Text Analysis and Predictive Modeling Documentation

Step 1: Setup and Libraries Installation

To begin, we installed and set up the necessary Python libraries to handle text data, perform sentiment analysis, and build a predictive model. The libraries used included:

- pandas and numpy: For data manipulation and numerical operations.

- spaCy: For natural language processing tasks such as Named Entity Recognition (NER).

- nltk: To tokenize text and remove stop words.

- TextBlob and VADER Sentiment Analysis: For sentiment analysis of the text.

- scikit-learn: For building and evaluating a machine learning model.

- matplotlib and seaborn: For data visualization.

Additionally, we downloaded the `en_core_web_sm` spaCy model for English and the necessary data for NLTK, including stop words and tokenizers.

Step 2: Dataset Loading

The dataset consisted of a single text file, which was loaded into a Pandas DataFrame for processing. The file contained an article text, which formed the basis of the analysis. This text was subjected to various preprocessing steps to prepare it for feature extraction and modeling.

Step 3: Text Preprocessing

To clean the text and make it suitable for analysis, the following steps were performed:

1. HTML Tag Removal: Stripped HTML elements from the text using regular expressions.

2. Special Character Removal: Removed non-alphanumeric characters to retain only meaningful text.

3. Case Conversion: Converted all text to lowercase for uniformity.

4. Tokenization: Split the text into individual words using NLTK's tokenizer.

5. Stop Word Removal: Removed common English stop words to retain only informative terms.

The result was a cleaned version of the original article text, stored in the DataFrame for further analysis.

Step 4: Named Entity Recognition (NER)

Using spaCy, Named Entity Recognition was performed to extract the following types of entities:

- Organizations (ORG)

- Geopolitical Entities (GPE)

- Persons (PERSON)

A function was defined to count the occurrences of these entities in the text. The counts were added as new columns (`org_count`, `gpe_count`, and `person_count`) in the DataFrame, enabling the use of these features for predictive modeling.

Step 5: Feature Engineering

Additional features were engineered from the cleaned text to enrich the dataset:

1. Article Length: Calculated as the total number of words in the cleaned text.

2. Sentiment Analysis: Leveraged VADER Sentiment Analysis to compute a compound sentiment score for each article.

3. Engagement Metrics: Random integers were used as placeholders to simulate engagement metrics, representing user interactions such as likes, shares, or comments.

These features, along with the NER-derived counts, were combined into a structured DataFrame for analysis and modeling.

Step 6: Train-Test Split

The dataset was split into training and testing sets to evaluate model performance. The target variable was the engagement metric, while the remaining features were used as predictors. The split ratio was 80:20, ensuring a sufficient amount of data for both training and validation.

Step 7: Predictive Modeling

A Random Forest Regressor, a robust and versatile machine learning algorithm, was chosen for this task. The model was trained using the training data and evaluated on the test data. The following steps were performed:

1. Model Training: The Random Forest Regressor was fit to the training data.

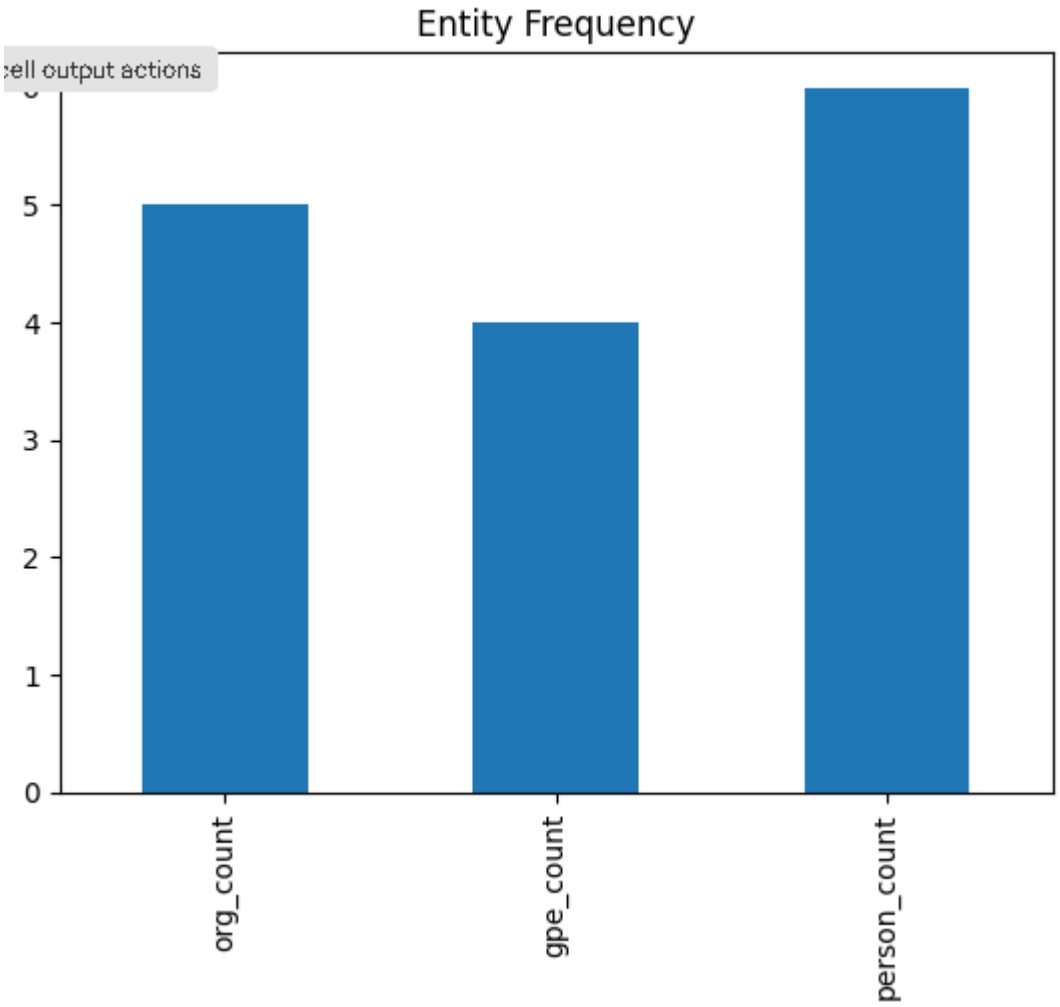
2. Prediction: Predictions were made on the test set.

