How to Use Fake News Detection Extension:

1. Load Extension
   a. For Chrome:
      i. Open Chrome browser
      ii. Type chrome://extensions/ in the address bar and press Enter
      iii. Toggle ON the "Developer mode" switch in the top-right corner
      iv. Click the "Load unpacked" button
      v. Navigate to and select “Fake News Detection Extension" folder
      vi. Click "Select Folder"
   b. For Edge:
      i. Open Edge browser
      ii. Type edge://extensions/ in the address bar and press Enter
      iii. Toggle ON "Developer mode" on the left sidebar
      iv. Click "Load unpacked"
      v. Navigate to and select “Fake News Detection Extension" folder
      vi. Click "Select Folder”
2. Navigate to any news article (Reuters, AP News, etc.)
   a. Example: https://www.reuters.com/article/us-usa-fiscal/as-u-s-budget-fight-looms-republicans-flip-their-fiscal-script-idUSKBN1EP0LK/
3. Click browser extension icon
4. View prediction that includes confidence score.

Links to the external files (data, libraries, modules etc.,) used for the project:

- Used Fake News Detection Dataset from Kaggle to train model: https://www.kaggle.com/datasets/emineyetm/fake-news-detection-datasets/data
- Used NLTK’s implementation of PorterStemmer for stemming preprocessing step (PorterStemmer class in popup.js) : https://github.com/nltk/nltk/blob/develop/nltk/stem/porter.py
- Python modules

numpy	Used for numerical operations like np.where for identifying non-zero features, np.exp for sigmoid calculation in manual prediction, and handling arrays for TF-IDF vectors and model coefficients.	NumPy Documentation
re	Used in the stemming function (re.sub) to remove non-alphabetic characters from text during preprocessing.	Python re Documentation
pandas	Used for loading datasets, checking for null values, and applying the stemming function across the text column.	Pandas Documentation

nltk	Used for downloading stopwords and for PorterStemmer in the text preprocessing stemming function to clean and normalize the news article text.	NLTK Website
sklearn (Scikit-learn)	Used for TfidfVectorizer to convert text data into numerical features (TF-IDF), train_test_split to divide data into training and testing sets, LogisticRegression for building and training the classification model, and accuracy_score to evaluate model performance.	Scikit-learn Documentation
joblib	Used to save (joblib.dump) and load (joblib.load) the trained LogisticRegression model and TfidfVectorizer for later use	Joblib Documentation
json	Used to export model parameters (model_params.json) to be used in browser extension.	Python json Documentation