Detecting Fake News with Supervised Learning

Benjamin R. Perucco

December 30, 2020

1 Definition

1.1 Project Overview

Our world is highly interconnected and it is of paramount importance that citizens are informed objectively about issues that influence and shape our world (like issues on geopolitics or climate change). The internet lead to a rise of news media (e.g. social media, news portals, etc.) to report on these stories. The vast amount of (online) articles available yield a new phenomenon called fake news. Fake news is false or misleading information presented as news and can reduce the impact of real news [1].

1.2 Problem Statement

This works' intention is about answering the question whether machine learning can be applied to classify a news article as truthful or fake. There are several articles available where Ahmed et al. classified news articles [2] or hotel guest reviews [3] to be truthful or fake. Ahmed et al. used natural language processing (NLP) models to transform text into a structured, mathematical representation and achieved accuracies of 92% [2].

Based on a set of features $\{f_1, f_2, f_3, ...\}$ extracted from news articles, a machine learning algorithm needs to be trained to find the relationship $\hat{\theta}$ between the a priori known classifiction of the news articles y ($= 0$ for truthful and $= 1$ for fake) and the set of extracted features $\{f_1, f_2, f_3, ...\}$. The relation can be written as

$$\hat{y} = \hat{\theta}(X) \tag{1}$$

where $\hat{y}$ is the predicted class label by the trained function $\hat{\theta}$ applied to a feature matrix X. $\hat{\theta}$ must be trained on available data, such that

$$\sum_{i=1}^{n} (y - \hat{y})^2 \tag{2}$$

is minimal. Here it is assumed that we have n news articles where it is already known whether they are truthful or not. A first step to be worked on in this report is to answer how to extract a feature matrix X from a set of texts (or documents). How reliable is the prediction of class labels on unseen news articles once the function $\hat{\theta}$ has been found.

1.3 Metrics

Expression 2 is rather used for regression problems where the predicted output is a continuous variable. For classification problems like news article classification into a truthful ($= 0$) and a fake class ($= 1$), the accuracy is a better metric. This can be nicely demonstrated by the confusion matrix

$$\begin{bmatrix} & 1 & 0 \\ 1 & \text{TP} & \text{FP} \\ 0 & \text{FN} & \text{TN} \end{bmatrix}. \tag{3}$$

The entries in the row indicate predicted classes and the entries in the columns represent the real classes. Furthermore, TP stands for true positives, TN for true negatives, FP for false positives and FN for false negatives. To asses the performance of $\hat{\theta}$, the accuracy a is considered, which can be calculated as

$$a = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FN} + \text{FP}}. \tag{4}$$

2 Analysis

2.1 Algorithms and Techniques

News articles must be converted into a structured, mathematical representation in order it can be used by machine learning techniques. The following chapters disucss how we can convert text into numerical features.

2.1.1 n-gram Model

The n-gram is usually defined 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 which is built 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 (1-gram), 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 (2-gram), 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]. Please note that punctuation is not considered when converting text into tokens.

In order to build frequency arrays for a set of texts (or documents), a common vocabulary needs to be built 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 based on the n-gram model. Usually for a set of tokens, only the m mostly occuring tokens in a corpus D are considered. In the following, the tokens are denoted as features as these construct the features (or independent variables) of a machine learning model.

2.1.3 Definitions

Let $\sigma(d_i, f_j)$ denote the number of occurances of feature f_j in document d_i. Then a feature matrix S can be built, where

$$S = \begin{bmatrix} \sigma(d_1,f_1) & \sigma(d_1,f_2) & \sigma(d_1,f_3) & ... & \sigma(d_1,f_m) \\ \sigma(d_2,f_1) & \sigma(d_2,f_2) & \sigma(d_2,f_3) & ... & \sigma(d_2,f_m) \\ \sigma(d_3,f_1) & \sigma(d_3,f_2) & \sigma(d_3,f_3) & ... & \sigma(d_3,f_m) \\ ... & ... & ... & ... & ... \\ \sigma(d_n,f_1) & \sigma(d_n,f_2) & \sigma(d_n,f_3) & ... & \sigma(d_n,f_m) \end{bmatrix}. \tag{5}$$

An element $\sigma(d_i, f_j)$ in matrix S (representing the document d_i and feature f_j) is abbreviated using the notation σ_{ij} for simplicity.

2.1.4 Term Frequency Model

The matrix S could be already used for machine learning. Features usually need to be normalized in machine learning to increase performance. Therefore, the term frequency (TF) model normalizes matrix S leading to matrix $\hat{S}$. An element $\hat{\sigma}_{ij}$ of matrix $\hat{S}$ is written as

$$\hat{\sigma}_{ij} = \frac{\sigma_{ij}}{\sum_{j=1}^{m} \sigma_{ij}}. \tag{6}$$

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 end up with a representation where the importance of each feature can be compared to other features in the same document.

2.1.5 Inverse Document Frequency Model

The inverse document frequency (IDF) model is used to define the feature importance not just in a document d_i but also compare its importance within a corpus D. A matrix E is introduced, where an element ϵ_{ij} of matrix E is 1 if $\hat{\sigma}_{ij} > 0$ (or $\sigma_{ij} > 0$). An inverse normalization is applied to the matrix E resulting in a vector $\hat{e}$. An element $\hat{\epsilon}_j$ of vector $\hat{e}$ is calculated as

$$\hat{\epsilon}_j = 1 + \log\left[\frac{|D|}{\sum_{i=1}^{n} \epsilon_{ij}}\right]. \tag{7}$$

Or spoken in plain language: in a corpus D which comprises of a set of documents $\{d_1, d_2, d_3, ..., d_n\}$, it is counted 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 7 results in 1, weighting feature f_j as 1. On the other hand, if a feature f_j occurs only in one document, expression 7 produces a much larger number, thus increasing the weight of feature f_j in the corpus D.

Finally, a matrix $\hat{P}$ is obtained as the element-wise product of the term frequency matrix $\hat{S}$ and the inverse document frequency vector $\hat{e}$, written as $\hat{P} = \hat{S} \odot \hat{e}$. For an element $\hat{\pi}_{ij}$ of matrix $\hat{P}$, this is written as

$$\hat{\pi}_{ij} = \hat{\sigma}_{ij} \cdot \hat{\epsilon}_j, \quad \text{element-wise for } j = 1, 2, 3, ..., m. \tag{8}$$

This term is denoted as the term frequency inverse document frequency model (TF-IDF).

2.2 Data Exploration

2.3 Benchmark

3 Methodology

3.1 Data Preprocessing

3.2 Implementation

3.3 Refinement

4 Results

4.1 Model Evaluation and Validation

4.2 Justification

5 Conclusion

5.1 Reflection

5.2 Improvement

References

[1] Fake news, Dec 30, 2020, `https://en.wikipedia.org/wiki/Fake_news`

[2] Ahmed H., Traore I., Saad S. (2017) Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. In: Traore I., Woungang I., Awad A. (eds) Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. ISDDC 2017. Lecture Notes in Computer Science, vol 10618. Springer, Cham. `https://doi.org/10.1007/978-3-319-69155-8_9`

[3] Ahmed, H, Traore, I, Saad, S. Detecting opinion spams and fake news using text classification, Security and Privacy, 2018, `https://doi.org/10.1001/spy2.9`