CIS 520 Fall 2020, Project Report

Fake News Detection

Team Members:

- Haowei Liu; Email: hwliu@seas.upenn.edu
- Yuchen Zhang; Email: zycalice@seas.upenn.edu

Abstract

With the popularization of digital media, news has become readily available to human at a click of finger. This development drastically increased our access to information, information that are non-discriminatory in nature. The multitude of sources that are more often than not contradictory has created new challenges for human - choosing what we should believe in. In this project, we explore machine learning approaches to tackle this problem. Using news that have been flagged by human as either fake or real as training dataset, we design classification models that can identify the legitimacy of a news article given its content. Specifically, we apply natural language processing techniques for text feature extraction, and consequently classification models for fake news detection. Our best-performing model reaches prediction accuracy of over 99%. We understand that our estimate for prediction performance might be limited to the scope of our data sources, and we welcome suggestions and applications built upon our models.

1 Motivation

In the current era of information overflow, identifying information validity becomes a challenging task for news outlet, social media, and more importantly, the general public. This process becomes increasingly more relevant when platforms or individuals voice opinions of personal belief, and are not as much concerned about the truthfulness these information. We see this phenomenon intensified in the midst of political turmoil. Therefore, we believe it's important to build a scalable model to preemptively identify fake news from more legitimate ones, the task that would otherwise be too inefficient to go through with human scrutiny.

We believe the application of text classification model can help both traditional news outlet as well as social media to systematically flag unreliable information. In addition to fact checking, we believe similar workflow can be used in other text classification tasks. For example, the classification of true and fake news could assist finance/investment companies in understanding sentiment in news and market reports sentiment and consequently contribute to more accurate market forecasts. In the future, this could also potentially be applied in the legal domain.

2 Dataset

2.1 Dataset Introduction

With this goal in mind, we have located a dataset on Kaggle titled, **Fake and real news dataset**.[1] Formally, the dataset is called "ISOT Fake News Dataset".[2] It consists of over 40,000 news articles that are manually annotated as either real or fake. This is a fairly balanced dataset, as around 50% of the news

[1]Kaggle dataset source, https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset.

[2]Kaggle dataset metadata - sources, https://www.uvic.ca/engineering/ece/isot/assets/docs/ISOT_Fake_News_Dataset_ReadMe.pdf.

article are classified as fake news. This dataset was collected from real-word sources dated 2015 to 2017 - the real news had been obtained by crawling Reuters.com, whereas the fake news are collected from unreliable sources as flagged by PolitiFact and Wikipedia.[1].

News	Number of Articles
"true" news	21,417
"fake" news	23,481

Table 1: Data Summary

In addition to the labels, the provided features includes:

1. **title**: Title of the article.
2. **article**: The full article itself, including punctuation.
3. **subject**: For fake news, the subjects include "GovernmentNews", "Middle-east", "US News", "left-news", "politics", "News". For true news, the subjects include "World-News" and "PoliticsNews".
4. **date**: The publication date of news.

2.2 Variables Exploration

2.2.1 Labels - Target variable

We believe it is crucial to understand how labels are created - this would allow us to better interpret our models and their applicability to new data.

As mentioned above, "true" articles are all from reuters.com, while "Fake" articles comes from PolitiFact and Wikipedia. According to PolitiFact, they consider a set of questions when deciding **which news to review.**[3]

PolitiFact also listed **how the team determines their ratings**. As opposed to binary "true" and "false", the team actually has six levels of ratings, ranging from "true", "mostly true", "half true", "mostly false", "false", and "pants on fire". Kaggle turned this into binary categories, and we have not yet to find resources that can validate Kaggle's methodology. The editors and reporters at PolitiFact indicatest that they rate reviewed articles by discussing a few questions [4]. This is important as the data collection process is related to how good our model results are.

Figure 1a shows some excerpt of the examples of "true" news (each paragraph is a separate piece of news). Note that the latter example centers around Tweets, which is not the norm among "true" news. As specified in data collection method, true news are scraped from Reuters news and as a result, the texts contain its source tag, **"(Reuters)"** at the beginning of the articles. This piece of identifying text will be removed in the proceeding analysis to ensure robustness of model when applied to different information sources.

