indicates extremely unique topics with distinct features that lack similarity among other points to form a cluster.

Cluster Purity
Cluster purity gives us insight into the level of dominance of the popularity label within a cluster.
In our case, **0 represents unpopular labels** and **1 represents popular labels**. The unpopular labels in clusters 0, and 4 are dominant which matches our analysis of the popularity score.
In contrast, the outstanding performance of cluster 6 indicates that the cluster captures highly engaging content. Thus, the score of cluster 6 is the highest among all clusters.