Text Mining and Search Project Report

Pasquale Formicola

Angelo Giuseppe Limone

16/01/2024

Contents

1 Introduction

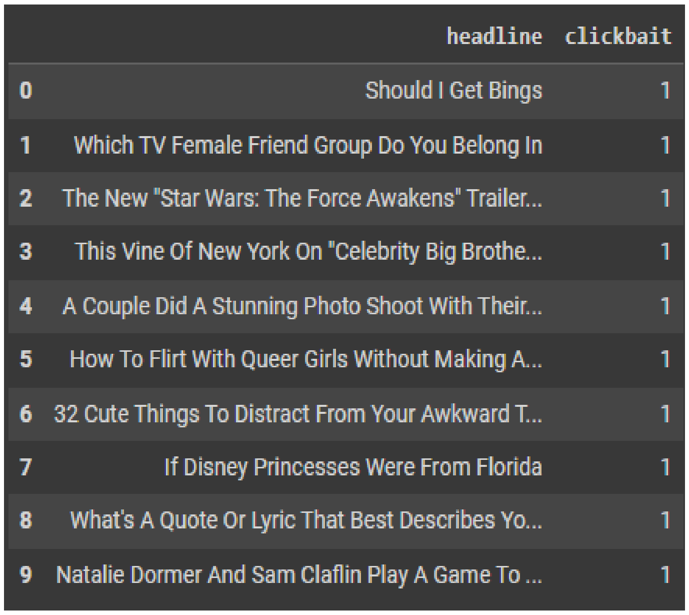
The report is about a Binary Classification and Clustering Text Mining Models creation. The dataset used is the Clickbait Dataset, which attempts to classify news headlines as clickbait or non-clickbait.

Clickbait, characterized by sensationalized headlines designed to attract attention, has become prevalent on various online platforms. The need to develop effective clickbait detection models is crucial to maintaining user trust and enhancing content quality. This report explores different aspects of clickbait detection, from data exploration to the application of Text Mining algorithms.

2 Data Description

The Clickbait Dataset, available on Kaggle, assembles information sourced from different news platforms. Clickbait headlines are extracted from websites like BuzzFeed, Upworthy, ViralNova, Thatscoop, Scoopwhoop, and ViralStories. Meanwhile, relevant or non-clickbait headlines are sourced from reputable news outlets such as WikiNews, New York Times, The Guardian, and The Hindu.

This dataset is structured with two columns: the first column contains the headlines, and the second column provides numerical labels indicating whether the headline is clickbait. A label of 1 denotes clickbait, while a label of 0 indicates a non-clickbait headline. In totality, the dataset encompasses 32,000 entries, evenly split with 50% being clickbait headlines and the remaining 50% being non-clickbait headlines.



	headline	clickbait
0	Should I Get Bings	1
1	Which TV Female Friend Group Do You Belong In	1
2	The New "Star Wars: The Force Awakens" Trailer...	1
3	This Vine Of New York On "Celebrity Big Brothe...	1
4	A Couple Did A Stunning Photo Shoot With Their...	1
5	How To Flirt With Queer Girls Without Making A...	1
6	32 Cute Things To Distract From Your Awkward T...	1
7	If Disney Princesses Were From Florida	1
8	What's A Quote Or Lyric That Best Describes Yo...	1
9	Natalie Dormer And Sam Claflin Play A Game To ...	1

Figure 1: Data Structure

3 Exploratory Data Analysis

Exploratory Data Analysis (EDA) constitutes a crucial phase within the Text Mining workflow. This process is instrumental in gaining a deeper comprehension of the dataset, revealing concealed patterns and trends. EDA serves the purpose of uncovering potential anomalies or issues within the data. Prioritizing an initial understanding of the data and extracting valuable insights is considered a best practice in the field.

3.1 Missing Values

The initial examination focused on the presence of missing values. We are pleased to confirm that no missing values were identified in the dataset. This condition enhances the robustness of our analysis, as all necessary data has been provided for each entry in the dataset.