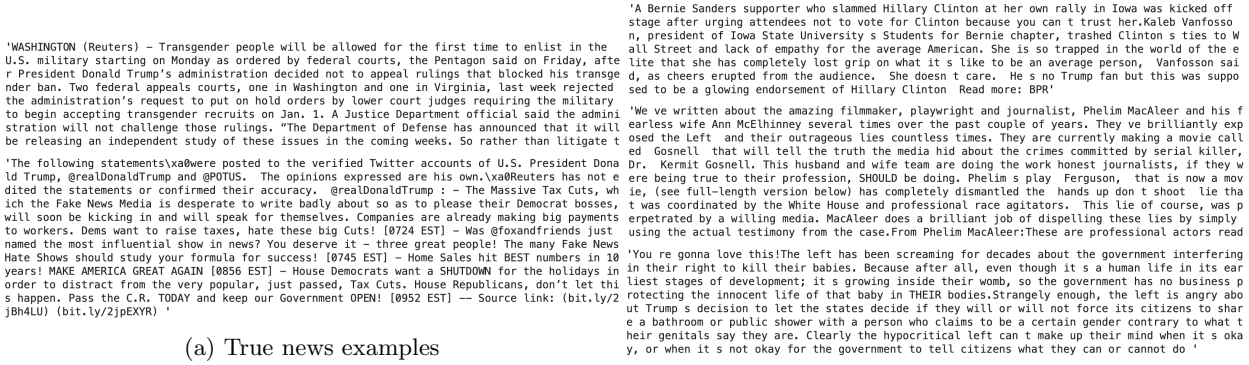
Figure 1b illustrates some excerpt of the examples for fake news (each paragraph is a separate piece of news).

As shown in the examples below, the content of the fake news spans from political news with strong partisan sentiment to rather non-sensible passages. We observe that fake news seem to be more conversational. In

[3] (1) Is the statement rooted in a fact that is verifiable?(the team does not check opinions) , (2) Does the statement seem misleading or sound wrong? (3) Is the statement significant? (4) Is the statement likely to be passed on and repeated by others? (5) Would a typical person hear or read the statement and wonder: Is that true?

[4] (1) Is the statement literally true? (2) Is there another way to read the statement? (3) Is the statement open to interpretation? (4) Did the speaker provide evidence? Did the speaker prove the statement to be true? (5) How have we (PolitiFact's) handled similar statements in the past? What is PolitiFact's jurisprudence?

addition, as suggested by the "subject" variable in the original dataset, the range of the topics is broader for fake news. In our main analysis, we did not include subject as a feature nor as a filter. As a sensitivity analysis, we attempt to include only domestic politics related news, to create models that are not as affected by the potential systematic different in input subject.



'WASHINGTON (Reuters) - Transgender people will be allowed for the first time to enlist in the U.S. military starting on Monday as ordered by federal courts, the Pentagon said on Friday, after President Donald Trump's administration decided not to appeal rulings that blocked his transgender ban. Two federal appeals courts, one in Washington and one in Virginia, last week rejected the administration's request to put on hold orders by lower court judges requiring the military to begin accepting transgender recruits on Jan. 1. A Justice Department official said the administration will not challenge those rulings. "The Department of Defense has announced that it will be releasing an independent study of these issues in the coming weeks. So rather than litigate t

'The following statements\xa0were posted to the verified Twitter accounts of U.S. President Donald Trump, @realDonaldTrump and @POTUS. The opinions expressed are his own.\xa0Reuters has not edited the statements or confirmed their accuracy. @realDonaldTrump : - The Massive Tax Cuts, which the Fake News Media is desperate to write badly about so as to please their Democrat bosses, will soon be kicking in and will speak for themselves. Companies are already making big payments to workers. Dems want to raise taxes, hate these big Cuts! [0724 EST] - Was @foxandfriends just named the most influential show in news? You deserve it - three great people! The many Fake News Hate Shows should study your formula for success! [0745 EST] - Home Sales hit BEST numbers in 10 years! MAKE AMERICA GREAT AGAIN [0856 EST] - House Democrats want a SHUTDOWN for the holidays in order to distract from the very popular, just passed, Tax Cuts. House Republicans, don't let this happen. Pass the C.R. TODAY and keep our Government OPEN! [0952 EST] -- Source link: (bit.ly/2jBh4LU) (bit.ly/2jpEXYR) '

