NATURAL LANGUAGE PROCESSING

SENTIMENT ANALYSIS OF TWITTER DATA

TEAM NUMBER 34

GANGULA RAMA ROHIT REDDY : 201525083, KOLISETTI VAMSI KRISHNA : 201501232

MOURYA KUMAR REDDY : 201501240

INTRODUCTION:

Sentiment Analysis is the process of computationally determining whether a given text is positive or negative. It's also known as opinion mining, deriving the opinion or attitude of a speaker.

Generally speaking, sentiment analysis aims to determine the attitude of a speaker, writer, or other subject with respect to some topic or the overall contextual polarity or emotional reaction to a document, interaction, or event. The attitude may be a judgment or evaluation or the emotional state of the author.

AIM OF THE PROJECT:

The Aim of this project is to build an algorithm that can accurately classify Twitter messages as positive or negative.

Our hypothesis is that we can obtain high accuracy on classifying sentiment in Twitter messages using machine learning techniques along with some extracted features.

DATASETS:

We used Sentiment140 Dataset which consists of 1600000 sentences annotated as positive, negative.

Sentiment140 Dataset is available at http://help.sentiment140.com/for-students/ .

PROCEDURE:

Step 1: Preprocessing (Get raw data in suitable format)

1. Case Folding of the Data
2. Common Abbreviations and Acronyms expanded.
3. Hash-Tag removal.
4. Removal of hyperlinks

Step 2: Vectorization (TF-IDF Model , One-hot encoding)

1. We used two models One-hot encoding for neural network classifiers and TfidfVectorizer for others .
2. Train these models using only the sentences (after preprocessing) from the corpus.
3. This generates vectors for all the words in the corpus.

Step 3: Language Models

1. For Unigram, take one vector for one word.
2. For Bigram, take one vector for two words.

Step 4: Feature Extraction From Tweet:

The following features were extracted from the tweet text.

1. Presence of swear words
2. Length of the tweet
3. Number of words in tweet
4. Number of positive emotion words
5. Number of negative emotion words
6. Number of strong subjective adjectives
7. Number of strong subjective nouns
8. Number of weak subjective adjectives
9. Number of weak subjective nouns
10. Number ‘wh’ words
11. Number of superlative words
12. Number of quantification words and modal words
13. Number of Interjections

Step 5: Training For Machine Learning Scores

Training data used is:

1)Vectorized sentences 2) Vectorized sentences + feature vector of the sentence

1. Use training data to make the models for:
    a. Support Vector Machines - Scikit Learn Python
    b. Multi Layer Perceptron Neural Network - Scikit Learn Python
    c. Naive Bayes Classifier - Scikit Learn Python

d. Decision Tree Classifier - Scikit Learn Python
e. Random Forest Classifier - Scikit Learn Python
f. Logistic Regression Classifier - Scikit Learn Python
g. Convolutional neural network - Keras module Python
h. Long Short Term Memory - Keras module Python
i. CNN with LSTM - Keras module Python

2. Classify the test data using the models.
3. Compare the output results with the actual classes to get the accuracy.
4. Divide data into 4:1 ratio and apply 5-fold Cross-validation to get accuracies.

Note : Based on the results, Hybrid of Unigram , Bigram and Extracted features with Convolutional neural network Classifier gave the best results.

Step 6: Emoticon Scoring

1. Search for Emoticons in the given text using RegEx or find.
2. Use this emoticon score in the model.

RESULTS :

Accuracy

	Unigrams	Bigrams	Unigrams + Bigrams	Unigrams + Bigrams + Features
Support Vector Machines	70.8%	71.7%	69.8%	72.7%
Naive Bayes Classifier	70.7%	70.7%	71%	72.1%

Multi Layer Perceptron Neural Network	69.3%	70.3%	70.8%	72.2%
Decision Tree Classifier	61.7%	60.7%	63.6%	63%
Random Forest Classifier	66.5%	67.5%	66.7%	68.9%
Logistic Regression Classifier	70.5%	70.6%	70.6%	72.7%
Convolutional neural network	**73.9%**	**73.5%**	**73%**	**74.75%**
Long Short Term Memory	73.5%	69%	70%	67.6%
CNN with LSTM	69.6%	69%	69.3%	70.1%

Precision

	Unigrams	Bigrams	Unigrams + Bigrams	Unigrams + Bigrams + Features
Support Vector Machines	71%	72%	70%	73%