3.2 Visualization of Data

Our subsequent analysis was about the distribution of clickbait to ascertain whether the classes were indeed balanced. We aimed to understand the proportion of clickbait headlines compared to non-clickbait headlines within the dataset. This investigation is crucial for gaining insights into the overall composition of the data and ensuring a balanced representation of both classes. The findings of this analysis contribute to the interpretation and reliability of our results, providing a foundation for subsequent steps in the further processes.

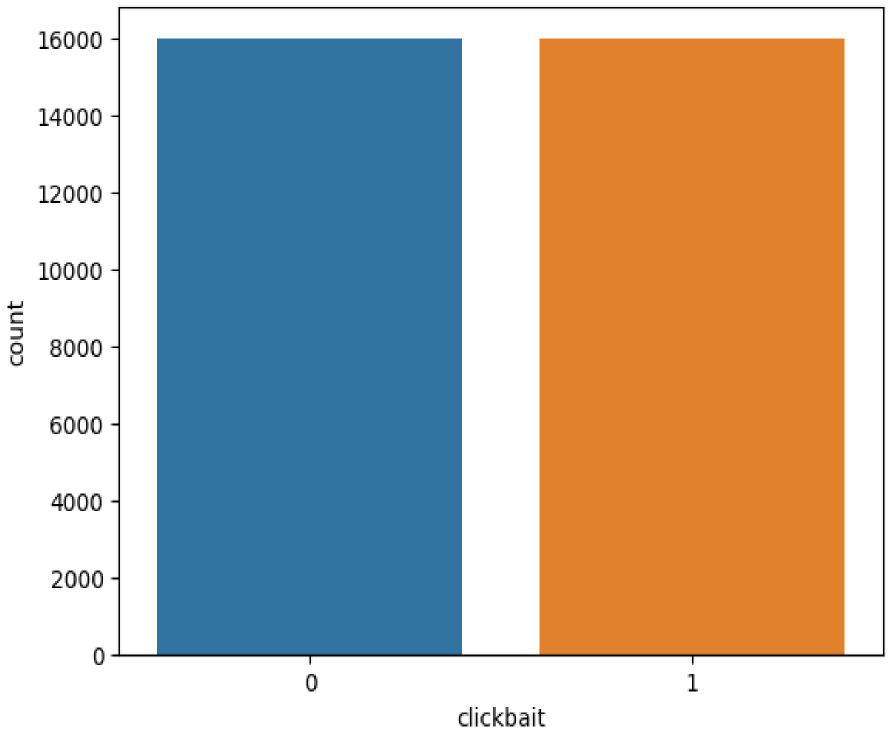


Figure 2: Distribution of Clickbaits and non-Clickbaits

3.3 Initial Findings

Findings:

- The dataset has 32000 rows with 2 columns.
- No missing values in all the columns.
- Dataset is balanced in number of Clickbaits and non-Clickbaits.

4 Data Pre-Processing

In order to enhance the quality of our analysis, we conducted a series of preprocessing steps on the Clickbait Dataset. These steps aimed to clean and standardize the textual data, ensuring a more uniform and meaningful representation.

To prepare the textual data for Text Mining, a text preprocessing pipeline is implemented. This includes lowercasing, removal of non-alphanumeric characters, tokenization, stopword removal, and stemming. Examples of original and preprocessed titles, later on, will demonstrate the effectiveness of this process.

The following key preprocessing steps were applied:

4.1 Removal of Unwanted Characters

Unnecessary characters, such as symbols and special characters, were removed from the headlines to ensure that only relevant textual information remains for analysis.

4.2 Conversion to Lowercase

All text was converted to lowercase to maintain consistency and eliminate the impact of case variations on subsequent analyses.

4.3 Stopword Elimination

Common stopwords, which do not contribute significantly to the meaning of the text, were removed. This step helps focus the analysis on more meaningful terms.

