Twitter_Sentiment_Analysis Public

1 Branch 0 Tags Go to file t Go to file + Add file Code ...

GraceMwende update README.md f0a21d7 · 4 minutes ago

.ipynb_checkpoints	presentation touchup	1 hour ago
data	modeling	3 days ago
.gitignore	finish up project summary	yesterday
README.md	update README.md	4 minutes ago
TwitterSentimentA...	presentation touchup	1 hour ago
function before.txt	add roc_auc score	3 days ago
sentiment_v1.ipynb	first commit	4 days ago
sentiment_v2.ipynb	first commit	4 days ago
sentiment_v3.ipynb	m	3 days ago
twitter_Sentiment...	m	3 days ago
twitter_Sentiment...	summary	yesterday

About

NLP model to analyze Twitter sentiment and rate the sentiment of a Tweet based on its content.

Readme

Activity

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

twitter_Sentiment... presentation touchup 1 hour ago

Jupyter Notebook 100.0%

README

#Twitter_Sentiment_Analysis

Summary

Dataset Overview

This dataset contains information about tweets, including their date, classification, and other relevant features. For this task, I extracted only two columns:

- **Feature Variable (Text/Tweets)**: The actual tweet content.
- **Sentiment**: The rating of each tweet, categorized as negative, positive, or neutral.

Data Preparation

The data preparation phase included **data cleaning** and **data preprocessing** using the pandas and NLTK libraries.

1. Data Cleaning (Using Pandas)

- The dataset had **no missing values**, so no imputation was required.
- **Removed duplicate tweets**—for instance, one tweet appeared 304 times.

- Dropped irrelevant sentiment labels**, such as 'not_relevant', to retain only the core sentiments **(negative, positive, and neutral)**.
- **Converted sentiments to integer values** to ensure compatibility with machine learning models.

2. Data Preprocessing (Using NLTK)

- **Removed hyperlinks, usernames, single-character words, and hashtags** (including their values) using **regular expressions**, as they do not meaningfully contribute to sentiment analysis.
- **Eliminated stopwords and punctuation ** using the **NLTK corpus library** for stopwords and the **string library** for punctuation, as these do not add significant meaning to sentences.
- **Applied lemmatization ** using the **WordNet Lemmatizer**, converting words to their root form (e.g., "running" → "run").

Data Visualization

To explore and understand the dataset, I used:

- **Seaborn's** countplot to visualize the distribution of target sentiment classes.
- **WordCloud** to generate a visual representation of the most common words in the dataset.

Modeling

- Used **Scikit-learn's model_selection** library to split the dataset into training and testing sets.
- Implemented **pipelines** to streamline **vectorization, SMOTE (Synthetic Minority Over-sampling Technique), and classification models**.
- Applied **TF-IDF Vectorizer** to convert text data into numerical representations.
- Used **SMOTE** to address **class imbalance**, as one sentiment category comprised **more than 50%** of the dataset.
- Evaluated various **machine learning algorithms** for classification to determine the best-performing model.
- Created **custom functions** to automate repetitive processes such as model fitting and prediction.

Evaluation Metrics

To assess model performance, I used the following evaluation metrics:

- **Accuracy Score**: The proportion of correctly classified instances out of the total instances.
- **Precision Score**: The ratio of correctly predicted positive instances to total predicted positive instances.
- **Recall Score**: The ratio of correctly predicted positive instances to actual positive instances in the dataset.
- **ROC Curve**: A graphical representation of the true positive rate versus the false positive rate.

Model Evaluation & Predictions

- The **test set** obtained from `train_test_split (X_test, y_test)` was used for model evaluation and making predictions.
- Additionally, I developed a **custom function** that allows classification of sentiment based on user input.