Labeling News

1 Introduction

With the ability to mass produce and spread information comes great risk. Misinformation and “fake news” have become well known concepts, yet we still struggle to identify them. There are people and instances who put a lot of effort into producing articles with the sole purpose of misleading people. There are, however, differences between real articles written by professional journalists and these mass produced “fake news”. This project focuses on finding these differences and using machine learning to identify which articles are fake, and which are true. Part two of the report will walk through a more detailed problem and idea formulation. Part three explains how the data is taken from its raw form into machine learning ready features and labels. Part four describes used features and machine learning methods as well as loss functions. Part five is about the conclusions that can be drawn from this project, such as whether or not it can be considered successful or not.

2 Problem Formulation and Dataset

2.1 Other contributions

With the problem of fake news being well known, the scientific community has made attempts and invented clever ways to approach this issue. [0] The most common way of analysis uses some word frequency analysis from the bread text of an article. This has been found to be useful in many natural language classification situations, with many methods available to link word usage to some labels. In this case, this is computationally intensive, and there are other ways to extract features. This project aims to use alternatives to counting all words from hundreds of thousands of media articles, and see whether the same level of accuracy can still be reached.

2.2 Idea for implementation

The data [1] used consists of two parts. There is one dataset containing true articles and one containing fake articles. Through web-crawling the true articles were retrieved from Reuters.com by the publisher, and the fake ones were news labeled fake by politifact (an american fact-checking organization.) The different columns in the data are: Title, Text, Subject and Date. The columns subject and date were left out and this project will not use them, since the goal is to use only textual and linguistic features. With the data set being relatively good for ML [2], the idea is to analyze word frequency only on the title, and combine that with feature extraction from the text and then use supervised learning to learn the relationship between these features and the label, that is either true or fake.

3 Data Preprocessing and Feature Extraction

Feature extraction from the title

The idea is to retrieve all the words used and create a common BoW (Bag of Words) and then count the occurrence of these words for each title . [3.1.] (The reference uses this method in different, yet equal context in the sense that both are cases of classification.) The first part of the process is word tokenization.[4.1] Stop words e.g. is, in and a don't bring a lot of relevant information, they are therefore removed.[4.2] After that, there are still a lot of words that are close to each other yet create their own instance in the BoW. Using a snowball/Porter2 stemmer of the nltk library [4.3] this problem is overcome by turning them into the stem of the word. The next step is wrapping the words into a frequency dictionary, where each unique word frequency is represented.[4.4] The final step is creating the BoW from all unique words and to create a matrix. [3.2] This can be done with DictVectorizer from the scikit-learn library. [5.1] Result matrix:

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{bmatrix}, \textit{An instance } a_{i,j} \textit{ represents the weight}(frequency) \textit{ of term } (word)\ j$$

$\textit{in title } i$

Each column is now a feature, representing the amount of times a word from the BoW is used. Each row, corresponding to a label or fake or true, now represents a title.

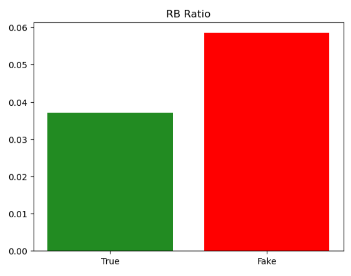
Example: Say we have a BoW containing: [photograph, use, republic] and a title "Photographer uses media to leak pictures of alcohol being used. This would in stem (photographer: photograph, uses & used: use) be represented in matrix as a row: { 1, 2, 0 } .

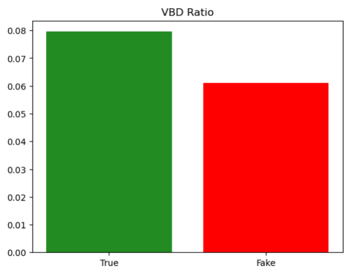
Feature extraction from the text

To get further information about used language, the text was used to create other features that represented language used. These were: average word length per article, average sentence length and the type of words used, nouns and adjectives etc. The counting of word types was done through Part-of-Speech (PoS) tagging. [3.3] By "tagging" each word with it's type, the ratio of eg. adjectives (shortened JJ) become a usable feature. POS-tagging method is also discussed in "A unique approach for detecting fake news". [0.2]

Example:

As seen in the graphs to the right. The ratio of adverbs (tagged 'RB', left) is significantly higher in fake news articles (red), while the usage of verbs (tagged 'VBD', right) is significantly higher in non-fake news (green.)