4.4 Stemming

Stemming, performed using the NLTK library, involved reducing words to their root or base form. This step aids in standardizing words and reducing dimensionality, thereby improving the efficiency of subsequent analyses.

4.5 Example of Preprocessed Sentences

1. **Original:** This is a Clickbait Title for sure!
   Preprocessed: clickbait titl sure
2. **Original:** How to Do Text Preprocessing?
   Preprocessed: text preprocess
3. **Original:** This is for Text Mining and Search Exam...
   Preprocessed: text mine search exam

5 Introduction to Text Classification

The initial part of our report is about the Text Classification.

We started splitting the dataset:

5.1 Train and Test Set

In the process of building Machine Learning models, it is crucial to ensure that the model generalizes well to new, unseen data. To simulate this scenario, the existing dataset undergoes a procedure known as data splitting. This involves dividing the dataset into two main portions: the Training Set, typically comprising 80% of the data, and the Testing Set.

The Training Set is the primary learning resource for the model. During this phase, the model learns patterns and relationships within the data. Following the training phase, the testing set comes into play. This set is employed to evaluate the model's accuracy and performance on data it has not encountered before. By assessing the model on this independent dataset, we gain insights into its ability to make accurate predictions on new, unseen data, thus gauging its overall effectiveness and generalization capability.

5.2 Text Vectorization

Word Embeddings, or Word Vectorization, is a method in **Natural Language Processing (NLP)** that involves mapping words or phrases from a given vocabulary to corresponding vectors of real numbers. These numerical representations enable algorithms to perform tasks such as word predictions, assess word similarities, and capture semantic relationships.

In simpler terms, the process of converting words into numerical vectors is known as **Vectorization**. This step is crucial in enabling machine learning models to analyze and draw insights from textual data, facilitating tasks such as language understanding and prediction.

TF-IDF vectorization is applied to convert textual data into numerical features.

5.2.1 Term Frequency-Inverse Document Frequency

In the context of natural language processing, particularly when employing a **TF-IDF (Term Frequency-Inverse Document Frequency)** approach, the goal remains to transform text into a format suitable for machine learning and deep learning applications.

In this project,**TF-IDF** approach is used and it involves in assigning weights to words based on their frequency in a specific document relative to their frequency across the entire dataset. This technique is utilized to create numerical representations of words or phrases, allowing algorithms to understand and analyze textual information.

In essence, the TF-IDF method is a way of capturing the importance of words within a document, emphasizing those terms that are more unique or distinctive. This approach plays a vital role in text vectorization, providing a numerical foundation for machine learning models to interpret and draw insights from natural language data.

6 Logistic Regression Classifier

In the preliminary stages of classification, we used a Linear Regression Classifier. The objective was to explore the feasibility of employing this specific machine learning algorithm to discern patterns and relationships within the textual data.

The classifier aimed to delineate distinctions between **Clickbaits** and **non-Clickbaits** of text based on the features derived from the TF-IDF representation of the textual content. This initial approach allowed us to assess the baseline performance and gain insights into the suitability of a linear regression-based model for our text classification objectives.

6.1 Results and Performance Evaluation

The Clickbait Detection Model was trained on **Train Dataset** and performed on**Test Dataset** Ṫhe performance metrics on the test set are as follows:

- **Accuracy:** 0.95125
- **Precision:** 0.9716
- **Recall:** 0.9319
- **F1-score:** 0.9513

Evaluation Metrics are:

- **Accuracy** represents the overall correctness of the model's predictions. In this case, the model is correct approximately 95.13% of the time on the test set.
- **Precision** measures the accuracy of positive predictions made by the model. In this context, it indicates that around 97.16% of the headlines predicted as clickbait by the model are indeed clickbait.
- **Recall** or **Sensitivity** measures the ability of the model to capture all the relevant instances. Here, it signifies that the model correctly identifies approximately 93.19% of all actual clickbait headlines.
- **F1-Score** is the harmonic mean of precision and recall. It provides a balanced measure that considers both false positives and false negatives. In this case, the F1-score is 95.13%, indicating a good balance between precision and recall.

