Classifying True & Fake News

A Problem More Than Just Classification

劉芊妤
資管三
National Taiwan University
Taipei, Taiwan

黃珮雯
資管三
National Taiwan University
Taipei, Taiwan

徐芊綺
資管三
National Taiwan University
Taipei, Taiwan

林昱辰
資管三
National Taiwan University
Taipei, Taiwan

ABSTRACT

In this project, having learned the techniques of text mining, we will attempt to perform classification on news dataset with the classes “True” and “Fake” by implementing the algorithm of the models all by ourselves, without any help of the advanced classification packages. Using Chi-square feature selection, with different number of features as a parameter, combined with three different models, including k-Nearest Neighbors, Rocchio, and Naïve Bayes Multinomial classification, the best model according to the 10-fold cross validation will be chosen in terms of the overall weighted average F1 accuracy, where the model will be further tested its performance.

MOTIVATION

Last year, the U.S. presidential election was the talk of the world. The vote of each person could possibly affect the result of the election, and even the political situation of most countries. Thus, many websites may spread “fake news” that is composed of false stories presented as news, with a view to changing one’s political view. However, such fake news without authenticity and neutrality, is in fact harmful for the public. Therefore, we would like to take the opportunity to try performing classification on documents into true news or fake news, which we believe should benefit the general public by reducing the information distortion among the media.

1 Dataset Introduction

Let’s begin by first introducing the dataset[1] used in the experiment. The dataset contains 44,898 news documents, 21,417 of which are true news, and the other 23,481 are fake news.

(1) *True.csv*

title	text	subject	date
As U.S. budget fight looms, Republicans flip their fiscal script	WASHINGTON (Reuters) - The head of a conservative Republican faction in the U.S. Congress, who voted...	politicsNews	December 31, 2017
U.S. military to accept transgender recruits on Monday: Pentagon	WASHINGTON (Reuters) - Transgender people will be allowed for the first time to enlist in the U.S. m...	politicsNews	December 29, 2017
Senior U.S. Republican senator: 'Let Mr. Mueller do his job'	WASHINGTON (Reuters) - The special counsel investigation of links between Russia and President Trump...	politicsNews	December 31, 2017
Trump wants Postal Service to charge 'much more' for Amazon shipments	SEATTLE/ WASHINGTON (Reuters) - President Donald Trump called on the U.S. Postal Service on Friday to...	politicsNews	December 30, 2017

Table 1: *True.csv* dataset (partial): title, text, subject, and date.

[1] Clément Bisaillon, *Fake and real news dataset: classifying the news* (2020), https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset

(2) *Fake.csv*

title	text	subject	date
Donald Trump Sends Out Embarrassing New Year's Eve Message; This is Disturbing	Donald Trump just couldn't wish all Americans a Happy New Year and leave it at that. Instead, he had...	News	December 31, 2017
Drunk Bragging Trump Staffer Started Russian Collusion Investigation	House Intelligence Committee Chairman Devin Nunes is going to have a bad day. He s been under the as...	News	December 31, 2017
Sheriff David Clarke Becomes An Internet Joke For Threatening To Poke People 'In The Eye'	On Friday, it was revealed that former Milwaukee Sheriff David Clarke, who was being considered for ...	News	December 30, 2017
Trump Is So Obsessed He Even Has Obama's Name Coded Into His Website (IMAGES)	On Christmas day, Donald Trump announced that he would be back to work the following day, but he i...	News	December 29, 2017

Table 2: *Fake.csv* dataset (partial): title, text, subject, and date.

As seen above, the two tables consist of true and fake news text and their title, subject and published date.

2 Methodology

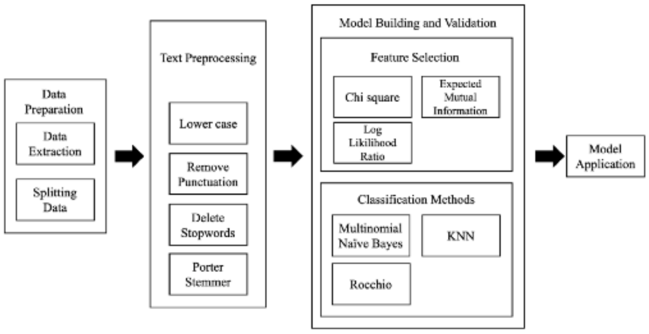

Figure 1: Flow chart of the overall process of the experiment.

2.1 Splitting train/test data

Since we want to evaluate our best model's performance on those documents which are not in the training dataset, the original data will be divided into training documents and testing documents by using a random approach with the use of **random**[2] Python package.

1. *rand_N*: Randomly choose an integer between 1000 and 5000, which will determine the size of the test set.
2. *rand_true*: Randomly choose a decimal between 0.4 and 0.6, determining the proportion of true news in the testing documents while not making the data too imbalanced in terms of the distribution of the classes. Then, *rand_fake* is computed by 1 - *rand_true*.
3. *ind_true_test*, *ind_fake_test*: Randomly sample the indices of documents in the set of true news and fake news with the size of *rand_true*, *rand_fake*, respectively.

2.2 Preprocessing

Upon acquiring the dataset, the preprocessing is the first step before using the text in any analysis. The tokenization and normalization process includes turning all the document words into lower cases and removing punctuations and accents, removing stop words using the **Stop List**[3], and having each tokens stemmed using **Porter Stemmer** in **NLTK**[4] Python package to reduce the variety of grammatical forms of the same term. At this point, another removal of stop words operation is performed again, so as to make sure that the data are clean enough without meaningless tokens after stemming. The terms for each document are stored individually in an entry of the DataFrame, where an additional label indicating its trueness is attached to the list of terms.

	terms	label
0	[washington, head, conserv, republican, factio…	1
1	[washington, transgend, peopl, allow, time, en…	1
2	[washington, special, counsel, investig, link,…	1

[2] *random- Generate pseudo-random numbers*, Python 3.9.1 Documentation, https://docs.python.org/3/library/random.html

[3] *stop list*, IR linguistics utilities, IR resources, School of Computing Science, University of Glasgow, http://ir.dcs.gla.ac.uk/resources/linguistic_utils/stop_words

[4] *nltk.stem.porter*, NLTK 3.5 documentation, https://www.nltk.org/_modules/nltk/stem/porter.html

	terms	label
3	[washington, trump, campaign, advis, georg, pa...	1
…	…	…
41285	[centuri, wire, say, report, earlier, unlik, m…	0
41286	[patrick, henningsen, centuri, wireremembered, oba...	0
41287	[centuri, wire, say, jazeera, america, histori…	0
41288	[centuri, wire, say, predict, year, look, host..	0

Table 3: *train_df*, DataFrame structure storing normalized terms in documents.

2.3 Feature Selection

Because the whole dictionary size is tremendous, features selection can be employed in order to achieve the similar accuracy result while speeding up the computation time. Initially, we implemented three different methods: Chi-square selection, Expected Mutual Information selection and Log-Likelihood Rate selection respectively.

Since there are two classes, the algorithm we design will first compute the Chi-square scores for all terms in the True class and Fake class separately, and choose the $\frac{k}{2}$ features which have the largest scores among all terms, where k represents the number of total features including both classes. Next, concatenate the two lists of features together, forming the final list of selected features.

In general, under Chi-square selection, True class have the following terms in the text cloud of Figure 1, including "Presid", "Unit", "State", … etc.

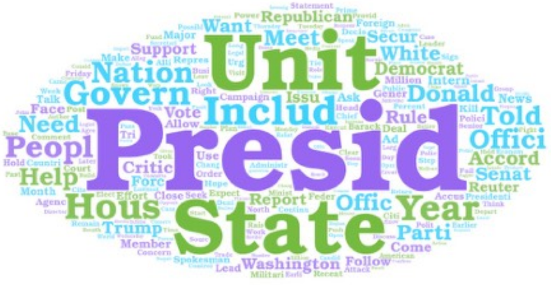

Figure 1: True class features under Chi-square selection, expressed in text cloud.

On the other hand, Fake class include the terms in the below text cloud of Figure 2, where "Trump", "State", and "Time" are among the most significant features.

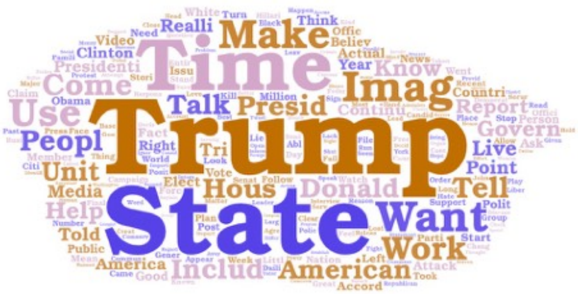

Figure 2: Fake class features under Chi-square selection, expressed in text cloud.

Note, quite importantly, that there are several features repetitively occurring in both classes, such as "State", "Presid", "Trump", and many more. This could severely affect our classification, as will be discussed in the following text.

2.4 Classification Model

With a view to comparing among classification models, three different algorithms will be implemented: Naïve Bayes Multinomial Classification and another two vector-based classification models— k-th Nearest Neighbor ($k = 3$) and Rocchio. These three models will be paired with several different parameters in the cross validation phase to get the best model.

3 10-fold Cross Validation

3.1 Splitting the training/validation sets

In order to perform the cross validation, by definition the documents in the validation set should be randomly chosen. Thus, we shuffle the original training data and randomly split it into ten folds. However, because the number of the training data is not divisible by ten, the rest of the data are assigned evenly into the 10 folds.

3.2 Approaches to cross validation

Aware of the fact that the feature size may also affect the result of the classification, we set the number of features selected to be a parameter in our model selection, where the feature size could be among one of the four: 300, 500, 800 and 1,000 features. Using different numbers of features and three classification models, we will perform all 12 combinations of training methods in each of the ten validating iterations. While comparing the selected features with different approaches, we found that the features chosen with three feature selection methods are not very different from each other, as there are some repetitive terms among the three result lists of features. Therefore, for the sake of simplicity, we only choose Chi-square score to be the main selection technique here.

3.3 Cross validation

In each iteration of the cross validation, nine of the ten folds are merged as the new training data set while taking the last fold as the validation data. We conduct feature selection to the new training data, and use the selected features to get the result of classification based on the three different training methods. Afterwards, F_1 score is obtained by calculating the precision and recall of the result. Moreover, since the number of documents in each fold is not necessarily equal, we use weighted average to compute F_1 so that the size of the data will also be factored in the validation score.

3.4 Comparison between the training methods

	Naïve Bayes Multinomial	Rocchio	*k*-th Nearest Neighbors
300 features	0.642137360433312	0.305591175519718	0.860730230730849
500 features	0.644273517484114	0.528114540829192	0.858480488749533
800 features	0.639451957410907	0.591272705139115	0.862005620872455
1,000 features	0.641470251937586	0.511333212293435	**0.862478233515475**

Table 4: Cross validation result with 12 different training methods.

According to **Table 4**, it is clear that kNN outperforms Naïve Bayes Multinomial and Rocchio. In terms of the numbers of features, though there are no significant differences between the results, the best performing model has to be determined. It turns out that, when choosing 1,000 features, F_1 scores the highest among them all, approximately 0.862. Therefore, in the next section, we will use kNN as the training method and 1,000 as the number of features to get the classification result of the testing data to evaluate the performance.

4 Model Performance Evaluation

4.1 Test performance— F_1 measure

As the cross validation result suggests, kNN with 1000 features is the best among all of the models. Thus, in testing, we will deploy kNN with 1000 features to classify whether the document belongs to true or fake news. As per shown on Table 5, the precision of this model is 0.39, recall is 0.60, and F_1 score is 0.48, which is not a fairly good performance.

	testing performance
precision	0.397121535181237
recall	0.608163265306122
F_1 measure	0.480490164463076

Table 5: Testing performance score.

4.2 Analysis on the difficulty of the problem

Based on the result, we decided to look deeper into the problem and issues that may have caused the bad performance. Here are five possible reasons that may be the reason of such inaccuracy.

1. No matter which feature selection method we use, both True and Fake news classes include some repetitive terms, including “State”, “Trump”, “Hous”, “Presid”,…, and so on. The redundancy rids these words off any discriminating power and the classification problem is consecutively much more difficult.

2. *k*-th Nearest Neighbors with 1000 features, which is our chosen model to perform testing, is a nonlinear classifier, meaning that it is highly possible for the model built in the training process to overfit with some features.

3. Of course, the definition of “True” and “Fake” news is not clear enough, which is solely the heart of the problem of classification for true and fake news. The criteria for classifying real and fakes news is obscure. Should the article belong to “True” news because it comes from a reputable website? Or because of some particular professional terms and jargons used in the news? Or perhaps based on whether the event described actually happened? Labeling the news as True or Fake requires humans to inspect and make a judgement on the trueness of the document, and sometimes such way of verification could be subjective and lacking accurate definition.

4. Even if the dataset utilizes the terms used in the news as the standards to classify, what if the writers of the fake news also use these terms frequently used in true news to confuse and delude the readers? Besides, fake news in this dataset is collected from a variety of different sources, which means that many fake news may generate different language models with very different use of terms adopted in the news. In this case, it obviously becomes fairly difficult to classify all of them correctly, since fake news may even have low similarities with each other.

5. Simple text classification based on features may not be the most suitable method to solve this problem. For example, sentiment analysis, natural language processing, and BERT are all possible alternatives to discern true news from wrong ones. However, since the instructor told us that the project should be done with self-crafted code, we are not allowed to use complex packages to perform the classification task. With the limit, more attention is placed in coding instead of exploring and experimenting with all other possible solutions or methods.

5 Conclusion

Summing it up, there are many reasons and factors that make the classification result not satisfactorily accurate. In near future, we will first attempt to clarify the definition of true and fake news and employ other techniques on the same dataset to check the improvement. Moreover, our team has also searched for some paper to get more insights about fake news detection. As indicated, detecting fake news is believed, among the academies, to be an extremely complex and difficult task. Recent advance in technology also makes the creation and spreading process of the fake news easier. , which only raises the urgency and importance of the solution to this problem to a higher level. After reading some academic research papers, it is suggested that *N*-gram is often employed to help detect the fake news from others[5] . In conclusion, we may need more methods other than our current model to discern fake from true news.

MEMBERS’ WORKLOAD

劉芊妤：text preprocessing, splitting train/test set, feature selection (log-likelihood ratio), testing model performance, report drafting, presentation slides edition

黃珮雯：splitting train/test set, feature selection (Chi-square, expected mutual information), kNN model implementation, report revision, oral presentation

徐芊綺：NB multinomial classification model implementation, cross validation implementation, report drafting

林昱辰：Rocchio model implementation, report drafting

REFERENCES

1. Clément Bisaillon, *Fake and real news dataset: classifying the news* (2020), https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset
2. *random- Generate pseudo-random numbers*, Python 3.9.1 Documentation, https://docs.python.org/3/library/random.html
3. *stop list*, IR linguistics utilities, IR resources, School of Computing Science, University of Glasgow, http://ir.dcs.gla.ac.uk/resources/linguistic_utils/stop_words
4. *nltk.stem.porter*, NLTK 3.5 documentation, https://www.nltk.org/_modules/nltk/stem/porter.html
5. Ahmed H, Traore I, Saad S. “*Detecting opinion spams and fake news using text classification*”, Journal of Security and Privacy, Volume 1, Issue 1, Wiley, January/February 2018.

[5] Ahmed H, Traore I, Saad S. “*Detecting opinion spams and fake news using text classification*”, Journal of Security and Privacy, Volume 1, Issue 1, Wiley, January/February 2018.