Result of feature extraction

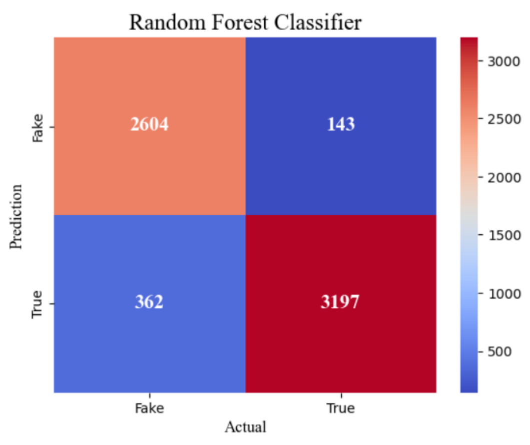
Combining the features from the title feature extraction and text extraction creates a large set of features that can all be analyzed for a relation to the label. [6] 10 % of the data was then removed to be used for final testing.

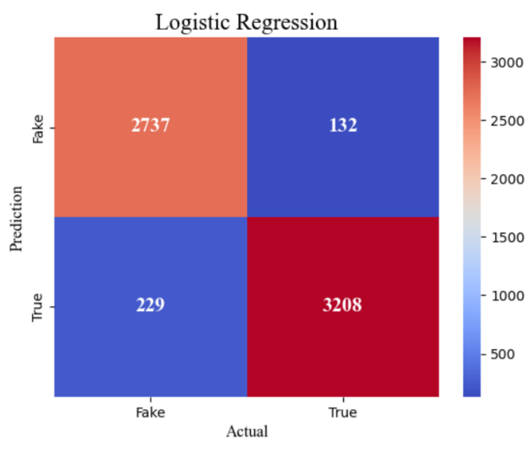
4 Methods

With such a wide amount of features (12000 + discrete word counts, 3 continuous statistics for the text and 10+ PoS tag ratios) and the data set given, some must be chosen. Of the words in the BoW they don't all bring value, since some might only occur a few times and thereby not give any reliable or relevant information. The machine learning model chosen is therefore LogisticRegression , formula: .[5.2] One reason for the choice is the ability of the model to assign weight to different variables and thereby deciding the relevance of them. This is key in this case, where the goal would be for the model to identify the text related statistics in the sea of word frequencies. The other reason for the choice is that when it is used with L1 loss function, it zeroes out a lot of irrelevant features.[7] Not only that but the amount of features with a weight of non-zero, can be adjusted indirectly by adjusting the hyperparameter C. Before training, the data rows will be split 80/20 (train/validation).[5.3] This enables the use of many features, while still keeping a sufficient number of instances in the validation set, so over/underfitting can be tested.

Another model that is good at finding feature importance is the decision tree. Especially when using many different ones to decide an average outcome as in a random forest classifier. [5.4] It is good at finding complex relationships between features and labels in the data, which in this case could prove beneficial. The parameters of the model will be tuned as it is a model that is very prone towards overfitting. This comes as the result of the complexity in the model combined with the large amount of features in the dataset. As the model in itself decides feature importance, the feature selection with the model will be done automatically during training based on which features correlate best with the training data. The splitting of the data and hyperparameter tuning will be done with cross-validation. The cross validation will be done with a randomized search algorithm. [5.4] The algorithm starts by splitting the data into smaller pieces, in this case 3 was chosen to remain adequate size. Then it chooses a set of parameters at random and evaluates their performance on the splits. This is iterated a few times, in this case 10 and then the best estimator is returned. This is a lightweight alternative to other similar ones, e.g. grid search algorithm. Since there are a lot of complex relationships between the features, the RFC will be applied as one of the alternative models. The hyperparameters will, like the features, be chosen based on performance during the hyperparameter tuning with random search.

