Fake News App

Description:
A web app that will allow a user to input an article link then return the sentiment score, evidence for or against the main claim made, "truthfulness" score of site, author, article.

Hypothesis:
Facts have a neutral sentiment score and the facts are presented in a straightforward way without bias. Therefore if the article is not using emotions to manipulate the reader and the claim is easily proveable without bias we should be able to see that factual reporting has a neutral sentiment score and evidence to back up the main claim made.

Outline:
When address is inputted the app will scrape the site for the article text. From the article text we will do two things.

1. We will find the major claim being made by the article by finding the "anchor sentence". Using this anchor sentence we will google the claim and find results from trusted websites (e.g. - .gov, .edu, scholarly article databases). We will return all trusted evidence if there is any.
2. We will score the sentiment of the article, Factual reporting has neutral sentiment score because facts have no emotional score.

The app will return the sentiment score, links for evidence to back up claim, stripped article text that will highlight words that have a sentiment score above or below the standard deviation, average "truthfulness" score of of site, author, and if this article/claim has been searched before.

Road Map

- Front End
    - Web Page:
        - User input box
        - User submit button
        - Display:
            - Sentiment score
            - Evidence links
                - Prove or disprove
            - Stripped article text
                - Highlight words that triggered score
            - "Truthfulness" score of website
            - "Truthfulness" of author
    - Tech Stack:
        - HTML/CSS

        - React

- Back End
    - Web Page:
        - Python (Django or Flask) framework
        - If article has been searched before display results
        - Store in database:
            - Article link
            - Source website
            - Stripped text
            - “Anchor” sentence / main claim made
            - Sentiment score
            - Running score of “truthfulness” of site
            - Running score of “truthfulness” of author
    - Tech Stack:
        - Django/Flask
        - MySql

- Detection
    - Python Tech Stack:
        - Web scraper library
            - Create algorithm to grab text between HTML tags
        - Sentiment library
        - Text mining library
        - Databases to access:
            - Scholarly article DB
            - Find government statistics
            - Find educational statistics
    - Frame work:
        - Input article link
        - Web scraper goes to link and strips text
        - Algorithm grabs text between HTML tags
        - Run sentiment analysis
            - Store score
        - Identify “anchor” sentence (main claim made)
            - Possibly create algorithm that finds sentences that are referenced the most
            - Store sentence
        - Search main claim made from trusted websites database
            - .gov, .edu, scholarly article database
        - Algorithm to score “truthfulness” of:
            - Site
            - Author

- Miscellaneous
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -
    -

Sprints

Sprint 1 [March 4 - March 8]:

Sprint 2 [April 8 - April 12]:

Sprint 3 [May 6 - Friday, May 10]:

Road Map: