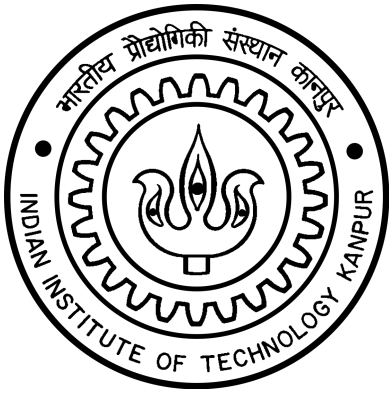

Project Report

Fake News Detection

Instructor:

Prof. Arnab Bhattacharya

Pankaj Kumar

15817475

Contents

1 Abstract

In the modern era, social network and online media houses are gaining popularity. Meanwhile, fake news is spreading like a wildfire. In the past few years, I have observed that fake news is creating lynching, riots, mob attacks, etc. In a recent election in India, maliciously-fabricated media has played a central role in Indian political discourse. A group of intellectuals is trying to find better ways to identify and label fake news to protect the public from misinformation. I aim to build a reliable model that can classify a given news article as fake or real.

2 Introduction

All of us has observed, with the advent of technology, there is an equal chance of negative and positive effects. Fake news is one of the adverse impacts among them, which has created a lot of chaos between different section of society around the world. Since accessing information from the internet is free for everyone, but at the same time, maliciously-fabricated media has erase the line between true and false. The recent development of Machine learning provides a possible solution to automate this process. However, accurately and repeatedly identifying fake news is still proven difficult due to the complex nature of human language.
To check the authenticity of a news article, we need to filter the information we receive every day. With this motivation, I tried to use my knowledge of python, Natural Language Processing, and machine learning to build a model that can act as a filter for the future news article.

3 Literature Review

I have gone through the previous work mentioned in the references section. I have also implemented some methods and mechanisms that have been discussed in these works as well as possible extensions that I could think of. In this section, I will discuss the approaches that have been presented in these works that I have used in the project as well.

3.1 Naive Bayes

Given the size of our feature space, I determined that Naive Bayes is an appropriate method to begin our analysis. Drawing from the lecture notes of CS771, the maximum-likelihood estimates for the model parameters are:

$$\phi_{j|y=1} = \frac{\sum_{n=1}^{m} 1(x_j^i = 1 \bigwedge y_j^i = 1) + 1}{\sum_{n=1}^{m} 1(y_j^i = 1) + 2} \tag{1}$$

$$\phi_{j|y=0} = \frac{\sum_{n=1}^{m} 1(x_j^i = 1 \bigwedge y_j^i = 0) + 1}{\sum_{n=1}^{m} 1(y_j^i = 0) + 2} \tag{2}$$

Using Naive Bayes algorithm, I identified the top-k tokens that were found to be the most indicative on the classification of the example. This was computed by finding the k/2 tokens which have the highest posterior probability of being in fake news, and the k/2 tokens with the lowest posterior probability of being in fake news. The following expression was used to rank the tokens by their indication of fake news:

$$TokenRank = \frac{exp(\phi_{j|y=1})}{exp(\phi_{j|y=0})} \tag{3}$$

The k/2 most indicative tokens for each class was used to form a new feature space for our Logistic Regression model. These tokens were also examined heuristically to ensure they pass the eye-test given our team's knowledge of contemporaneous fake news.

3.2 Logistic Regression

Due to its simplicity and elegance, Logistic Regression (LR) was used as the third algorithm within the AverageHypothesis model. The LR model uses gradient descent to converge onto the optimal set of weights () for the training set. Where J is the loss function and alpha is the learning rate. For our model, the hypothes used is the sigmoid function:

3.3 PassiveAggressiveClassifier

Initialize, w = (0,0,....,0)
Monitor a stream:
receive new doc, d = $(\mathrm{d}_1, d_2, ..., d_v) then$,
apply TfIdf and normalize d using porter stemmer
predict positive,if $d^t w > 0$
Observe true class: $y = \pm 1$
want to have:
$d^t w \geq +1$ if positive $(y = +1)$
$d^t w \leq -1$ if negative $(y = -1)$
same as:$y(d^t w) \geq 1$
Loss, L = max(0, 1-$y(d^t w)$)
Update: $\mathrm{w}_{new} = w_{old}$+y $\times L \times d$

3.4 Support Vector Machine

Due to it's robustness, a support vector machine (SVM) was used as the second algorithm in our Average-Hypothesis model. The SVM algorithm used uses a hinge loss that seeks to maximize the margin between the two classes of data. The SVM algorithm uses a second-order Gauss kernel that operates on the full 5078 token feature space. The expression for this kernel is given by the following expression:
Note that this expression is provided for the 1-D case. In retrospect, the selection of this high-order kernel seems rather naive, since it may have caused the SVM model to over fit the training set.

