Article Credibility Checker

This repository contains the code required to build the Article Credibility Checker algorithm.

The internet is filled with numerous articles from all over researching and discussing the same topic. Oftentimes many of these articles are unable to verify whether the information they provided is credible. In the most recent days, many social media sites have begun to implement fact checking sources within posts that are providing potential facts on a certain subject. With that, in today's world it is hard to separate 'Fake News' from credible sources. Fact-checking sites do the job of manually verifying with the eyewitnesses' accounts. However, through the increase in Fake News sites there has been an increase in demand for automatic fact checking algorithms.

This program will check the credibility of the site by categorizing words into separate groups based on positive and negative words using the K-means clustering algorithm. Currently the code works by receiving a csv file that contains text. It then creates a bag of words, does tf-idf, and then uses that to do k-means. The code then takes that results of k means and put the words/sentence into its corresponding cluster. The program will search through and look for key uncredible words. The uncredible words are stored in a list. If certain words are found in the list, the program will increase the credibility score. Currently the score is 1-3 is credible, 4-6 moderately credible, and 7-10 is uncredible. The more uncrediable words are found, the higher the credibility score will get. Finally, credResults is the output file that will display the results mentioned above.

In addition, for the bag of words list, it includes Stop words (ex. "I", "We") because these words are considered to make a site uncredible. Traditionally Stop words are not included in clusters for but for the program to work, some Stop words are used.

Packages specific to this project that need to be installed

1. Natural Language ToolKit (NLTK)
2. SciKit Learn
3. Seaborn
4. pip install spacy
5. python -m spacy download en_core_web_sm
6. import nltk
7. from nltk.corpus import stopwords
8. import string
9. import pandas as pd

10. from sklearn.cluster import KMeans
11. from sklearn.feature_extraction.text import TfidfVectorizer
12. from spacy.lang.en import English
13. import spacy
14. import csv

Files in this repository

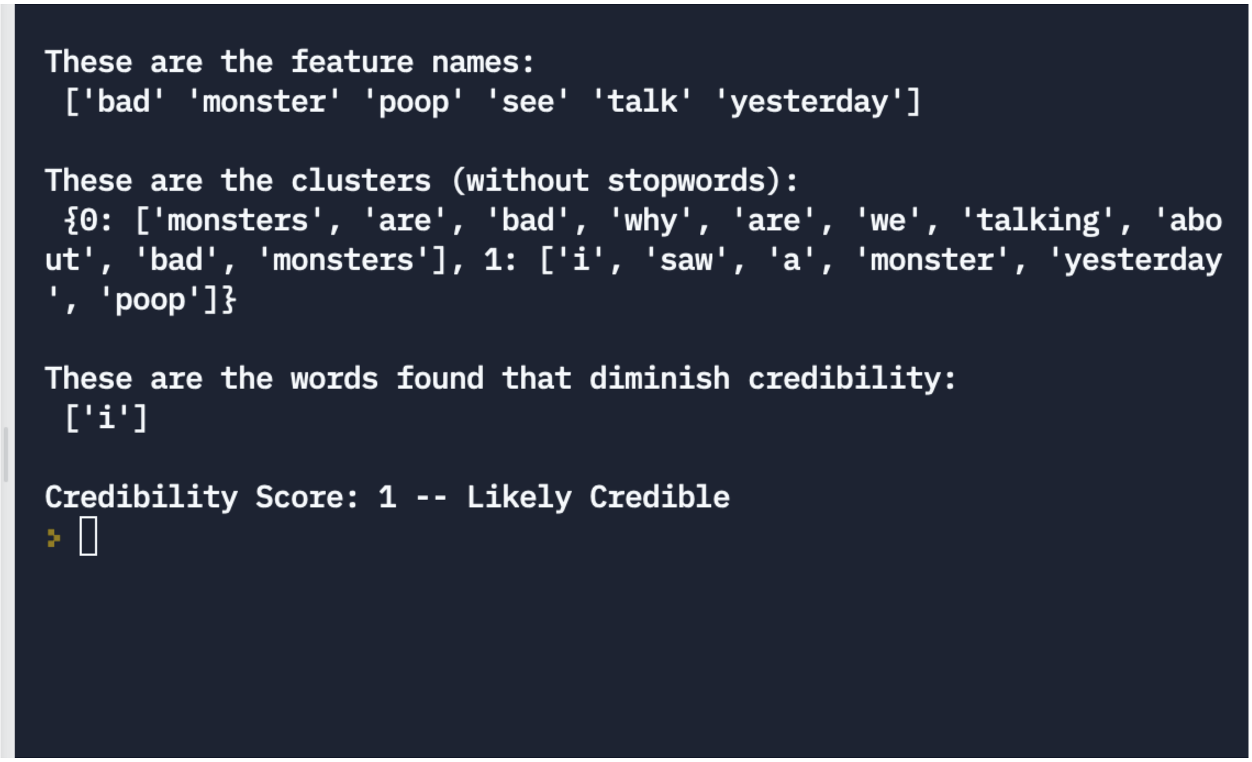
1. Requirements.txt
2. credResults.csv
3. File.csv
4. credibilityTest.py

Installation Process

In order for the project to run effectively the users must install the required packages stated above and include .csv files to be implemented throughout the code.

Example of Results

Below is an example of the result for the code. On this screen, it lists the feature names (words from the sentence used in TFIDF), the cluster, and words found that can diminish credibility. Finally the credibility score is displayed at the end of the code stating, “Likely Credible”.



```
These are the feature names:
 ['bad' 'monster' 'poop' 'see' 'talk' 'yesterday']

These are the clusters (without stopwords):
 {0: ['monsters', 'are', 'bad', 'why', 'are', 'we', 'talking', 'abo
ut', 'bad', 'monsters'], 1: ['i', 'saw', 'a', 'monster', 'yesterday
', 'poop']}

These are the words found that diminish credibility:
 ['i']

Credibility Score: 1 -- Likely Credible
›
```

Future Developments

Our next step in the development process is to combine our code with remaining the groups to see how all of our code works together to execute the initial goal of the project for the SpelCheck webpage. In addition, the uncredible words, we would like to expand to include more words in this list. Finally, be able to implement the elbow method in this program. The elbow method will determine the amount of clusters for k means, instead of the team hard coding the number of clusters.

Code Contributors

Kennedy Butts, Chyna Hester, Coriyanna Osbonre-Willis, Mumbi Whidby, and Hawa Wague