Detecting Fake News using Supervised Machine Learning Algorithms

1. Task Description

The aim of this project is to develop a ***Fake News Detection Application*** using supervised machine learning algorithms, which can predict whether a piece of news is real or fake. More precisely, the task of the project is to train a classifier on a labeled training dataset (fake, real), which can be then used to classify new unseen pieces of news as real or fake, based on the features of the news article.

2. Dataset Description

The dataset we'll use for this project- we'll call it **news.csv**. This dataset has a shape of **7796×4**. The first column identifies the news, the second and third are the title and text, and the fourth column has labels denoting whether the news is **REAL** or **FAKE**. The dataset takes up 29.2MB of space and you can download it here.

3. Main steps involved in the project

The project consists of the following main steps:

A. **Text Preprocessing** is the process of cleaning and normalizing text data before feeding it into a machine learning algorithm. The main steps involved in text preprocessing are:
   - Tokenization,
   - Stemming or Lemmatization,
   - Removing special characters, URLs, emails, hashtags…
   - Removing stop words that are not relevant to the task at hand, such as words that occur frequently in a text but don't carry much meaning.

⇨ **To perform Text Preprocessing, you can use regular expressions, *NLTK* or *spaCy* libraries (As done in assignment 1).**

B. **Features Extraction,** also known as **Text Representation** in NLP, is the process of transforming raw data into a set of features that can be used to train a machine learning model. Features extraction is a key step in many machine learning applications, such as text classification, image classification, object detection, and so on.
For NLP applications, the quality of document analysis depends on the amount of information that can be extracted from it. For this purpose, several methods have been proposed in the literature, including:

- **Bag-of-words representation,** shortened as BOW, counts the number of occurrences, or frequency, of each word in the given corpus of documents. It involves a vocabulary of known words and a measure of the presence of these words.
The widely used BOW method is ***Term frequency–inverse document frequency (TF-IDF)*** model, which measures the importance of a word in a document by determining its frequency in the document, and its inverse document frequency. In other words, it yields a value that shows how important a given word is by not only looking at term

frequency, but also analyzing how many times the word appears across all the documents of the corpus.

Given a corpus of N documents, the *TF-IDF* of a word w in a document d, is defined as follows:

$$TF - IDF(w^d) = x_w^d . \log\left(\frac{N}{N_w}\right)$$

Where N is the number of documents in the corpus, $x x_w^d$ is the number of occurrences (or frequency) of the word w in the document d, and N_w the number of documents that contain the word w. This measure is an approximation of the representativeness of a word in a document. Thus, a high *TF-IDF* value means that the word w important in the document d and appears little in the other documents of the collection.

- **Topic Modeling methods** aim to extract information from texts that describe the main "topics" occur across all the documents. The most common methods for topic modeling are ***Latent Dirichlet Allocation (LDA)*** and **Latent Semantic Analysis (LSA)**.

- **Word Embedding methods** aim is to learn high-quality word vectors from huge datasets with billions of words, and with millions of words in the vocabulary, which reflect similarities and dissimilarities between them rather than treating individual words as independent symbols. These vectors can then be used to train machine learning models. Common word embedding methods include ***word2vec*** and ***GloVe***.

- **Sentence Embedding methods** aim to encode sentences into dense vectors, represented in a low dimensional vector space, that accurately capture the semantic and syntactic relationships be- tween these sentences constituents. Common sentence embedding methods include ***BERT*** and ***ELMo***.

⇨ **In this first version of project, we focus on TF-IDF model as feature extraction method. To do this, check the *TfidfVectorizer module* from *scikit-learn library* that converts a collection of raw documents into a matrix of TF-IDF features.**

C. Text Classifier Selection: several methods for text classification exist, including:

- **Generative machine learning models** focus on the distribution of a dataset to return a probability for a given example using Joint Probability. In this work, we use the ***Naïve Bayes (NB) Classifier***.

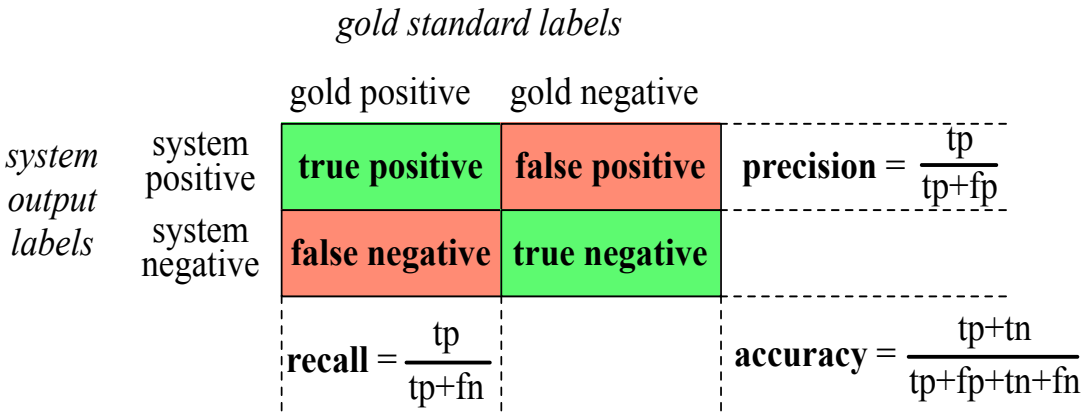
- **Discriminative machine learning models** calculate the probability of a latent trait using conditional probability. Common supervised ML models include:
  - ***Logistic Regression (LR)***
  - ***Support Vector Machines) (SVM)***
  - ***K-nearest-neighbors (K-NN)***

⇨ **Perform fake news classification using the above models (NB, LR, SVM, and K-NN). You do not need to implement these models from scratch, it is recommended to use *scikit-learn library* (**https://scikit-learn.org/stable/supervised_learning.html#supervised-learning**).**

D. Model Training, Testing, and Evaluation

- Split the dataset into training and testing sets. It is preferable to apply ***K-Fold cross-validation*** (*https://scikit-learn.org/stable/modules/cross_validation.html*).

- Evaluate your model using the **Accuracy** metric and the confusion matrix (**Recall**, **Precision**, and **F-1 measure**).



$$F\text{-}1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

⇨ **Compare the performance of the different models (NB, LR, SVM, and K-NN), and discuss the results.**

4. Useful Readings:

Support Vector Machine:

- https://bookdown.org/mpfoley1973/data-sci/maximal-margin-classifier.html
- https://www.projectpro.io/data-science-in-r-programming-tutorial/support-vector-machine-tutorial
- https://nlp.stanford.edu/IR-book/pdf/15svm.pdf
-

Machine Learning Methods for Text Classification Application:

- https://www.projectpro.io/article/machine-learning-nlp-text-classification-algorithms-and-models/523