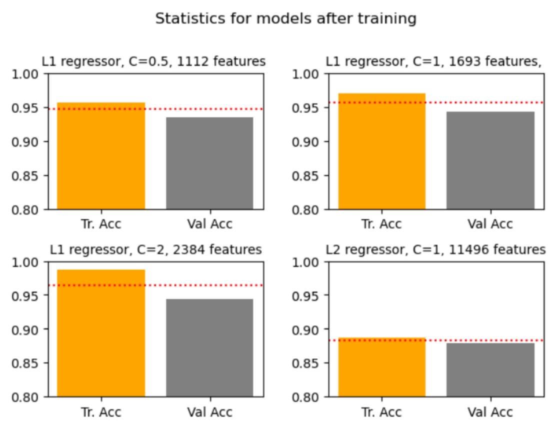
5 Results





Above the performance of the two models can be visually compared. The graphs represent the confusion matrices produced by the models. The upper rows are the ones predicted fake and the lower rows are predicted true. The left columns are actually fake and the right columns actually true. The logistic regression model outperforms the random forest classifier. They got 94% and 91% accuracy scores on validation data respectively. They both had about 1% higher score on the training data. Considering the risk for overfitting, especially with the random forest classifier, this is positive. Because of the higher score, the Logistic Regression model was favored in this project.

For testing the model hypothesis, four logistic regression models were trained on the training set of features and labels. The results are visualized to the left and it shows that the ones using lasso (L1) regression performed better than the one with ridge (L2) regression (bottom right). In addition, using a lower value for hyperparameter C meant that the model would perform worse on both training and validation data. Increasing it on the other hand, would increase overfitting. This is logical, since an increased C value meant more features taken into account. For example, C at .5 led to 1112 features, while C at two led to 2384 features. The L2 loss function led to having 11, 496 features, which certainly was too much considering the amount of data. This could explain its poorer performance. Finally, C=1 was chosen even though it had a little worse performance on validation data than C=2, because C=2 led to overfitting.



6 Conclusions

There are clearly relationships between the linguistic data and the truth value of an article, which this project managed to find using supervised machine learning, evident after the final model got a test score of 94%. It should be noted that the datasets [1] news are pretty homogeneous and gathered from the same place. It is also limited to news articles in English and there is no certainty that it will work for non-American news, as that has not been validated. It cannot separate between a user with a number as username posting on a unsecure website and a professionally certified journalist publishing on unbiased media platforms if the texts contain the right balance between word types and words used. This comes as a result of the model only analyzing the textual and linguistic components. The model, however accurate with this data, needs to be improved and developed to be adapted in practice. It did prove that the idea of the project was possible. The alternative strategy that this project set out to implement worked and ended up with a well-performing machine learning model. To make some sort of application which could work well in practice, context should be included. When predicting the truth value of social media posts for example, the properties of the user and application etc. could be used as features. Combined sets of data could also be used to remove bias.

References

[0] Similar project references:

[0.1] Published to geeksforgeeks, Fake news detection using Machine Learning, retrieved 10.5.2023: https://www.geeksforgeeks.org/fake-news-detection-using-machine-learning/

[0.2] A unique approach for detecting fake news, retrieved 10.5.2023, google scholar: https://www.researchgate.net/profile/Yash-Shukla-7/publication/336164499_An_Unique_Approach_For_Detection_of_Fake_News_using_Machine_Learning/links/5e48feb7458515072da0b97b/An-Unique-Approach-For-Detection-of-Fake-News-using-Machine-Learning.pdf

[0.3] An-Innovative-Approach-For-Detection-of-Fake-News-using-ML, retrieved 10.5.2023, google scholar https://sirsyeduniversity.edu.pk/ssurj/rj/index.php/ssurj/article/view/565/195

[0.4] Fake news detection model, retrieved 10.5.2023 https://www.kaggle.com/code/noorsaeed/fake-news-detection-model

[1] News Articles https://www.kaggle.com/datasets/emineyetm/fake-news-detection-datasets , 14.9.-23

[2] Checklist for good data, 14.9.-23 https://services.google.com/fh/files/blogs/data-prep-checklist-ml-bd-wp-v2.pdf

[3] Daniel Jurafsky & James H. Martin. Speech and Language Processing, 2014, Second Edition.

