Exploring Sentiment Analysis on Social Media Data Using Machine Learning Techniques

Name: Pratheek KB

Contact Information: pratheekkb93@gmail.com

Date: 17/3/2024

ABSTRACT:

This paper explores sentiment analysis on social media data using machine learning techniques. The dataset analyzed consists of tweets collected from various social media platforms. We employed natural language processing and machine learning algorithms to classify the sentiment of the tweets. The key findings reveal insights into the prevailing sentiments expressed on social media platforms.

INTRODUCTION:

The project deals with building machine learning models to predict the sentiment of the tweets where we want to figure out how strongly someone feels a particular emotion, called X, when they write a tweet. We'll give their feeling a score between 0 and 1. If they feel a lot of emotion X, they'll get a higher score closer to 1. If they don't feel much of it, they'll get a lower score closer to 0. These scores help us understand how much emotion X is present in the tweet.

Data set:

Source : https://saifmohammad.com/WebPages/EmotionIntensity-SharedTask.html

Format : text file

Training and test datasets are provided for four emotions: joy, sadness, fear, and anger. For example, the anger training dataset has tweets along with a real-valued score between 0 and 1 indicating the degree of anger felt by the speaker. The test data includes only the tweet text.

The analysis relied on several Python packages to explore and analyze the dataset. These packages include Pandas for managing data, NumPy for numerical computations, Matplotlib and Seaborn for creating visualizations, and NLTK for text processing tasks such as tokenization and removing unnecessary words. Additionally, scikit-learn was utilized for various operations including counting words and making predictions, while Tensorflow and Keras were employed for building and training deep learning models. Further support for data preprocessing and normalization was provided by packages like StandardScaler.

Two models are built:

1) Linear Regression with and without regularization
2) Multilayer Neural Network

Data Analysis:

Relationship between Word Count and emoji count with Intensity

The created datasets were additionally added two new features: word_count and emoji_count , to analyse the effect of their count on the sentiments of the tweet.

The regex library provides advanced regular expression capabilities, including support for Unicode characters like emojis.

From the plots its observed that the word count scatter was uniformly distributed across all intensities however the number of tweets with increasing number of emojis in a tweet decreasing intensities which is contrary to our intuition

Identifying trending handles :

The top 10 most frequently mentioned Twitter handles are identified from the dataset. Then, for each of these top mentions, the code counts the number of tweets containing that mention and categorizes them by their sentiment labels (e.g., joy, sadness, anger, fear).

The results are visualized using a bar plot, where each bar represents the count of tweets mentioning a specific handle. The bars are segmented by sentiment labels, with different colours indicating different sentiments. This visualization provides insights into the distribution of mentions across sentiment categories for the top Twitter handles.

From the plots its observable that the handle @realDonaldTrump and @HillaryClinton appeared more on the tweets and most of the tweets belonged to emotion fear, anger and sadness and showing the tweets are posted during American presidential elections time

Similarly plot of AirBNB shows that certain people feared for lack of privacy, and some really enjoyed their holiday stay.

Similarly, lot of inferences can be drawn from the plots for different mentions.

Identifying most used hashtags in respective emotion category :

A bar plot is generated for all four emotions showing top 10 common hashtags used by a user with their count.

Data Preprocessing:

Tokenization and Stemming:

Tokenization is the process of breaking down text into individual words or tokens. We employed the WordPunctTokenizer from the Natural Language Toolkit (NLTK) library to tokenize each tweet. Additionally, stemming was applied to reduce words to their root or base form, which helps in reducing the dimensionality of the feature space.

Removal of Mentions, Links, and Digits:

To clean the tweet text, we removed Twitter mentions (usernames prefixed with '@'), hyperlinks, and digits. These elements typically do not contribute to the sentiment or content analysis and are thus excluded from further processing.

Stopword Removal:

Stopwords, which are common words like "the," "is," and "and," were removed from the tokenized text to improve the quality of the analysis. We utilized the English stopwords provided by the NLTK library for this purpose.