Naive Bayes Classifier	71%	71%	71%	72%
Multi Layer Perceptron Neural Network	69%	70%	71%	73%
Decision Tree Classifier	62%	61%	64%	63%
Random Forest Classifier	67%	68%	67%	69%
Logistic Regression Classifier	69%	71%	71%	73%
Convolutional neural network	74%	73%	73%	75%
Long Short Term Memory	74%	69%	70%	69%
CNN with LSTM	70%	69%	69%	71%

Recall

	Unigrams	Bigrams	Unigrams + Bigrams	Unigrams + Bigrams + Features
Support Vector Machines	71%	72%	70%	73%
Naive Bayes Classifier	71%	71%	71%	72%
Multi Layer Perceptron Neural Network	69%	70%	71%	73%
Decision Tree Classifier	62%	61%	64%	63%
Random Forest Classifier	67%	68%	67%	69%
Logistic Regression Classifier	69%	71%	71%	72%
Convolutional neural network	74%	73%	73%	75%
Long Short Term Memory	73%	69%	70%	68%
CNN with LSTM	70%	69%	69%	70%

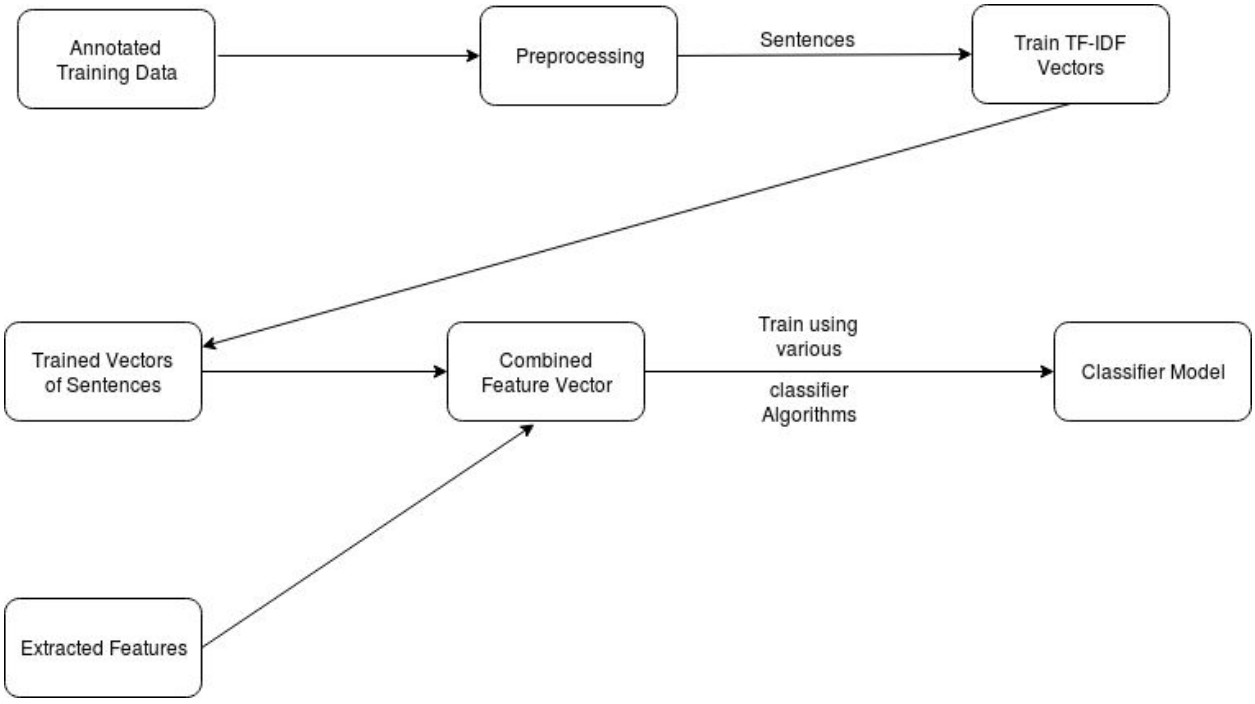
F1 score

	Unigrams	Bigrams	Unigrams + Bigrams	Unigrams + Bigrams + Features
Support Vector Machines	71%	72%	70%	73%
Naive Bayes Classifier	71%	71%	71%	72%
Multi Layer Perceptron Neural Network	69%	70%	71%	73%
Decision Tree Classifier	62%	61%	64%	63%
Random Forest Classifier	66%	67%	67%	69%
Logistic Regression Classifier	69%	71%	71%	72%
Convolutional neural network	74%	73%	73%	75%

