Detection of Fake News

Saide Tang and Anyang Lu

Introduction

The fake news is a made-up story with an intention to deceive and mislead readers. The fake news has a significant impact on our social life. To differentiate and eliminate fake news is important for the media. The object of this project is to apply the techniques of natural language processing to detect and label the fake news from the true news. In this project, we focus on the true and fake politics news during the US presidential election in 2016, when fake news was very popular in the media.

The data source is from Kaggle, including two files true_news and fake_news. There are 21417 pieces of true news and 23481 pieces of fake news from the two files. In each file, there are 4 attributes, including title, text, subject and date. The text is the content of the news with more than 1000 words in most news. The true_news dataset has two subjects, including politicsNews and worldnews. The fake_news dataset has six subjects, including government news, middle-east, politics, US_news, world news and left-news. The date is the published date of news, ranging from January 12, 2016 to December 31, 2017. Table 1 shows the part of the original dataset of fake_news.

title	text	subject	date
TRUMP DITCHES PRESS To Make "Last Minute'	President-elect Donald Trump played a	politics	31-Dec-16
COUNTDOWN TICKER: Obama Leaves Office I	(function(){ var s=document.createElem	politics	31-Dec-16
BOOM! TOMI LAHREN'S Top Tips For Liberals		politics	31-Dec-16
YES, OBAMA···There Is A Magic Wand! [Video]		politics	31-Dec-16
HA! DONALD TRUMP'S Unusual New Year's Tv	President-elect Donald Trump is throwi	politics	31-Dec-16

Table 1 A representative part of the fake_news dataset

Data Cleaning

Politics is the common subject of both true_news and fake_news datasets. So we selected the news of politicsNews subject from true_news dataset and news of politics subject from fake_news dataset as our data source. More specifically, we chose the news published in the year of 2016, when the US presidential election happened. After dropping the news with empty content, we achieved 4715 pieces of

true news and 2473 pieces of fake news. We combined these 7188 observations as our working data source for this project. Applying the regular expression techniques, we further removed the punctuations, url, html and stopwords from both Title and Text columns. After the data cleaning, we got the dataset as shown in Figure 1.

	title	text	subject	date	label
0	trump ditches press make “last minute” surpris...	presidentelect donald trump played round golf ...	politics	2016-12-31 00:00:00	Fake
1	countdown ticker obama leaves office in…54321…	function var sdocumentcreateelementscriptssrcw...	politics	2016-12-31 00:00:00	Fake
4	ha donald trumps unusual new years tweet “many...	presidentelect donald trump throwing private n...	politics	2016-12-31 00:00:00	Fake
7	sheriff clarke destroys idiocy gun control dem...	usual milwaukee outspoken sheriff david clarke...	politics	2016-12-30 00:00:00	Fake
8	putin pushes reset button america surprising move	chess match political wills putin coin well ob...	politics	2016-12-30 00:00:00	Fake

Figure 1. The head of dataset after data cleaning

To get a preliminary analysis about the content of the select news, we applied wordcloud technique to both title and text data. Figure 2 shows the wordcloud result based on the title data. We can see the words like “trump” and “clinton” are the hot topics, which were related to the 2016 US presidential election.

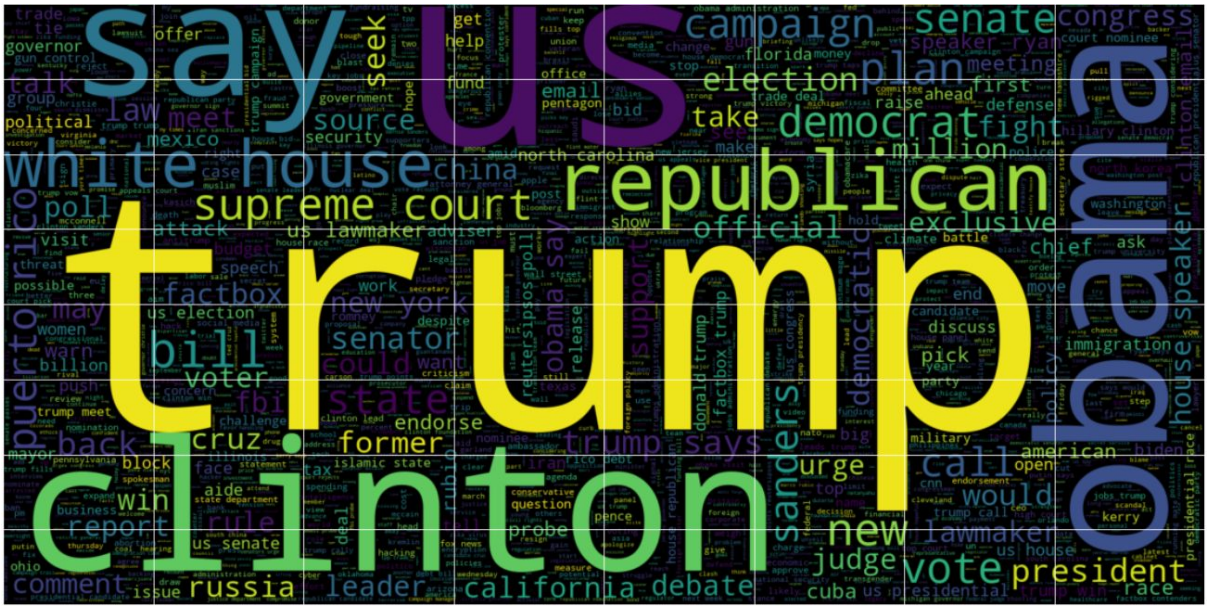

Figure 2. The wordcloud result based on the titles of the news

For the purpose of model training and testing, we split the total 7188 observations as training dataset and test dataset with the ratio of 80:20.

