ISFANDYAR VIRANI

EMOTION DETECTOR

Introduction & Background

Emotion detection is a branch of sentiment analysis that deals with the extraction and analysis of emotions. Humans express emotions in their everyday lives when they communicate with each other. These emotions can be examined by cues such as gestures, text and facial expressions. The goal of this project is to analyze text corpora (text-based datasets) and build models that detect emotions by using natural language processing (NLP) and machine learning techniques.

Emotion detection has wide-ranging applications such as in public health (detecting depression), politics, brand management, psychology research, improving communication, helping people with Asperger's identify emotion, and much more. Emotion detection is a trendy topic in the data science community with BERT classifier being the industry-leading model for these kinds of NLP problems.

Data

The corpus (dataset) is made of 5 corpora.

1. Emotion Stimulus: The data are built from sentences containing emotions as well as factors that cause emotions. It contains 1594 emotion-labelled sentences and 820 sentences with both cause/stimulus and emotions. The emotion-labelled sentences are annotated in compliance with Ekman's basic emotion categories (Happiness, Sadness, Anger, Fear, Surprise, and Disgust) plus shame.
2. DailyDialog: This dataset was built by crawling dialogues from regular human conversations. It contains a total of 13 118 sentences annotated for neutral, anger, disgust, fear, happiness, sadness, surprise discrete emotion labels.
3. Emotions dataset for NLP (Kaggle): This dataset was posted on kaggle.com. It contains 20000 sentences split into 3 files (train.txt, test.txt & va.txt). The sentences are labelled with these emotions: joy, sadness, anger, fear, love, and surprise.
4. Emotion Detection from Text (Kaggle): The data is a collection of 39827 tweets annotated with the emotions behind them. It has three columns 'tweet_id,' 'sentiment,' which is the emotion behind the tweet, and 'content,' which is the raw tweet. The 'sentiment' column contains the following emotions: neutral, worry, happiness, sadness, love, surprise, fun, relief, hate, empty, enthusiasm, boredom, and anger.
5. ISEAR Dataset (Kaggle): This dataset is a subset of the original ISEAR dataset. This dataset has 6921 sentences labelled as the following emotions: fear, anger, joy, and sadness.

In total, 82289 sentences were labelled with their respective emotion in thecorpora.

To simplify this project, I will be focusing on Ekman's six core emotions: happy, sad, anger, fear, surprise, and disgust.

Preprocessing, feature engineering, data-cleaning/transformation, and exploratory data analysis (EDA)

Since there are 5 different corpora, I had to clean them individually and merge them as one single corpus. These corpora were either .txt file or .csv file. I performed these steps:

- For .txt file: Use Str.split() function to split the rows into 2 columns: emotion & text.
- Since I am focusing on Ekman's 6 core emotions, I excluded the other emotions.
- Renaming emotions (Example: converting joy & happiness into happy emotion).

I noticed that data is biased towards happy emotions, followed by sad emotions as these emotions far r exceeded the other emotions in frequency. I took the following actions to deal with this:

- Only used happy and sad emotions while importing data up to a certain threshold to ensure that equal data was available for each emotion.
- Used TextBlob to get sentiments and removed Negative & Neutral labelled sentiments from Happy emotion.
- Used TextBlob to get sentiments, and removed Positive & Neutral labelled sentiments from Sad emotion.

TextBlob sentiments helped make the dataset less biased as well as I removed Negative Happy and Positive Sad emotions.

I noticed that the data was still very biased. I decided to exclude theisgust emotion as I was having trouble finding data labelled disgust.

I selected a random sample of 4800 sentences from each emotion and made a new corpus of 24000 sentences with these random samples.

Tokenizing and Vectorizing

- Built a tokenizer that removed numbers, punctuation marks & non-letter symbols, converted text to lower case, split sentences into individual words, removed stop words, lemmatization and removed stemming from words.
- Use TF-IDF vectorizer on the corpus with tokenizer.

Scaling data

Split corpus into train and test sets, and scaled data with the following scaling techniques:

- **MinMaxScaler** - 75.92% test accuracy using vanilla logistic regression
- **StandardScaler** - 64.36% test accuracy using vanilla logistic regression
- **RobustScaler** - 76.75% test accuracy using vanilla logistic regression

I will be using **RobustScaler** for data for my models since it performed the best on my logistic regression tests.

Modelling and evaluation

I have trained **seven** different types of models. Each model is trained using scaled data. Recall and precision are important in this project as I want to avoid as many false positives and false negatives as possible. Model results are as follows:

Model	Accuracy	Precision	Recall
Multinomial Naive Bayes	72	72	72
Logistic Regression	77	77	77
K-Nearest-Neighbors	49	63	49
Support Vector Machines (SVM)	78	79	78
Decision Tree Classifier	73	73	73
Random Forest Classifier	78	79	78
XGB Classifier	77	79	77

The table above is showing accuracy, precision and recall on test data.

From the 7 models, **SVM** and **Random Forest Classifier** performed the best, followed by **XGB Classifier** and **Logistic Regression** which were just off by 1%. **KNN** performed the worst.

Insights

- Examining confusion matrices, the Surprise emotion seemed to have higher false-negative and false positives in the models.
- SVM took much longer to compute than any other classifier, while Random Forest Classifier, XGB Classifier, and Logistic Regression were much faster to train and have similar accuracy.
- The most frequent token/word in the cleaned corpus is 'feel' followed by 'realli', 'thank', 'like' and 'good'.
- The models have a hard time understanding sarcasm.
- Models can be tricked to give wrong emotion (common words from an emotion class can sway the prediction)

Next steps

I have successfully made machine learning models that help predict 5 emotions. Ican improve my project by:

- Training using deep learning models (CNN/RNN)
- Looking at more emotions than 5. Possibly, every emotion from plutchik's emotion wheel
- Making a text generator using Neural networks that will rephrase the sentence