TF-IDF Vectorization:

After preprocessing, we transformed the cleaned tweet data into TF-IDF (Term Frequency-Inverse Document Frequency) vectors. TF-IDF is a numerical statistic that reflects the importance of a word in a document relative to a collection of documents. We employed the TfidfVectorizer from the scikit-learn library to generate TF-IDF features, limiting the maximum number of features to 1000 to manage computational complexity and improve model efficiency.

Model Training :

Linear Regression:

The linear regression model was trained using TF-IDF vectorized features derived from the cleaned tweets. The trained model was evaluated on both development and test datasets to assess its performance in predicting the intensity of emotions conveyed in the tweets. Here are the performance metrics for the linear regression model:

Training Set Performance:

- Mean Absolute Error (MAE): 0.1016
- Root Mean Squared Error (RMSE): 0.1297

Development Set Performance:

- Mean Absolute Error (MAE): 0.1611
- Root Mean Squared Error (RMSE): 0.2040

Test Set Performance:

- Mean Absolute Error (MAE): 0.1552
- Root Mean Squared Error (RMSE): 0.1957

Cross-Validation Results

Additionally, a cross-validation procedure was conducted to validate the model's generalization performance. The model was trained on the combined training and development datasets and evaluated using 5-fold cross-validation. The following are the cross-validation results:

- Mean MAE: 0.2091
- Mean RMSE: 0.2595
- MAE Standard Deviation: 0.0129
- RMSE Standard Deviation: 0.0119

Ridge Regression Model Performance

Furthermore, a Ridge regression model was trained using TF-IDF vectorized features with an L2 regularization parameter (alpha=5). Similar evaluation metrics were computed for the Ridge regression model:

Training Set Performance:

- Mean Absolute Error (MAE): 0.1199
- Root Mean Squared Error (RMSE): 0.1499

Development Set Performance:

- Mean Absolute Error (MAE): 0.1378
- Root Mean Squared Error (RMSE): 0.1721

Test Set Performance:

- Mean Absolute Error (MAE): 0.1421
- Root Mean Squared Error (RMSE): 0.1760

Mulitlayer Neural Network:

A neural network model was developed using TensorFlow and Keras to predict the intensity of emotions conveyed in tweets. Here's an overview of the model architecture and its performance:

Data Preparation:

- The tweet texts were tokenized and converted into sequences of integers using a tokenizer.
- Padding was applied to ensure all sequences have the same length, using the maximum sequence length in the training set.

Model Architecture:

- **Input Layer**: Dense layer with 64 units and the sigmoid activation function, taking input of shape **(max_len,)**.
- **Hidden Layer**: Dense layer with 64 units and the sigmoid activation function.
- **Output Layer**: Dense layer with 1 unit and the sigmoid activation function.

Model Compilation:

- The model was compiled using the Adam optimizer and Mean Squared Error (MSE) loss function.
- Mean Absolute Error (MAE) was used as a metric to evaluate the model during training.

Model Training:

- The model was trained for 10,000 epochs with a batch size of 50.
- Training and development datasets were used for training, with validation performed on the development set.

Model Evaluation:

- After training, the model was evaluated on the test dataset to assess its performance.
- The Mean Absolute Error (MAE) was computed to measure the difference between the predicted and actual intensity labels.
- Additionally, the percentage of absolute differences less than or equal to 0.1 (with tolerance) was calculated.

Performance Metrics:

- **Test MAE:** The Mean Absolute Error (MAE) achieved on the test dataset.
- **Percentage of Absolute Differences <= 0.1:** The percentage of absolute differences between predicted and actual labels that fall within a tolerance of 0.1.

Results and Analysis:

- The model achieved a Test MAE of 0.07825274765491486.
- 76.197 % of the absolute differences between predicted and actual labels were less than or equal to 0.1.