[3.1] Chapter 20.2.1 Feature Extraction for Supervised Learning

[3.2] Chapter 23.1.1 The Vector Space Model

[3.3] Chapter 5.3 Part of Speech tagging

[4] nltk (library), 17.9.-23

[4.1] word tokenizer https://www.nltk.org/api/nltk.tokenize.word_tokenize.html

[4.2] stopwords example https://www.nltk.org/search.html?q=stopwords

[4.3] snowball stemmer https://www.nltk.org/howto/stem.html

[4.4] FreqDist https://tedboy.github.io/nlps/generated/generated/nltk.FreqDist.html

[5] scikit-learn (library), 20.9.-23

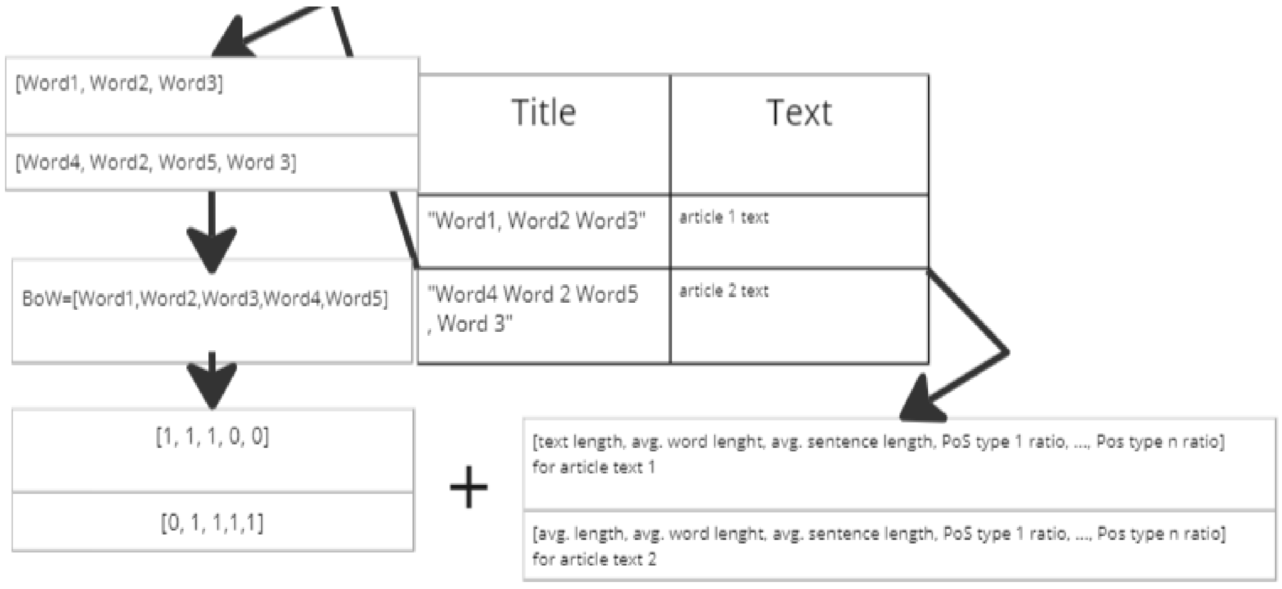
[5.1] DictVectorizer https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.DictVectorizer.html [5.2] LogisticRegression https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[5.3] Train_test_split https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html#sklearn.model_selection.train_test_split

[5.4] RandomForestClassifier https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[5.5] RandomizedSearchCV https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

[6] Feature extraction visualized.


[Word1, Word2, Word3]
[Word4, Word2, Word5, Word 3]
Title
Text
"Word1, Word2 Word3"
article 1 text
"Word4 Word 2 Word5 , Word 3"
article 2 text
BoW=[Word1,Word2,Word3,Word4,Word5]
[1, 1, 1, 0, 0]
[0, 1, 1,1,1]
+
[text length, avg. word lenght, avg. sentence length, PoS type 1 ratio, ..., Pos type n ratio] for article text 1
[avg. length, avg. word lenght, avg. sentence length, PoS type 1 ratio, ..., Pos type n ratio] for article text 2

[7] Difference between L1 and L2 loss in Logistic Regression, 21.9.-23
https://explained.ai/regularization/L1vsL2.html#why