(a) True news examples

'A Bernie Sanders supporter who slammed Hillary Clinton at her own rally in Iowa was kicked off stage after urging attendees not to vote for Clinton because you can t trust her.Kaleb Vanfosson, president of Iowa State University s Students for Bernie chapter, trashed Clinton s ties to Wall Street and lack of empathy for the average American. She is so trapped in the world of the elite that she has completely lost grip on what it s like to be an average person, Vanfosson said, as cheers erupted from the audience. She doesn t care. He s no Trump fan but this was supposed to be a glowing endorsement of Hillary Clinton Read more: BPR'

'We ve written about the amazing filmmaker, playwright and journalist, Phelim MacAleer and his fearless wife Ann McElhinney several times over the past couple of years. They ve brilliantly exposed the Left and their outrageous lies countless times. They are currently making a movie called Gosnell that will tell the truth the media hid about the crimes committed by serial killer, Dr. Kermit Gosnell. This husband and wife team are doing the work honest journalists, if they were being true to their profession, SHOULD be doing. Phelim s play Ferguson, that is now a movie, (see full-length version below) has completely dismantled the hands up don t shoot lie that was coordinated by the White House and professional race agitators. This lie of course, was perpetrated by a willing media. MacAleer does a brilliant job of dispelling these lies by simply using the actual testimony from the case.From Phelim MacAleer:These are professional actors read

'You re gonna love this!The left has been screaming for decades about the government interfering in their right to kill their babies. Because after all, even though it s a human life in its earliest stages of development; it s growing inside their womb, so the government has no business protecting the innocent life of that baby in THEIR bodies.Strangely enough, the left is angry about Trump s decision to let the states decide if they will or will not force its citizens to share a bathroom or public shower with a person who claims to be a certain gender contrary to what their genitals say they are. Clearly the hypocritical left can t make up their mind when it s okay, or when it s not okay for the government to tell citizens what they can or cannot do '

(b) Fake news examples

Figure 1: News Text Example

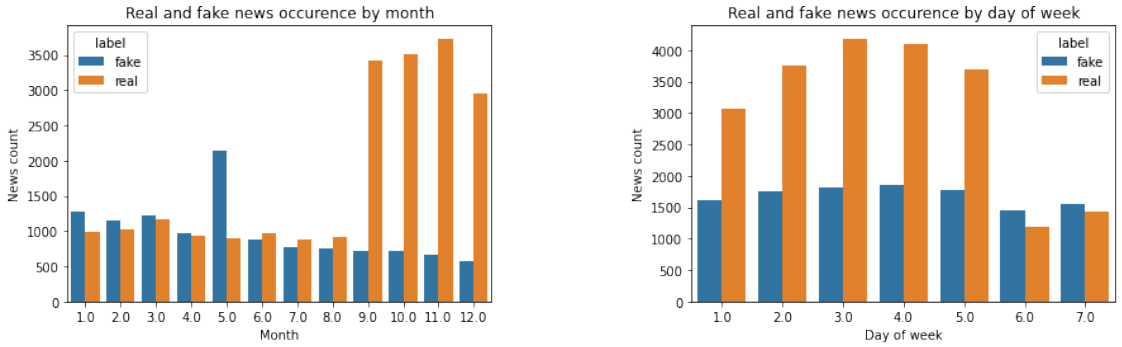
2.2.2 Features/predictors

We have explored the labels in the previous section, and we will now provide an overview of features provided by the dataset, namely title, text, news subject and publication date. [5]

- **Publication date**
  Given publication date for each news article, we first saw that all news were published during 2015 to 2017. Although there seems to more real news collected in 2017, we don't think there exists reasonable explanation to this phenomenon except for data collection method. And we are not going to discriminate between articles published in different years.

  Furthermore, we found some interesting pattern in the month and the day of week when the news is published. **Figure 2a** shows that there are a lot more real news publication during the last four months of the year. We should be cautious of this trend since we only have three years' data. This again could be sensitive to how researchers collected data. We also found that real news are more likely to be published during weekdays. We suspect this might correspond to working schedules at more formalized publication companies.

[5]There are 626 fake news observations and 1 true news entry with empty text fields. We have removed these incomplete observations prior to generating the following graphs and statistics.



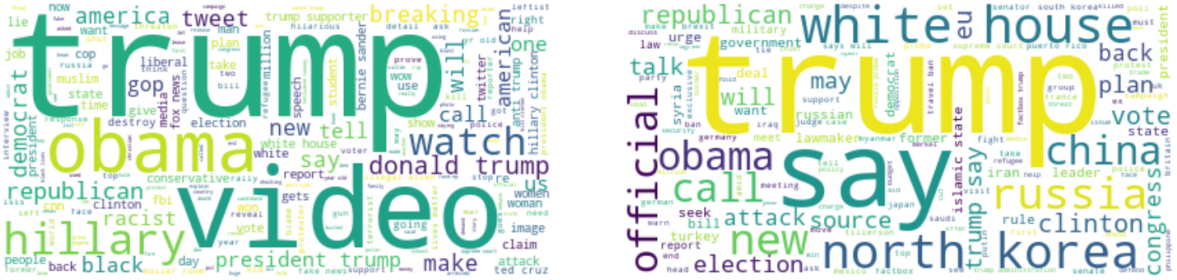
(a) News publication - Month

(b) News publication - Day of week

Figure 2: News occurrences and publication time

- **Title**
  To visualize the vocabulary occurrences in fake and real news titles, we created the following word clouds graphs separately for "true" and fake news title[6] - not surprising how "Trump" is highly reoccurring in both "fake" and "true" news.

  Given our inspection of news title, there doesn't seem to much interesting pattern except both "fake" titles and "true" titles seems highly political. From this analysis however, We did notice that only fake news titles tend to include an indicator of whether the news content includes video by adding **(VIDEO)** at the end of sentence. This results in a dominant occurrence of the word "VIDEO". We think this could be a strong but undesirable indicator of fake news since it is highly related to data collection process (Reuters does not have VIDEO typically in the headline). Therefore, we removed the trailing (VIDEO) in fake news titles.



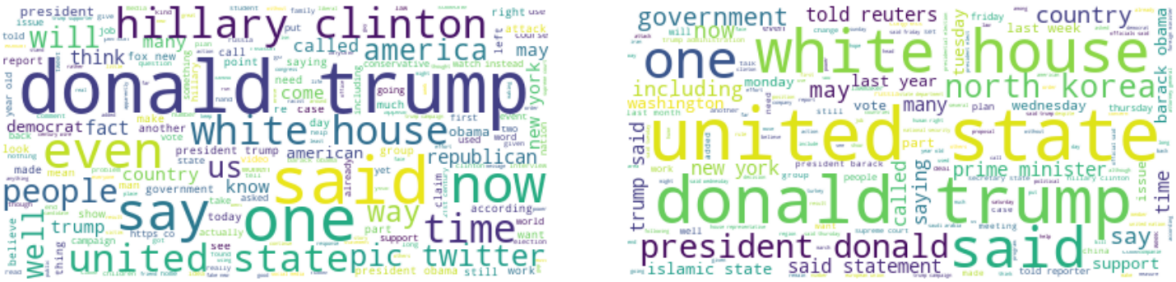
(a) Fake news

(b) True news

Figure 3: New title - word cloud

- **Text**
  We have shown some examples of news text in the previous section, see Figure 1a and Figure 1b. We demonstrate here the most frequent words/phrases in fake and real news texts.

[6]Words/phrases were converted to lowercase prior to generating graphs. This applies to the title wordcloud graphs as well.



(a) Fake news

(b) True news

Figure 4: News text - word cloud

From the 4 figures above, we see that phrases such as "donald trump", "white house", "united states", and other country names are frequent in both fake and real news contents. This is to be expected since our data was collected from a pool of more politics-centered news between 2015 and 2017 for the "true" articles, and PolitiFact are more likely to review politics related news. At the same time, there are words such as "twitter" and "breaking" that are more prominent in fake news, while "official" and "government" are highly used in real news. We expect our text feature extraction mechanisms to pick up these differences.

We also investigated if the percentage of unique words used for fake news is less than true news. The distribution after removing the empty articles looks like below. Interestingly, the distribution for true news and fake news are mainly overlapping, with the true news shifted more to the right. Both distributions centered around 0.55-0.60.

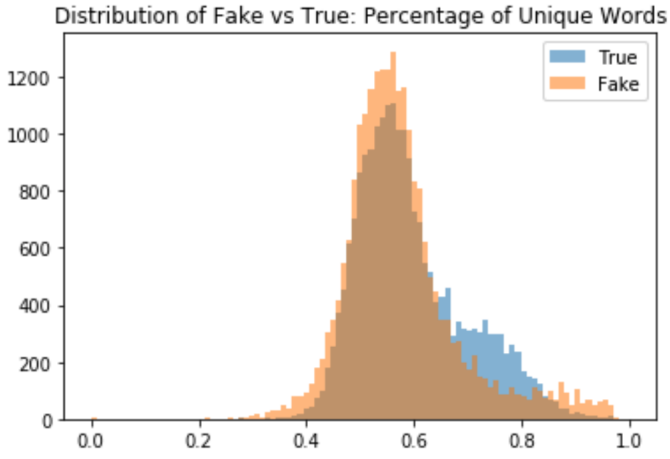


Figure 5: Percentage of unique words for true and fake news text

We think it's not exactly challenging for an educated adult to tell apart most real and fake news based on our training datasets, since fake news tend to have more inciting sentiments and less rigorous sentences. The question remains whether machine learning models can learn to do the same thing by studying term frequencies and sentence embeddings.

- **Subjects**
  Finally, we examined the news subjects reported in the dataset. We noticed that different categorization rules were applied to fake and real news, as illustrated in Figure 6.

News	9050
politics	6435
left-news	4310
Government News	1499
US_News	783
Middle-east	778

(a) Fake news subjects and occurrence counts

politicsNews	11271
worldnews	10145

(b) True news subjects and occurrence counts

Figure 6: News subject comparison

We understand these categorizations are not exactly comparable or informative, and the categories are not even mutually exclusive among the fake news. However, by investigation, we did notice that the contents of *non-politics* fake and real news do vary quite a lot, and we suspect that our model may learn to differentiate between real and fake news very well using the seemly non-ideal categories. This may be useful in some cases, but it would be also desirable to design a classifier that performs well to general news in the world, which may or may not have the exact same labels. As demonstrated above in the label section, these subjects are different due to the different data collection processes for fake news and true news. Thus, we decided not to include subjects as a feature, but to experiment with subsets of data corresponding to politics subjects.[7]

3 Related Work

3.1 General Research and Academic Articles on Fake News Detection

One major feature engineering problem of the project is to sensibly translate text information to numeric representation and use this information to predict the validity of a news article. This task falls under the widely discussed area of text classification. A variety of methodologies and applications have been proposed by previous research, and we found the following sources especially relevant to our project.

P'erez-Rosas and Kleinberadfa (2018) introduced a few linguistic features, including Ngrams, Punctuation, LIWC, Readability, and Syntax in their paper Automatic Detection of Fake News. The authors incorporated a combination of these features.[6] For model selection, the authors used a linear SVM classifier with a cross validation method for six topic domains. The paper used accuracy, precision, recall, and F-score as evaluation metrics. It also calculated accuracy by feature types and found the best performing classifiers are the ones "that rely on stylistic features", such as punctuation and readability. In addition, it is interesting that the paper tested domain-specific news with different types of features. For technology, features that represent readability are the most important predictors. This paper took a more linguistic approach. It is very comprehensive in evaluating different types of features, while its limitations lies in the selection of a single model: linear SVM. This is understandable as the paper has a slightly different research purpose.

O'Brien et. al. (2018) explored the use of CNN for fake news detection in another paper The Language of Fake News: Opening the Black-Box of Deep Learning Based Detectors.[4] This research tested the transferability of their models and demonstrated the potential of applying deep learning classification model to novel topics. Although we do consider this an interesting approach, we would not train CNN, and we also do not attempt to use our model as a transfer learning technique here due to resource and time constraints. Additionally, this may not be applicable in our analysis due to data size. We will focus on our chosen dataset and in terms of deep learning application, use a fully-connected neural network model instead of CNN.

[7] We deem fake news of subject *politics* and *Government News*, and real news of subject *politicsNews* to be politics relevant.

3.2 Prior Work Using the Same Dataset

The most relevant paper (to our problem) is by Ahmed H, Traore I, and Saad S.(2018), who utilized the same dataset source in their paper, Detecting Opinion Spams and Fake News Using Text Classification[1]. The paper achieved a 98% accuracy initially using "a combination of news articles from different years with a broad variety of political topics." This accuracy would be a reference point for our project, specifically the topic-restricted version.

