DOCUMENTATION

**FAKE NEWS DETECTION USING PYTHON**

**Project Objectives and Scope**

*Goals*

* The model accuracy of detecting news articles should be at least 85%
* The model should be an interpretable model that has transparency with its predictions, and they can be explained easily.
* Develop a user-friendly interface for easy interactions with fake news detection system.

*Scope*

* News articles will be related to politics, health and medical and finance.
* Data sources will be from reputable news sources and some potential misinformation sources.
* User interface will be basic and easily navigable.
* System complexity will be minimal as we will not use complex models to do predictions.
* We will focus on the news text feature to do feature engineering.

**Data Collection**

I downloaded the dataset from this [link](https://www.kaggle.com/datasets/saurabhshahane/fake-news-classification).

(WELFake) is a dataset of 72,134 news articles with 35,028 real and 37,106 fake news. For this, authors merged four popular news datasets (i.e. Kaggle, McIntire, Reuters, BuzzFeed Political) to prevent over-fitting of classifiers and to provide more text data for better ML training.

Dataset contains four columns: Serial number (starting from 0); Title (about the text news heading); Text (about the news content); and Label (0 = fake and 1 = real).

The dataset contained lots of rows which were not in the right format. So, I deleted those rows and only kept the rows which had the right format. For simplicity, I only took 2 columns from the dataset, text and label. Then I converted the labels into ‘fake’ and ‘real’ depending on the label (0 or 1) used.

I cleaned the dataset using excel and converted the dataset into a .xlsx file. After the cleaning of the dataset. I got 34,977 real and 36,248 fake news.

Due to the size of the dataset, I have separated the dataset into 2. One to train model using small dataset and other to train with larger dataset.

For information regarding manually classifying fake and real news, look at the following-

Check out one blog related to [Fake News and Misinformation](https://libguides.library.umaine.edu/fakenews/identification).

Check out another blog on [What makes a news story fake?](https://guides.library.stonybrook.edu/fakenews)

In future if I want to expand the dataset or want my model to be trained further, I can use the following-  
Check out real and fake news dataset from [here](https://github.com/KaiDMML/FakeNewsNet/).

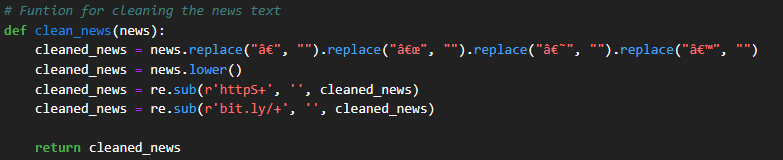
Sources to get fake and real news through web scraping are Snopes, FactCheck.org, PolitiFact.

**Data Cleaning**

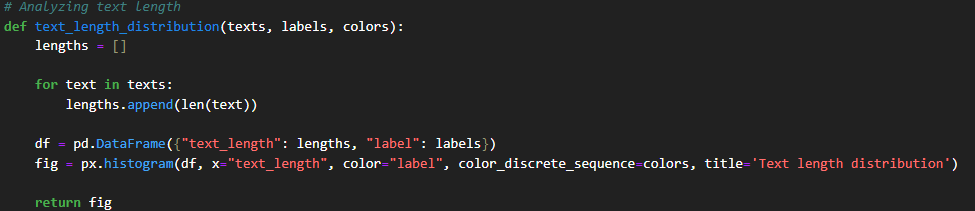
First, I will do some simple cleaning on the text. Some text data looks like this and they contain links and odd characters.



* Replace â€, â€œ, â€˜, â€™ with BLANK.
* Transform the text to lowercase.
* Remove httpS+, bit.ly/S+ from text.



**Exploratory Data Analysis (EDA)**

**Text Length Analysis**A white rectangular object with text

Description automatically generated with medium confidence

**Class Distribution**A screen shot of a computer code

Description automatically generatedA red and green circle with black text

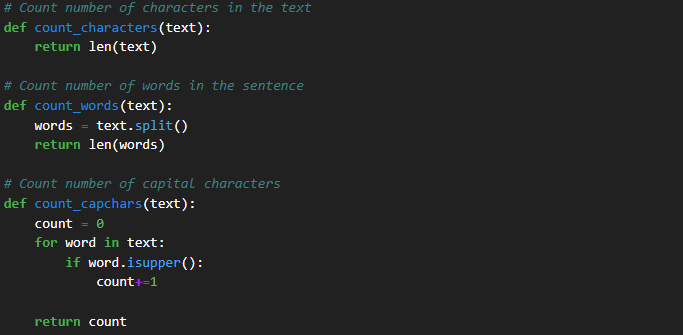
Description automatically generated

**Most Common Words**A graph of a bar and a bar of a graph

Description automatically generated with medium confidence

**Feature Engineering**

* Generated the following features from existing text:
  + Number of characters
  + Number of words
  + Number of capital characters
  + Number of capital words
  + Number of punctuations
  + Number of words inside quotes
  + Number of sentences
  + Number of unique words
  + Number of hash tags
  + Number of mentions
  + Number of stopwords
  + Average sentence length: Number of words / Number of sentences
  + Unique words vs number of words: Number of unique words / Number of words
  + Stopwords vs number of words: Number of stopwords / Number of words

A screen shot of a computer code

Description automatically generatedA computer screen shot of a black screen

Description automatically generatedA computer screen shot of a computer code

Description automatically generatedA screen shot of a computer program

Description automatically generatedA black screen with red text

Description automatically generated

**Data Preprocessing**

* Remove users and hashtags.
* Remove punctuation.
* Lowercase the text and remove stopwords.
* Apply stemming on the text.
* Vectorize the texts using TF-IDF vectorizer.

**Model Selection**

* Start with traditional models like logistic regression, Naive Bayes.
* Explore more advanced models like decision trees, random forests, support vector machines, and deep learning models.
* Fine-tune hyperparameters using cross-validation.

**Model Training**

* Train the selected model on the preprocessed and engineered dataset.
* Monitor training performance and adjust parameters as needed.
* Validate the model on a separate validation set.

**Model Evaluation**

* Evaluate the model's performance using relevant metrics:
  + Accuracy, precision, recall, F1-score.
  + Confusion matrix analysis.
  + Consider using techniques like cross-validation for a robust evaluation.

**Model Interpretability and Explainability**

* Ensure that the chosen model provides interpretability.
* Implement techniques for explaining model predictions, such as SHAP values or LIME.

**Deployment Preparation**

* Develop a user-friendly interface for interacting with the model.
* Ensure scalability and efficiency of the model for real-time or batch processing.
* Consider security measures to protect the model and data.

**Model Deployment**

* Deploy the trained model to a production environment.
* Integrate the model with the user interface for easy access.
* Implement monitoring for model performance in real-time.

**END OF DOCUMENTATION**