6.2 Test the Classifier

We tested the Linear Regregression Classifier Model made, with an invented sentence:

- *Amazing secrets to pass Text Mining exam in just 3 days!*

The result of the Classifier Model was **1** = **Clickbait**.

Figure 3: Clickbait Title example for a YouTube video

7 Decision Tree Classifier

In order to check for a better accuracy, if possible, we also trained a Decision Tree Classifier and tested the model.

A **Decision Tree Classifier** is a versatile tool for making decisions or predictions based on input data. At its core, a decision tree comprises nodes, representing decision points, and branches, depicting the potential outcomes based on different conditions. The process starts at the top, known as the root node, and progresses down to the bottom, culminating in leaf nodes that provide the final predictions.

7.1 Results and Performance Evaluation

The Clickbait Detection Model was trained on **Train Dataset** and performed on**Test Dataset** The **accuracy** achieved here is of **90%**.

7.2 Considerations

After built two different Classifier for this Clickbait problem we achieved a **95,12% Accuracy** with the **Linear Regression Classifier** so we accepted the good result.

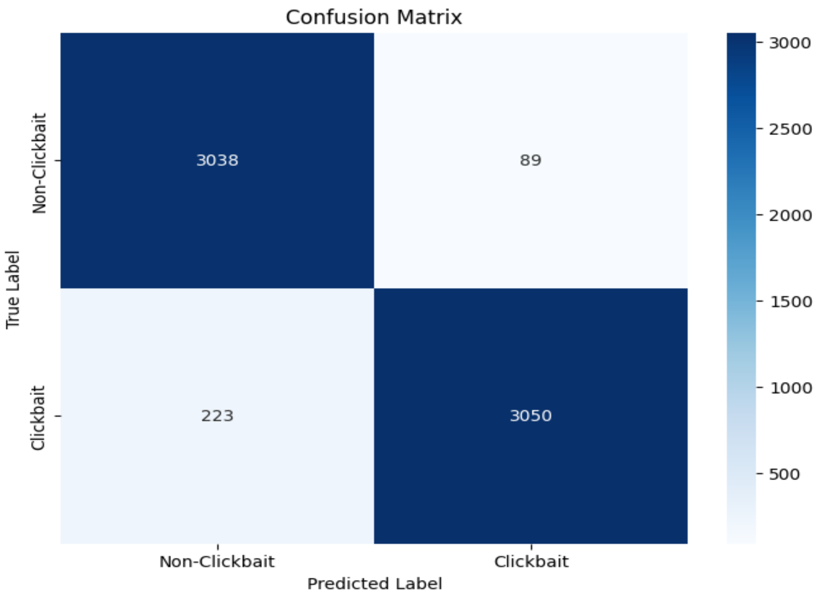


Figure 4: Confusion Matrix of Linear Regression Classifier.