3.5 Neural Network

A one-layered neural network model was used on the 80 tokens identified to be most causal to a sources classification. The hidden layer neurons uses sigmoid activation functon and, the output layer uses the softmax activation.
Also, ReLU and tanh function were tested for the activation function of the hidden layer. Although the results from sigmoid are not good enough to be used as compared to other models discussed above, it was better than ReLU and tanh activation function.

Figure 1: A view Neural Network Architecture

4 Models and Data-sets

In this section, I describe the models and methods that I have used to obtain the results. The datasets that I have used are the following:

- **Fake News Dataset** - These data set consists of the text of news, and it's headline as input. It has also been mixed with fake news articles and is a binary classification data set which if I solved can be used directly as a filter for future news articles. The link for this data set :
  https://s3.amazonaws.com/assets.datacamp.com/blog_assets/fake_or_real_news.csv.

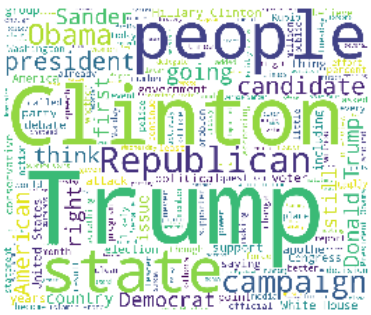

Figure 2: True news tokens

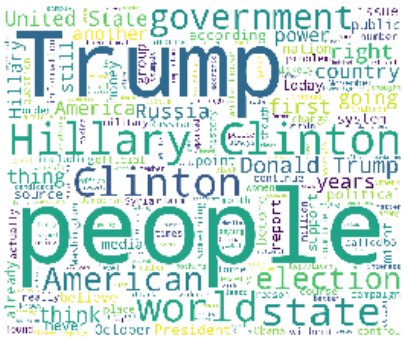

Figure 3: Fake news tokens

- **Pre-Processing** - Dataset has no missing values and class imbalance problem. Each tuple corresponds to a news article headline and their body which is mainly text data so to make available for the model, first I mapped label from string to boolean as Fake $\rightarrow$ 0 and Real $\rightarrow$ 1 then removed punctuation marks, used NLTK in python to tokenize the body title, and then Word stemming and finally joined all the space separated string.

  Following are the models that I have used:

  - **Baseline** - For a baseline, I have chosen to work with Naive Bayes and Logistic Regression in which I fed the output of CountVectorizer, TfidfVectorizer and hashVectorizer after pre-processing.
  - **Logistic regression** - Logistic regression has parameter θ which was learned during training and make predictions as follows: $P(y|x) = h_\theta(x) = g(\theta^T x)$ where g is the sigmoid function. We predict a value of 1 if $P(y|x) > 0.5$ and 0 otherwise.
  - **Support vector machine** - I have used Soft- Margin SVM, has parameter C controls the trade-off between large margin vs small training error with linear kernal.
  - **Neural Network** - I chose a network with 3 hidden layers with 10 neuron units on each layer, depending on the dataset it may differ. I used relu as the activation function. I worked with datasets having 2 classes so the binary cross entropy loss with the Adam optimizer was used and trained the network in steps with mini-batches randomly sampled from the training dataset.

5 Results

First of all, I fed the output of CountVectorizer, TfidfVectorizer and hashVectorizer to Naive Bayes model and got an accuracy of 89.3%, 85.7% and 90.2% respectively on test data. So, for further analysis I used hashVectorizer and fed its output to following models:

Feature Set	**Naive Bayes**	**Logistic Regres.**	**Passive Aggress. Classifier**	**Support Vector Machine**	**Neural Networks**
Accuracy					
Body + Title	0.89	0.91	0.92	0.93	0.86
Body	0.89	0.91	0.92	0.92	0.87
Title	0.88	0.91	0.92	0.93	0.89

The results above have been compiled by splitting the dataset into 3 part in the ratios 7:1:2 where the first part is meant for training the classifier, the second part for cross-validation. After we obtained the rules from trained classifiers, it was tested on last segment of data.

The accuracy for this specific dataset is highest for Support Vector Machine. The accuracy of Neural networks is lowest as it may be due to overfit.

6 Future Work and Applications

A lot of the results circle back to the need for acquiring more data. Generally speaking, simple algorithms perform better on less (less variant) data. Since I had less data, SVM PassiveAggressiveClassifier, and Logistic Regression outperformed Neural Networks, and Naive Bayes did not perform well. Given enough time to acquire more fake news data, and gain experience in python, I will try to better process the data using n-grams, and revisit Artificial Neural Network algorithm. I tried to tweak all knobs of various algorithms.

References

[1] http://cs229.stanford.edu/proj2018/poster/125.pdf

[2] http://cs229.stanford.edu/proj2017/final-reports/5244348.pdf

[3] NLTK3.2.5:NaturalLanguageProcessingToolkit// .¡https://pypi.python.org/pypi/nltk¿

[4] https://www.bonaccorso.eu/2017/10/06/ml-algorithms-addendum-passive-aggressive-algorithms/

[5] https://s3.amazonaws.com/assets.datacamp.com/blog$_a ssets/fake_o r_r eal_n ews.csv$