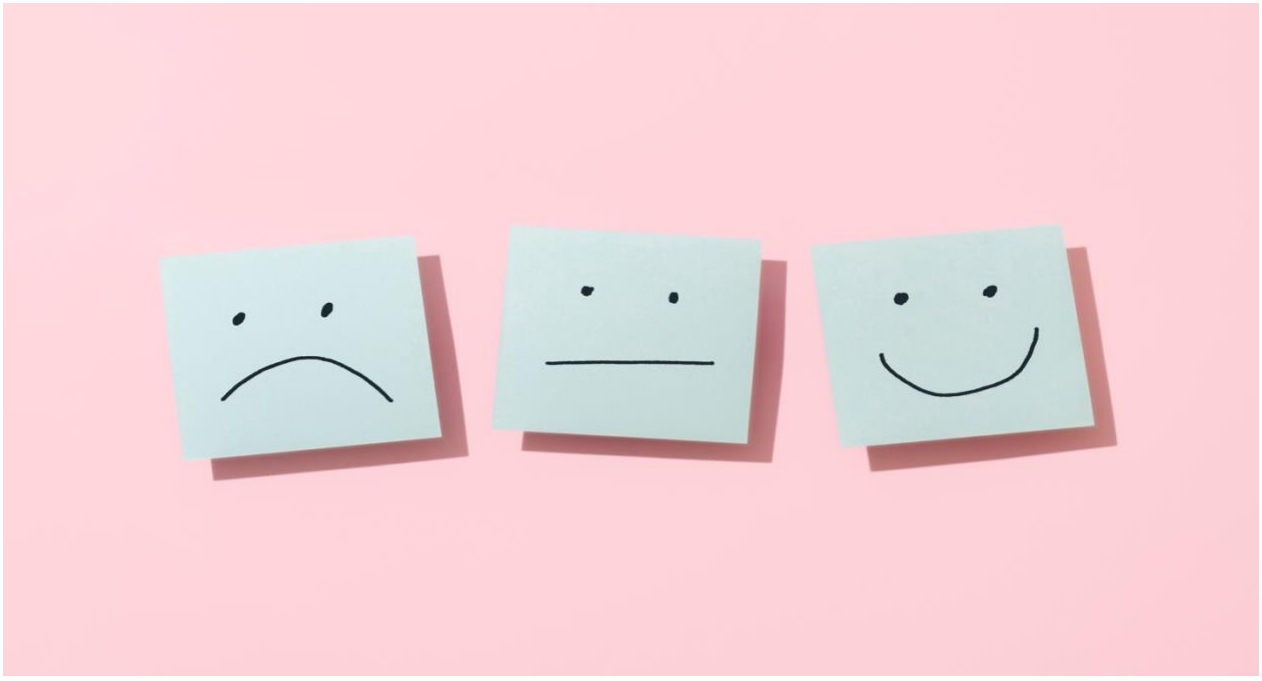
Twitter Sentiment Analysis: Classification Vs Deep Learning Approach



Project Write-up by: Himani Kaushik

Abstract

The purpose of the model is to analyze tweets and predict whether it expresses positive or negative sentiment. The model aims to implement and evaluate Naive Bayes model for classification and LSTM model for deep learning, and then select the better performing model for the client. A local client wants to implement a sentiment analysis model utilizing the tweets obtained about the product. This will help them detect angry customers or negative emotions before they escalate. The data was obtained from the Kaggle website. The dataset had 1.6 million tweets extracted using the twitter api. Since the dataset was too huge to work on a regular machine, I trimmed the dataset to $1/4^{th}$ of its original size. The data balance was maintained while trimming the data. The data was transformed and evaluated ysing Naive Bayes and LSTM model. LSTM mode was concluded to be the best model.

Design

The purpose of the model is to analyze tweets and predict whether it expresses positive or negative sentiment. The model aims to implement and evaluate Naive Bayes model for classification and LSTM model for deep learning, and then select the better performing model.

Data

The data was obtained from the kaggle website - https://www.kaggle.com/datasets/kazanova/sentiment140 . The dataset has 1.6 million tweets extracted using the twitter api. Since the dataset was too huge to work on a regular machine, I trimmed the dataset to $1/4^{th}$ of its original size. The data balance was maintained while trimming the data.
The data was cleaned and preprocessed using NLTK. I used TweetTokenizer to tokenize the data and split the text based on various criterions well suited for tweets. I used WordNetLemmatizer to lemmatize the text. WordCloud was used to show frequencies of positive and negative words.

Algorithms

- Data preprocessing:
    - Tokenization (TweetTokenizer)
    - Lemmatization (WordNetLemmatizer)
    - Cleaning
    - Removal of stop words
    - Custom fine-tuning
    - Transforming into dictionary structure
    - Visualizing (WordCloud)
- Naive Bayes Model
    - Naive Bayes Classifier
- LSTM Model:
    - Word Embeddings using GloVe

- Data Padding
- Sequential model
  - Embedding layer
  - Pair of Bidirectional LSTMs
  - Sigmoid layer
  - Binary cross-entropy loss function
  - ‘Adam’ optimizer
  - Accuracy metric
- Training Model
  - Batch size of 20 and epoch size of 20
- Dropout
  - LSTM model with a dropout rate of 40%
  - Two rounds of training
- Inspecting unknown words and further data cleaning
- Model built and trained on cleaned data

Tools

- Pandas and Numpy: For cleaning data and preprocessing.
- Keras, NLTK, scikit-learn: For processing data, Naive Bayes classification, GloVe embedding, LSTM
- Matplotlib, Wordcloud: For visualizing the data.

Communication

The information is presented in the pdf – ‘Twitter Sentiment Analysis’.