Real-Time Social Media (Twitter) Sentiment Analysis for Brand Reputation Management

Introduction

In today's digital age, social media platforms serve as a rich source of real-time information, providing insights into public opinion and sentiment on various topics. Among these platforms, Twitter stands out as a key player, allowing users to express their thoughts and emotions in concise messages. This project focuses on conducting a sentiment analysis of tweets.

The primary objective of this project is to analyze a substantial dataset of tweets, utilizing advanced **Natural Language Processing (NLP)** techniques to categorize sentiments into positive, negative, and neutral.

Through a structured workflow that includes data collection, preprocessing, visualization, and modeling, this analysis not only sheds light on audience reactions but also demonstrates the practical application of data science techniques in understanding social media dynamics. Ultimately, the insights gained from this project can serve as a foundation for further research and applications in sentiment analysis within the broader context of social media analytics.

Preprocessing

The preprocessing code focuses on cleaning and standardizing text data, specifically for social media content like tweets. It includes several regular expressions (regex) and functions to handle the following aspects:

a. **Hashtags:** Extracts and processes hashtags in tweets.
b. **Handles:** Identifies and processes Twitter handles (e.g., `@username`).
c. **URLs:** Detects URLs within the text and standardizes them.
d. **Word boundaries:** Splits text into words using word boundaries.
e. **Repeating Characters:** Reduces elongated words (e.g., "soooo" becomes "soo").
f. **Emoticons:** Recognizes and categorizes various emoticons, such as smileys and frownies.
g. **Punctuations:** Handles punctuation marks and replaces them with predefined tags.
h. **Query Terms:** Detects specific query terms and replaces them with placeholders.

Functions are defined to process text in a structured manner:

a. **processAll:** Combines all preprocessing steps into a single function that handles hashtags, handles, URLs, emoticons, word boundaries, and repeating characters.
b. **Individual Processing Functions:** Separate functions for handling hashtags, handles, URLs, emoticons, punctuations, and repeating words.

Twitter Features

This section involves extracting features from tweets and converting them into feature vectors. It uses predefined patterns to detect specific words or phrases that may indicate sentiment or emotions.

Feature Detection: The code identifies various features such as:

- Words indicating positive or negative sentiments (e.g., "awesome," "broken," "happy," "hate").
- Emojis and expressions that imply specific emotions (e.g., `:D` for happiness, `:(` for sadness).

Functions provided in this section include:

- **make_tweet_nparr:** Converts the tweet text into a NumPy array representing the feature vector.
- **make_tweet_dict:** Converts the tweet text into a dictionary format, indicating the presence of specific features.
- **Conversion Functions:** Includes utility functions to convert between dictionary and NumPy array formats for the feature vectors.
- **is_zero_dict:** Checks if the feature vector contains any identified features or if it is empty.

The primary focus is on preparing raw tweet data for analysis by standardizing and cleaning it, followed by extracting specific features that might be useful for sentiment analysis or other natural language processing tasks. The preprocessing steps handle text normalization, while the feature extraction identifies words, emoticons, and phrases that indicate emotions or opinions. This approach helps convert unstructured text data into structured feature vectors, making it suitable for further analysis or machine learning models.

Twitter Sentiment Analysis

The analysis can be summarized into several key sections that detail the process of performing sentiment analysis on tweets related to "Money Heist".

1. Libraries and API Setup

Libraries Imported –

- **TextBlob:** For sentiment analysis.
- **Tweepy:** For accessing the Twitter API.
- **Matplotlib:** For plotting data visualizations.
- **Pandas:** For data manipulation and analysis.
- **WordCloud:** For visualizing word frequencies.
- **Nltk:** For natural language processing tasks.

Twitter API Configuration –

The Jupyter notebook uses authentication keys to connect to the Twitter API:

- Consumer Key
- Consumer Secret
- Access Token
- Access Token Secret

This section sets up the environment necessary to fetch tweets related to the search term "Money Heist".

2. Fetching Tweets

Tweet Retrieval –

- Search Term: "Money Heist"
- Number of Tweets: 1000
- Tweets are fetched using **`tweepy.Cursor`**, which handles pagination and collects tweets matching the search term.

Sentiment Analysis on Tweets –

A loop iterates through each tweet to analyze sentiment using **TextBlob**:

- Polarity: Ranges from -1 (negative) to +1 (positive).

Counts are maintained for:

- Positive tweets
- Negative tweets
- Neutral tweets

Polarity Calculation –

- A function calculates the percentage of positive, negative, and neutral sentiments based on the total number of tweets.

Output Results –

- The sentiment results are printed, and a pie chart visualizes the distribution of sentiments among the tweets.

3. Loading and Preparing Data

Dataset Loading –

- Two CSV files are loaded: **`train_tweet.csv`** and **`test_tweets.csv`.**
- The shapes of both datasets are printed for verification.

Data Inspection –

- Null values in the datasets are checked, and sample tweets from both positive and negative labels are displayed.

Visualization –

- A bar plot visualizes the distribution of tweets based on sentiment labels.

4. Data Preprocessing

Tweet Length Analysis

- Histograms visualize the length distribution of tweets in both the training and test datasets.
- New columns representing tweet lengths are added to the datasets.

Word Cloud Generation

- A word cloud visualizes the vocabulary used in the reviews, created from the frequency of words.

5. Hashtag Analysis

Extracting Hashtags

- A function extracts hashtags from tweets.
- Hashtags from both regular (neutral) and negative tweets are collected and visualized using bar plots.

6. Data Cleaning and Preparation

Text Preprocessing

- Unwanted characters are removed from tweets.
- Tweets are converted to lowercase, tokenized, and stemmed using **PorterStemmer.**
- Stopwords are removed to focus on meaningful words.

Creating Bag of Words

- The CountVectorizer is used to create a bag-of-words model from the cleaned training and test datasets.

7. Model Training and Evaluation

Splitting Data

- The training data is split into training and validation sets.

Standardization

- Standardization of features is applied using StandardScaler.

Model Training

Several machine learning models are trained and evaluated:

- Random Forest Classifier
- Logistic Regression
- Decision Tree Classifier
- Support Vector Classifier (SVC)
- XGBoost Classifier

Model Evaluation Metrics

- Training and validation accuracies are printed for each model.
- The F1 score and confusion matrix are computed to assess model performance.

Summary of Results

The notebook effectively implements a comprehensive workflow for sentiment analysis on tweets related to "Money Heist". Sentiment analysis identifies public reactions, while the modeling phase compares various machine learning algorithms to classify sentiments. Visualizations provide insights into the data distribution, sentiment proportions, and common hashtags.

Final Remarks

The analysis effectively demonstrates the ability to gather real-time Twitter data, perform sentiment analysis, preprocess textual data, and apply various classification algorithms to predict sentiment accurately.

The final choice of the best model can be made based on evaluation metrics, particularly focusing on the F1 score and confusion matrix results for a more nuanced understanding of model performance.