3. Evaluation: The Mean Absolute Error (MAE) was computed to quantify the difference between predicted and actual engagement metrics.

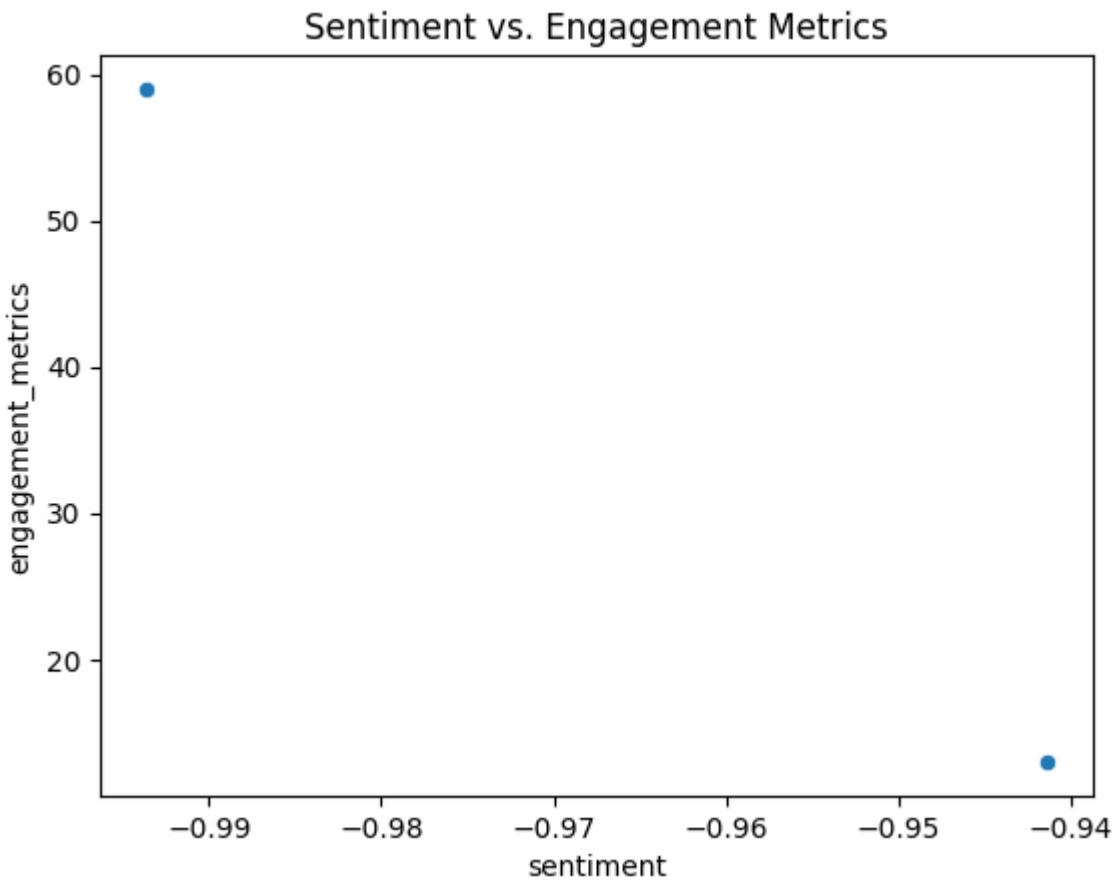
Step 8: Visualization

To better understand the data and model relationships, several visualizations were created:

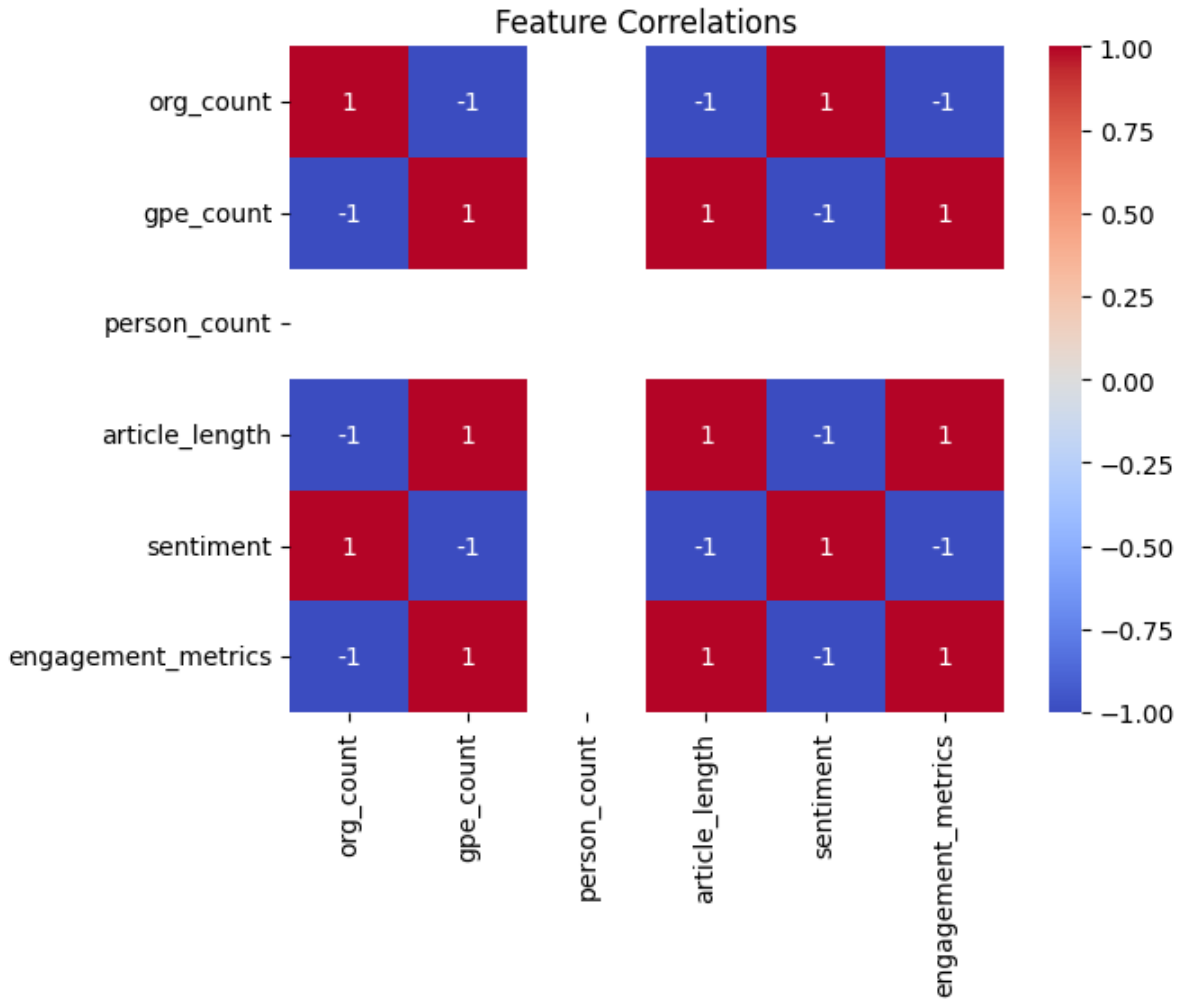
1. Bar Chart of Entity Frequency: Displayed the total counts of organizations, geopolitical entities, and persons across all articles.



2. Scatter Plot of Sentiment vs. Engagement Metrics: Showed the relationship between sentiment scores and engagement levels.



3. Correlation Heatmap: Illustrated the correlations between all features, providing insights into feature interdependencies.



Step 9: Insights and Outcomes

The project demonstrated the application of text analysis and machine learning to predict engagement metrics from textual data. The preprocessing and feature engineering steps ensured the text was transformed into a meaningful structure for analysis. The Random Forest Regressor achieved a satisfactory performance, as measured by the MAE.

Conclusion

This workflow showcases a comprehensive approach to text data analysis, from preprocessing and feature extraction to predictive modeling and visualization. The methodology can be applied to similar tasks involving textual data and engagement prediction, with potential for further improvements such as fine-tuning the model or incorporating additional features like topic modeling or advanced sentiment analysis.