Fake News Classifier:

By: Pinky Chauhan (University of Illinois at Urbana Champaign)

**Overview**:

This objective of this project is to build a classifier system based on machine learning that is able to identify fake news from real/reliable news given a news title and/or news text content as the input. Such a tool can be integrated with social media platforms to flag potentially fake articles or filter those out.

This is essentially a data categorization problem where I have trained several classifier models on the following dataset from Kaggle:

<https://www.kaggle.com/c/fake-news/data>

After evaluation based on various performance metrics, one of the models (in this case, Linear SVC over unigram bag-of-words/TF-IDF representation) is integrated in the final tester notebook to test with news data.

The classifier takes a news article (title and text) as input and provides a prediction for the news article as either of the 2 categories:

* Fake News
* Reliable News

**Software Implementation Details**:

• Data Analysis:

* Comprises of checking several attributes to evaluate their contribution towards classification. For such a classifier, the text and title of the news make obvious choices as features.
* I also analyzed authors’ distribution using pandas and polarity/sentiment differences using nltk vader sentiment intensity analyzer library on the dataset.

• Preprocessing of data:

* Comprises of handling missing values by removing any rows with no text and title, preprocess data to remove any punctuations, stop words removal, tokenization and lemmatization using nltk libraries

• Feature selection:

* Concatenated and cleansed/preprocessed news title and text comprises the feature to train the model

• Vectorization

* Different vector forms have been used using nltk vectorization libraries:
  + Term frequency (TF) based vector over unigrams bag of words representation
  + Term frequency/inverse document frequency (TF-IDF) based vector over unigrams
  + Term frequency (TF) based vector over unigrams and bigrams
  + Term frequency/inverse document frequency (TF-IDF) based vector over unigrams and bigrams
  + Term frequency/inverse document frequency (TF-IDF) based vector over unigrams, bigrams and trigrams
* Training/hyperparameter tuning/validation using classification models:
* Models used (sklearn libraries):
  + Naïve bayes (With/without smoothing, TF vs TF-IDF vectors, Unigram/N-gram)
  + Logistic Regression (TF-IDF vectors using Unigrams/N-grams)
  + SVM using Linear SVC (TF-IDF vectors using Unigrams/N-grams, Regularization)
  + SGDC classifier (TF-IDF vectors using Unigrams/N-grams)
  + Decision Tree (TF-IDF vectors using Unigrams/N-grams)
* Performance evaluation:
  + Compute and analyze metrics using sklearn metrics libraries
    - Precision (macro/micro), recall (macro/micro), F1 (macro/micro)
    - Classification Accuracy
    - Confusion matrix to see distribution of true/false positives/negatives
  + Select the best performing model based on evaluation results (SVM using Linear SVC using TF-IDF vector over unigrams)
* Save/export trained model:
  + Using pipeline to specify all steps (vectorizer/classifier), fit training data and exporting model using joblib library
* Kaggle submission:
  + Predicted results for data in test.csv and submitted notebook/results to Kaggle
  + Accuracy: 96%
* Create script (Jupyter notebook) that will take news text as input and generate classification as reliable news or fake news.

**Installation/Execution Details**:

Code is written using Jupyter notebook and python 3

Code structure:

* data: This directory contains the dataset from Kaggle (). There are 3 files:
  + train.csv: To use for analysis, training, validation
  + test.csv: Test dataset for submission of results to Kaggle competition
  + submit.csv: File containing results/predictions for data in test.csv
* notebooks: This directory contains 2 notebooks:
  + FakeNewsClassifier\_Training.ipynb: Jupyter notebook containing code/results for data analysis, cleanup, features set up, vectorization, training using various classifier algorithms, tuning and performance evaluation/comparison, model pipeline creation/export, prediction of results for test.csv for Kaggle submission
  + Tester.ipynb: This notebook loads the pretrained/exported model and predicts the category for a given news article.
* model: This directory contains the pretrained model exported by FakeNewsClassifier\_Training.ipynb notebook and loaded by Tester.ipynb

Code Setup:

* Install python 3 and Jupyter notebook
* Install the following python/machine learning libraries:
  + re: For regular expression matching
  + pandas: For Data analysis/representation as Dataframes
  + nltk: Natural language toolkit
  + sklearn: For model selection, training, evaluation, export using pipeline
  + matplotlib: For visualization
  + joblib: For model export and load
* Checkout the project from main branch
* Launch Jupyter notebook and navigate to the directory where project is checked out
* Tester.ipynb located in notebooks folder should be used for testing
* FakeNewsClassification\_Training.ipynb can also be executed to see all stages discussed in implementation details live