Luca Buchoux

STOR 390

10/25/204

Midterm Paper

One of the hottest buzzwords being thrown around in recent years is “fake news”. Misinformation being spread by people who don’t know any better, or by those who know their information is false but have other motives. Either way, finding the truth through all the noise can be extremely challenging or even downright impossible. Furthermore, we live in a time when more and more people’s source for news is shifting to the internet where it is extremely easy to post anything and pass it off as the truth. Even those who still get their facts from traditional sources such as television are not free from this influence, as those sources could be regurgitating false information or half-truths from any of their sources. This is a huge issue because people can easily be swayed by what they hear and read, especially if it has the appearance of coming from a reliable source. Misinformation such as this often revolves around controversial topics and is specifically designed to make people believe something that is not true. Not only that, but as more and more misinformation finds its way into circulation, it makes finding the truth even harder. Luckily, thanks to various methods in machine learning, researchers have come up with many models that can help classify the validity of pieces of information. Also, as this issue has become more relevant, a lot of data has been compiled specifically to help train such models to help improve accuracy. The paper I have chosen to analyze is titled *Fake news detection using machine learning algorithms* by Sharma et al. in which the authors discuss the issue and related works, then fit 4 different classifiers on significant data and discuss the results.

Summary of Method

The overall system developed by the authors is split into three parts: static search, dynamic search, and URL search. Static search uses different machine learning classifiers to determine if an article is fake or not. Dynamic search takes key words from user searches to give the truth probability of news. URL search determines the authenticity of an inputted URL. Three methods of classification are used for the static search: Naïve Bayes Classifier, Random Forests, and Logistic Regression. For the dynamic search, the authors use Passive Aggressive Classification, and the URL search method is not explored in great detail for the remainder of the paper. Since the URL search is not highlighted and I am not as familiar with Passive Aggressive Classification, I will mainly be focusing on the static search. I will first explain the methods used then discuss the data used before looking at the results.

Naïve Bayes Classifier. The Naïve Bayes Classifier is a classification method built around Bayes theorem, which gives us a way to calculate conditional probabilities. In order to make Bayes theorem work for us, we make the naïve assumption that all our predictors are independent of each other. This allows us to calculate the probability of an observation belonging to a certain class using the information we already know. This assumption is rarely true in any real-world setting, but this method still performs surprisingly well even when the assumption does not actually hold.

Random Forests. In simple terms, random forests are almost like a combination of decision trees and K Nearest Neighbors. Random forests consist of a group of decision trees and classify observations by majority rule of each individual prediction by the trees. Each individual tree is made to be uncorrelated from one another, otherwise the forest would be comparable to a single decision tree. The methods to create uncorrelated trees are bagging and feature randomness.

Bagging is when each tree is trained on a random subset of the whole training data, which causes significant differences in tree structures since decision trees are sensitive to the data they are trained on. Feature randomness is when a decision tree is only allowed to pick from a random subset of features when splitting a node, increasing variation and reducing correlation between unique trees. These two methods allow for the entire forest to give better predictions by considering a wide variety of uncorrelated trees which look at a variety of features and subsets of the training data.

Logistic Regression. Like the Naïve Bayes Classifier, Logistic Regression gives us the probability of an observation belonging to a specific class using the logistic function. Using the probability, we can determine the odds of the event occurring, which in turn leads us to the logit function which is the log-odds of the same event. The logit function is essential for interpreting logistic regression, as each coefficient relating to our predictors gives us the log-odds change for one unit change of the predictor. This easily gives us the how much the odds change by exponentiating, which can easily be interpreted. Using this, we can classify our observations based on whether they are more likely to fall into one class or the other.

The paper uses two datasets when fitting their models: the LIAR dataset and REAL_OR_FAKE.CSV. The LIAR dataset is what is used for the static search, so I will focus on it. It consists of around 12,000 short statements from various sources which have been fact checked and assigned truth labels varying from pants-fire to true. To clean the data, the authors performed the following steps: removed punctuation, split text into units (words or sentences), removed common words that appear in any text, and reduced words down to their stems. Next, the text needs to be stored numerically, also known as vectorized. The first method is Bag-Of-Words, which gives each unique word a column and is given a 1 if present in the statemen or 0 if

not. The next method is N-grams, which are combinations of adjacent letters/word of length n, giving us an idea of what letters/words precede or follow each other. Finally, TF-IDF gives us the frequency of a word in a document relative to its frequency across all documents. This is achieved by multiplying the frequency of a term within a document (Term Frequency) by the Inverse Document Frequency, which increases for unique words and decreases for words found in many documents. TF-IDF can be used at word level and at N-gram level. These three vectorized fields are the predictors used to train the models in static search.

Summary of Results

The LIAR data was split using 3-fold cross validation, with 67% used for training and 33% used for testing. Models were fit using the three methods previously discussed—Naïve Bayes, Random Forest, and Logistic Regression—and confusion matrices were produced providing us with true/false positive and negative rates. Four metrics were used to assess each model: precision, recall, F1 score, and accuracy. Precision is the true positive rate among observed results, recall is the true positive rate among predicted results, and accuracy is the overall accuracy among the whole training data. F1 score is a metric for model accuracy that combines precision and recall by assessing performance by class rather than the entire dataset. In the end the results came out as follows: Naïve Bayes had a precision of 0.59, a recall of 0.92, F1 score of 0.72, and an accuracy of 0.6; Random Forest had a precision of 0.62, a recall of 0.71, F1 score of 0.67, and an accuracy of 0.59; and finally Logistic Regression had a precision of 0.69, a recall of 0.83, and F1 score of 0.75, and an accuracy of 0.65. Thus based on overall performance and the model with the highest accuracy, the authors chose Logistic Regression as the final model. After tuning the parameters using a method called grid search parameter optimization, the researchers were able to further increase the performance of the model to an accuracy of 80%.

Normative Consideration

The issues that these methods are trying to tackle are quite significant, and as such a lot of ethical and moral implications should be considered. One of the main issues I could see arriving from this study is how easily the same tools used to find the truth could be spun to foster further misinformation. Specifically, training these methods on a biased or purposefully altered set of data could be used to falsely propagate fake news. Even without altering how the models are trained, none of them are perfect, as is the case with all models. As such, predictions that result in a false positive or a false negative could be used to alter the truth by those who rely too much on the models or paint them to be the absolute truth as a means to deceive. Thus it is important to treat these models how all other models, such as Chat GPT and the like, are treated. They should be seen as tools for thinking, not replacements. Another issue more specific to the authors' methods that I noticed was the labeling system from the training data. Reading the documentation for the data, the labels for truth were classified on a scale with 6 values, ranging from pants-fire to true. The authors specify that they break the classification down into simply true or false, but it raises the question of how we interpret the truth. Some things are completely true, and some are completely false, but there is a lot out there that is in between. Is it right to throw out information that contains some of the truth? Is it still not beneficial to consume information with the disclaimer that some things might not be entirely accurate? As a whole, the methodology might oversimplify its discrimination, and it could be beneficial to keep the truth classification as a scale. The solution may be to break larger pieces of information down into smaller units and give a truth rating or score for the whole, based on the classification of the smaller pieces. This is something that could be explored further regarding the methodology laid out in this paper.

Works Cited

Sharma, Uma, Sidarth Saran, and Shankar M. Patil. "Fake news detection using machine learning algorithms." *International Journal of creative research thoughts (IJCRT)* 8.6 (2020): 509-518.