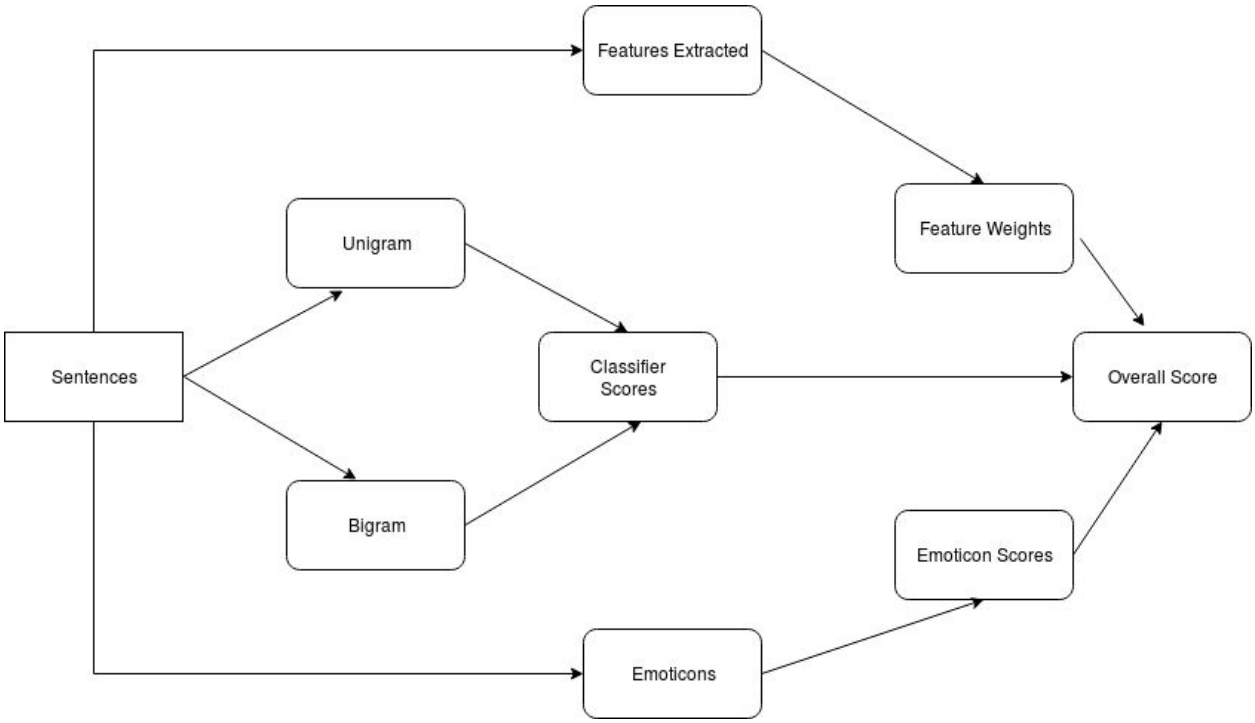
Long Short Term Memory	73%	69%	69%	67%
CNN with LSTM	69%	69%	69%	70%

Work Flow Diagram:

Training Classifier and Vector Model



The Overall Process:



CHALLENGES:

➔ **Randomness in Data:**

Twitter is written by Users, hence it is not very formal.

➔ **Emoticons:**

Different of types of emoticons with new ones coming very frequently.

➔ **Abbreviations:**

Users use a lot of abbreviation slangs like AFAIK, GN, etc. Capturing all of them is difficult.

➔ **Grapheme Stretching:**

Emotions expressed through stretching of normal words. Like, Please -> Pleaaaaaseeeeee

➔ **Reversing Words:**

Some words completely reverse sentiment of another word.

E.g: not good == opposite(good)

➔ **Technical Challenges:**

Classifiers take a lot of time to train, hence silly mistakes cost a lot of time.

➔ **Other Challenges:**

A small emoticon like :P at the end or a word can change the entire polarity of the tweet.

E.g: This is too good :P or “This is awesome super fantastic mind blowing NOOTTT!!!”

FUTURE IMPROVEMENTS:

➢ Handle Grapheme Stretching

➢ Handle authenticity of Data and Users

➢ Handle Sarcasm and Humor

REFERENCES:

1. https://dl.acm.org/citation.cfm?id=1183626
2. http://scikit-learn.org/stable/documentation.html

GitHub Link:

https://github.com/Rama-007/SENTIMENT_ANALYSIS_OF_TWITTER_DATA