CS-521: Introduction to Python

Final Term Project

Professor: Alan Burstein

Topic: Sarcasm Detection

Group Members:-

1) **Dhairya Timbadia**
2) **Dharmik Timbadia**
3) **Mohamed Faadil Shaikh**

Abstract

Sarcasm is a sophisticated form of irony widely used in social networks and micro-blogging websites. It is usually used to convey implicit information within the message a person transmits. Sarcasm might be used for different purposes, such as criticism or mockery. However, it is hard even for humans to recognize. Therefore, recognizing sarcastic statements can be very useful to improve automatic sentiment analysis of data collected from micro-blogging websites or social networks. Sentiment Analysis refers to the identification and aggregation of attitudes and opinions expressed by Internet users toward a specific topic. In this project, we propose a pattern-based approach to detect sarcasm on Twitter.

Understanding Sarcasm Detection

To understand sarcasm, there are different approaches to identify the sarcasm in the sentences. There are three popular ways to detect sarcastic sentences which are lexical, hyperbole, pragmatic. We have used multiple algorithms which can determine whether a statement or some part of text is sarcastic or not. From the dataset used, the model is being trained so that it can check for any given sentence to detect whether it is sarcastic or not. This will help in obtaining appropriate analysis about the sentiment hidden in the sarcastic statement.

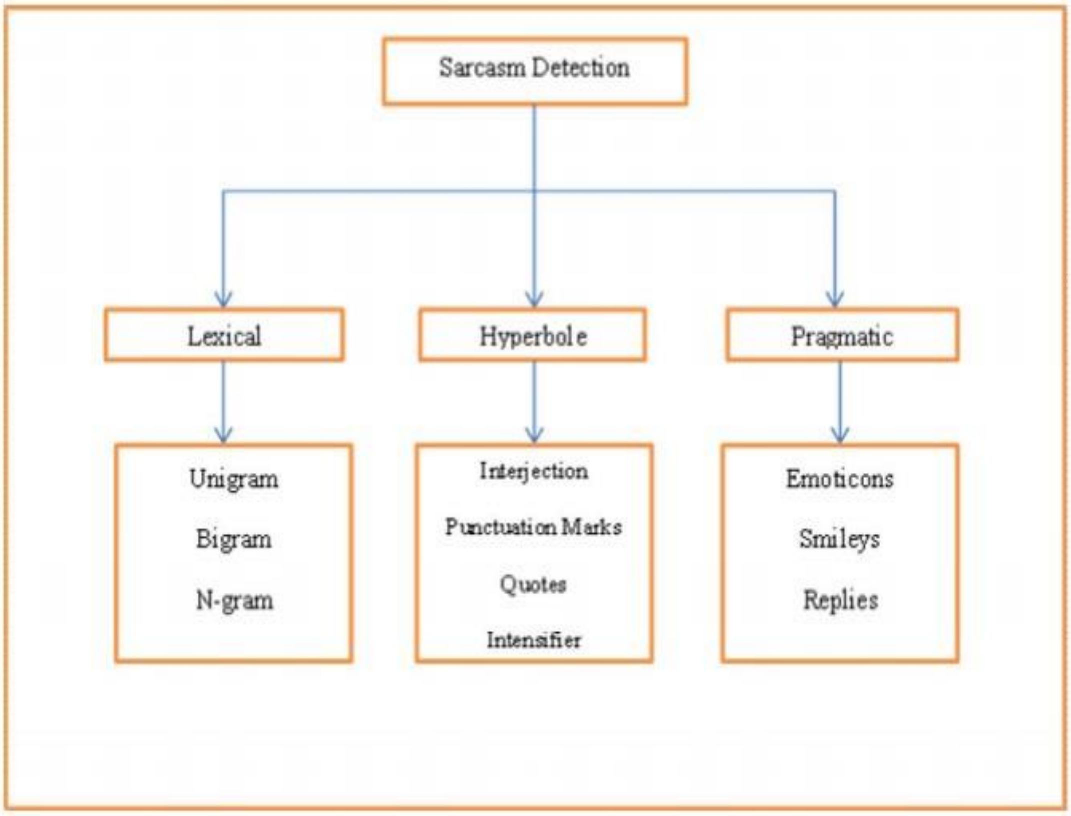

Figure [1] Approaches and Components used in the Sarcasm Detection

Dataset details:

News Headlines Dataset For Sarcasm Detection | Kaggle Kaggle

News Headlines Dataset For Sarcasm Detection | Kaggle

Flow:

Download the dataset using Kaggle API
Load the dataset JSON file

Pre-processing the data:

- Lowercase
- Remove the Hashtags from the text
- Remove the links from the text
- Remove the Special characters from the text
- Substitute the multiple spaces with single spaces
- Remove all the single characters in the text
- Remove the twitter handlers

Components used in Natural Language Processing: -

The tweets are differentiated among sarcastic and non-sarcastic based on the detection engine. In this project we have sarcastic headlines and non – sarcastic headlines shown in graphs. We have also processed and shown words in the articles of the dataset which are sarcastic. We used different methods to remove all the unnecessary components in a sentence. The following are natural language components we used in our project.

Removal of Stop Words: -

Stop words are words that are so common to languages that removing them doesn't affect the overall message enough to lose meaning. The stop words are removed using python programming. The stop words corpus distributed with NLTK have been used. The stop words corpus consists of 2400 stop words for 11 different languages. Example: "This is a long bar sentence". After stop words removal we get `long' `bar' `sentence'.

Tokenization: -

Tokenization is process of breaking texts into words, phrases and symbols called tokens. The tokens generated helps in parsing and performing other mining tasks. Tokenization is done using Text Blob package installed in NLTK. Example: "Beautiful is better than ugly". After tokenization using Text Blob, we get every word separated such as `Beautiful', `is', `better', `than', `ugly'.

Part of Speech Tagging: -

Part of Speech Tagging divides sentences into words and assign part of speech for each word. In this project Text Blob is used for Part of Speech Tagging. Example: "Sarcasm detection using twitter data". Part of Speech tagging for the example is Sarcasm-NNP, detection-NN, using-VBG, twitter-NN, data-NNS. Here NNP stands for proper noun singular, NN stands for noun singular, VBG stands for verb present participle and NNS stands for noun plural.

Each record consists of three attributes:

- is_sarcastic: 1 if the record is sarcastic otherwise 0

- headline: the headline of the news article
- article_link: link to the original news article. Useful in collecting supplementary data

Splitting it into training and testing: - ratio 85 – 15%
When splitting into training and testing, we saw that while training data the accuracy of all the algorithm is 100% or close to 100% but while testing the data, we got variable accuracy.

Applying Count Vectorizer

Applying Algorithms:

SVM: - The training accuracy of the SVM algorithm is 98.06% while the testing accuracy which we actually consider while running an algorithm for SVM is 70.85%.

Adaboost: - Adaboost is one of the most popular algorithms used in python as well as natural language processing. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The training accuracy of adaboost algorithm is 66.53 and during testing, the accuracy is very close which is 65.57%.

Decision Tree: - A tree structure is constructed that breaks the dataset down into smaller subsets eventually resulting in a prediction. The training accuracy is 100% while the testing accuracy of this algorithm is 71.3%.

Logistic Regression: - Logistic regression is a fundamental classification technique. It belongs to the group of linear classifiers and is somewhat similar Logistic regression is fast and relatively uncomplicated, and it’s convenient for you to interpret the results. The training accuracy is 99.90% while testing accuracy of logistic regression is 80.31%.

Conclusion: -

Therefore, we calculated the accuracy is of each and every algorithm is calculated while taking every dataset in the database into consideration and showing different visualisations of sarcasm detection. We also used and projected various other visualisation techniques for better understanding in our project.