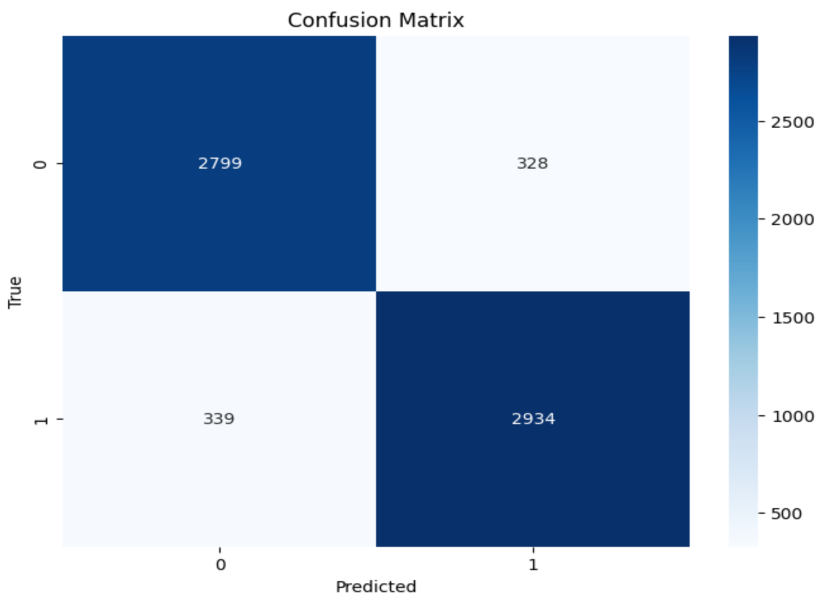


Figure 5: Confusion Matrix of Decision Tree Classifier.

8 Introduction to Text Clustering

The final section of this project is about the **Text Clustering**. In this section, we presented Text Clustering techniques on the provided dataset of news headlines. The primary goal was to discern underlying patterns within the headlines using different clustering algorithms.

In addition to supervised learning, the report explores unsupervised learning through **KMeans Clustering**. The TF-IDF vectors of headlines are used to group them into clusters. Evaluation metrics like **V-Measure and Adjusted Rand Index** quantify the clustering effectiveness. A 2D PCA visualization aids in understanding the separation of clusters.

8.1 KMeans Algorithm

Our approach involved applying the **KMeans Clustering** algorithm with the assumption of two clusters, corresponding to clickbait and non-clickbait headlines. The model was trained on the TF-IDF transformed training data, and the resulting clusters were analyzed for their coherence.

8.2 Number of Clusters

To determine the optimal number of clusters in KMeans, the **Elbow Method** is applied. The inertia values for different cluster numbers are plotted, providing valuable insights into the optimal k value.

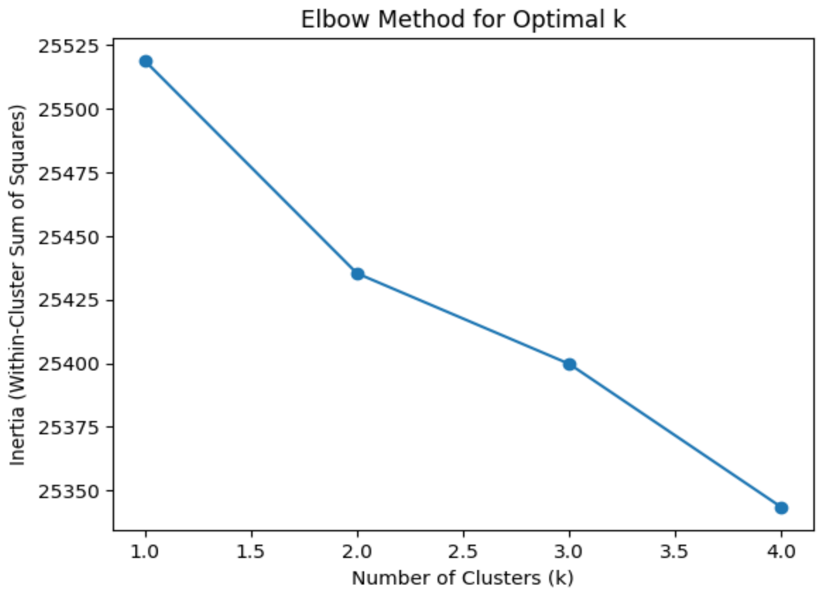


Figure 6: Elbow Method for Optimal Number of Clusters.

8.3 Dimensionality Reduction

To gain further insights into the structure of the TF-IDF transformed data, we used **Principal Component Analysis (PCA)** for dimensionality reduction. The primary objective was to visualize the distribution of headlines in a lower-dimensional space (2 dimensions) and assess whether clusters become more separated.