Methods and Discussion

Since we have both “title” and “text” data, we would build models with each of these two data sources in this project. For the type of input, we would try both TF-IDF and embedding methods. For the model building, we start from simple methods, including Naive Bayes classification and simpler neural network architecture. We would also try to apply RNN techniques, such as the LSTM method to build complex models. We will finally compare the accuracies (F1 score and confusion matrix) of different methods, and analyze the differences between different models. In this Interim report, we only use the title data source for model building.

Naive Bayes model based on TF-IDF

Applying the TfidfVectorizer module, we can vectorize the bag of words and calculate the TF-IDF value of each word. Then we applied the MultinomialNB module to build Naive Bayes model. We achieved both modules at the same time through the make_pipeline module. Based on the obtained model, we predicted the label of the test dataset. Compared with the true label, we can get the micro-averaged F1 score and macro-averaged F1 score as 0.907 and 0.894, respectively, shown in Table 2.

Neural Network model based on TF-IDF

Applying the TfidfVectorizer and MLPClassifier modules through the make_pipeline function, we were able to build neural network models based on TF-IDF. To compare the neural network models with different architectures, we built one model with one hidden layer with 50 units and one model with four hidden layers with 200 units in each layer. The obtained F1 scores are shown in Table 2. Based on F1 scores, we can see the model with more hidden layers and units didn’t improve the prediction performance. Figure 3 shows the confusion matrix of the prediction based on the neural network model with four hidden layers and 200 units in each.

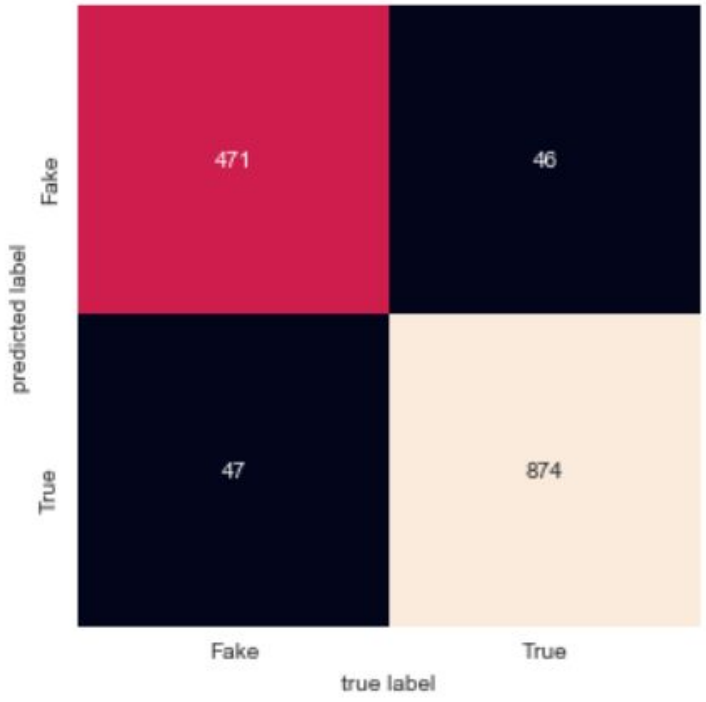

Figure 3. Confusion matrix of prediction by Neural Network based on TF-IDF with hidden_layers=(200,200,200,200)

Neural Network model based on embedding

In this project, we use 100d Glove embeddings and calc_doc_embedding function in hw3 to build an input matrix of embedding vectors using the centroid formula. Using the MLPClassifier module from Scikit-learn package, build a neural network model on the training data and predict the labels for the test data. Using embedding vectors as input, we fitted the neural network architectures with one hidden layer with 50 units as well as a model with 4 hidden layers and 200 units in each layer. Table 2 shows F1_scores obtained by two neural network models. We can see the performances of the simple model and the complex model are similar.

Model	Micro-averaged F1_score	Macro-averaged F1_score
Naive Bayes based on TF-IDF	0.907	0.894
Neural Network based on TF-IDF hidden_layers=(50)	0.936	0.930
Neural Network based on TF-IDF hidden_layers=(200,200,200,200)	0.935	0.930

Neural Network based on embedding hidden_layers=(50)	0.912	0.905
Neural Network based on embedding hidden_layers=(200,200,200,200)	0.913	0.908

Table 2 Micro-averaged F1_score and Macro-averaged F1_score of different models

Further Work

So far we built the simple Naive Bayes model and neural network model based on TF-IDF and embedding inputs. The data we used is only the titles of the news. The performances of the models are fine. As the forward work, we will try to build RNN models with the LSTM method. We would tune the parameters of the RNN models to check the model performance. Moreover, we plan to build models based on the text data, which can offer more language information.

Reference

Ahmed H, Traore I, Saad S. “Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques”, International Conference on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments-ISDDC 2017,127-138.