Detecting Fake News with Supervised Learning

Benjamin R. Perucco

December 30, 2020

1 Definition

1.1 Project Overview

1.2 Problem Statement

1.3 Metrics

2 Analysis

2.1 Algorithms and Techniques

The news articles must be converted into a structured, mathematical representation in order to apply machine learning techniques. The following chapters disucss how we can convert text into numerical features.

2.1.1 n-gram Model

The n-gram defines usually a contiguous sequence of words with length n. For example, if $n = 1$, we speak of a unigram that contains only single word tokens. Or if $n = 2$, we denote this as a bigram that is build on two adjacent word tokens.

Consider the following text: "Sometimes we eat green apples, and sometimes, the apples we eat are red." Based on a unigram, we obtain a set of tokens: {'sometimes', 'we', 'eat', 'apples', 'green', 'and', 'the', 'are', 'red'}. We can derive a frequency array of tokens in the text: [2, 2, 2, 2, 1, 1, 1, 1, 1]. For the bigram, another set of tokens is obtained: {'sometimes we', 'we eat', 'eat green', 'green apples', 'apples and', 'and sometimes', 'sometimes the', 'the apples', 'apples we', 'eat are', 'are red'}. The corresponding frequency array of tokens in the text is: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1].

In order to build frequency arrays for a set of texts (or documents), we need to build a common vocabulary of which the n-gram model is underlying principle.

2.1.2 Vocabulary

Consider a corpus D which contains a set of documents $\{d_1, d_2, d_3, ..., d_n\}$. Then a vocubulary F is a set of tokens $\{f_1, f_2, f_3, ..., f_m\}$ extracted from the corpus D. Please remind yourself that a token is created on the n-gram model. Usually for a set of tokens, we consider the m mostly occuring tokens in a corpus D. In the following, the tokens are denoted as features as these build the features (or independent variables) of a machine learning model.

2.1.3 Definitions

Let $\sigma(f_j, d_i)$ the number of occurances of feature f_j in document d_i. Then we can build a feature matrix X, where

$$X = \begin{bmatrix} \sigma(f_1,d_1) & \sigma(f_2,d_1) & \sigma(f_3,d_1) & ... & \sigma(f_m,d_1) \\ \sigma(f_1,d_2) & \sigma(f_2,d_2) & \sigma(f_3,d_2) & ... & \sigma(f_m,d_2) \\ \sigma(f_1,d_3) & \sigma(f_2,d_3) & \sigma(f_3,d_3) & ... & \sigma(f_m,d_3) \\ ... & ... & ... & ... & ... \\ \sigma(f_1,d_n) & \sigma(f_2,d_n) & \sigma(f_3,d_n) & ... & \sigma(f_m,d_n) \end{bmatrix}. \tag{1}$$

For a specific element in row i and column j (representing the document d_i and feature f_j) in the matrix X, we abbreviate by using σ_{ij}.

2.1.4 Term Frequency Model

Matrix X could be already used for machine learning, but in the term frequency model, matrix X is normalized to matrix $\hat{X}$, where an element $\hat{\sigma}_{ij}$ is

$$\hat{\sigma}_{ij} = \frac{\sigma_{ij}}{\sum_{j=1}^{m} \sigma_{ij}}. \tag{2}$$

Or spoken in plain language: the number of occurances of a token f_j in a document d_i is divided by the total number of occurances of all tokens $\{f_1, f_2, f_3, ..., f_m\}$ in the same document d_i. So we have a representation where the importance of each feature can be compared to other features in each document.

2.1.5 Inverse Document Frequency Model

The inverse document frequency model is used to define the feature importance not just in a document d_i but compared within a corpus D. We introduce a matrix Y, where an element δ_{ij} is 1 if $\hat{\sigma}_{ij} > 0$. An inverse normalization step is performed on the matrix Y resulting in a vector $\hat{y}$, where an element $\hat{\delta}_j$ is

$$\hat{\delta}_j = 1 + \log\left[\frac{|D|}{\sum_{i=1}^{n} \delta_{ij}}\right]. \tag{3}$$

Or spoken in plain language: in a corpus D which comprises of a set of documents $\{d_1, d_2, d_3, ..., d_n\}$, we count in how many documents the feature f_j appears. This number is used to divide the number of documents $|D|$ in a corpus D. Consider two examples: if a feature f_j occurs in each document $\{d_1, d_2, d_3, ..., d_n\}$, we divide the number of documents $|D|$ in a corpus D by the same number. So expression 3 results in 1, weighting feature f_j as 1. On the other hand, if a feature f_j occurs only in one document, expression 3 produces a much larger number, increasing the weight of feature f_j in the corpus D.