8.3.1 Transformation and Visualization

The TF-IDF vectors, representing the headlines, were transformed using PCA into two principal components. This reduction to two dimensions allowed for easy visualization of the data. The resulting scatter plot depicted each headline point in the new PCA space, with colors indicating the clusters obtained through KMeans.

The scatter plot revealed the distribution of headlines in the reduced space. While the clusters were not perfectly separated, there was some visible grouping of points. The colors on the plot corresponded to the clusters obtained from KMeans, providing a visual representation of the algorithm's performance.

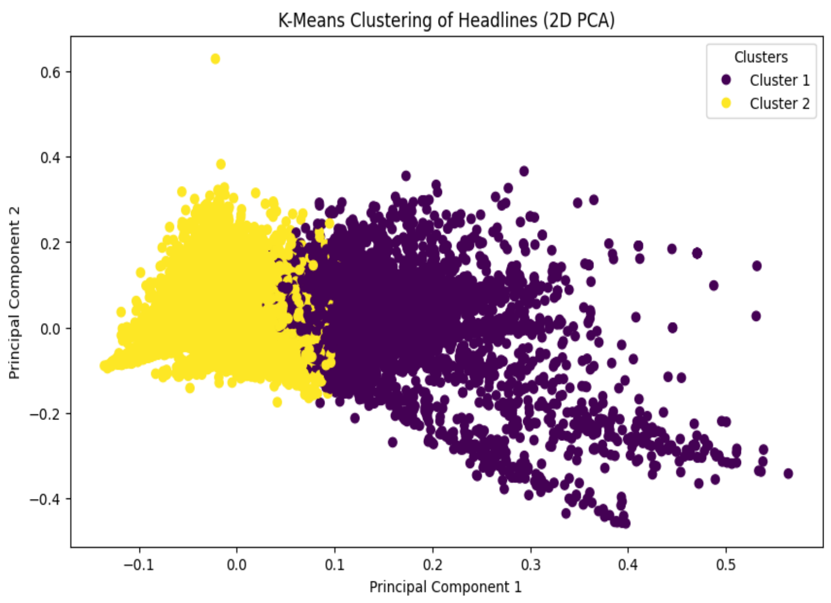


Figure 7: KMeans Clustering of Headlines in 2D PCA.

8.4 Evaluation Metrics for Clustering

In the application of **KMeans Clustering**, the analysis includes an evaluation of the clustering results using **Adjusted Rand Index (ARI)** and **V-Measure Score**.

An **ARI of 0.13** is indicative of a relatively low similarity between the true labels and the predicted clusters. This suggests that the clustering algorithm may not perfectly align with the actual clickbait and non-clickbait labels.

Furthermore, the **V-Measure Score of 0.24** means that the clustering has some degree of agreement with the true labels. However, the score suggests that the agreement may not be very high, emphasizing the challenges in achieving precise clustering of clickbait headlines.

The modest V-Measure Score implies that there might be inherent complexities in grouping headlines based on their content, and further refinement or alternative clustering approaches could be explored.

9 Future Applications and Conclusions

Further exploration may involve experimenting with alternative **Text Embeddings** or combining multiple feature representations.

The integration of PCA into the analysis contributes to a better understanding of the data structure, emphasizing the importance of preprocessing and feature engineering in the text clustering process.

This project provides a comprehensive analysis of clickbait detection techniques, emphasizing the importance of Exploratory Data Analysis, Text Preprocessing, and Text Mining techniques. The **Logistic Regression and Decision Tree Classifiers** demonstrate robust performance, while **KMeans Clustering** offers insights into potential patterns within the data.

Further research could explore advanced models or ensemble methods for enhanced clickbait detection. Overall, the techniques presented in this project form a strong foundation for developing effective clickbait detection systems.

References

[1] “Clickbait Classification.” Techniques for Binary Text Classification Labels.

[2] “Binary Text Preprocessing”. Improving binary classification on text problems using differential word features.

[3] “A critical review of k means text clustering algorithms”. International Journal of Advanced Research in Computer Science, 2013, 4.9.