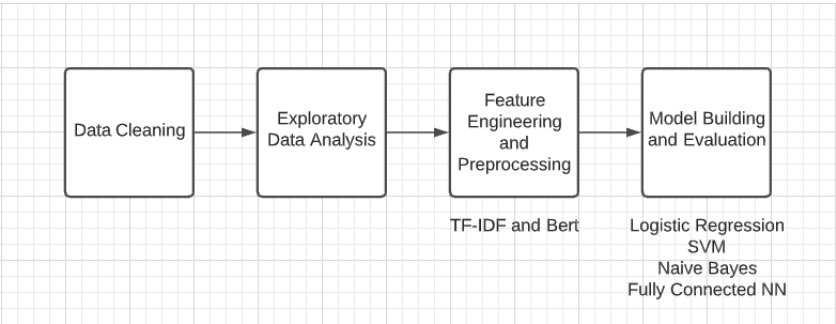
The paper decided to only examine a subset of articles around the 2016 US election due to high accuracy for simple models, which could capture structural differences in the data inputs. The authors picked 2000 real articles and 2000 fake articles. However, there are still two limitations. One is that there could be some biases when the authors choosing which data points to use. Secondly, even though the authors picked 2000 real articles, the real articles still came from a single data source: Reuters. Fortunately, the authors' accuracies seems more realistic.

For word representations, the authors used TF (Term Frequency), and TF-IDF (Term Frequency - Inverse Document Frequency) with a combination of different feature size and n-grams, the paper presented results for SVM, LSVM (Linear Support Vector Machine), Logistic Regression, KNN, and Decision Trees. The best model is LSVM, using TF-IDF (as opposed to TF) and Bigram. The use of n-gram tried to consider the context of the word representation, while Bert (which is not available yet at the time of this paper) could certainly do a better job.

Due to time limitations, we will not hand select 2000 articles. In addition, we explore a slightly different combinations of these models, and additionally use Bert as a context-based embedding to compare with context-free TF-IDF. At the time of this paper, which uses TF-IDF with n-gram, Bert has not been introduced yet.

A different paper by Liqiang Wang also used PolitiFact news[7]. The paper created various features for classification model training, and used "All Words" and "Headline" as the final features to feed in to deep learning followed by a logistic regression. Our neural network model resembles this structure.

4 Problem Formulation



We can solve fake news detection as a supervised learning problem. We would like to train a binary classifier that can accurately identify a news article as "fake" or "true", given its title, text, and/or other properties associated with its publication, such as publication date information. The graph provided above summarized our process. At the end we will also discuss how our input data affected model performance.

The uniqueness of this classification problem lies in the nature of our input data - natural languages. We understand that for machines to understand these text information, we would need to adequately convert text into numeric representation in the feature engineering process. In designing this transformation mechanism, it's important to compare it to how human make the conscientious choice of marking a news article as illegitimate. For instance, we may consider word choices and/or the context of the words. At a high level, we think that there are discernible differences between real and fake news characteristics; in other words,

we believe that real news and fake news are similar to those in their own groups on certain aspects. This property should remain in the numeric representations - similar news title and articles should also be in close proximity to each other in the newly found vector spaces. We will experiment with both context-agnostic (TF-IDF) and context-sensitive (BERT) text feature extraction methodologies. It should be noted that our main objective is not to develop new or improve current natural language processors, but rather employ existing NLP techniques for our ultimate goal of classification. More details will be mentioned in the Methods section.

We'll discuss below the specific methods we use regarding (1) text feature extraction and (2) binary classification.

5 Methods

5.1 Feature Engineering

1. **TF-IDF vectorization with Principal Component Analysis**
   TF-IDF stands for Term frequency - Inverse document frequency. It numerically encodes a word in a document, and its value represents how important the word is to one document in a dataset. Term frequency measures how frequent a word is in a document. But term frequency might not be representative of the document, since some words appear systematically more frequently than others, thus we account for this overall frequency by multiplying TF by inverse document frequency (IDF). IDF accounts for how many documents in the corpus has this word. Incorporating the IDF factor essentially decreases the weight of vocabulary that occurs frequently across the dataset, and increases weight of rare ones. As a result, vocabulary exclusive to certain document would have a high value, since it's representative of that document.

   Given this property of TF-IDF statistics, we think it would be appropriate to encode news title and news text separately. In addition, since we expect there to be numerous distinct words in our news dataset, and given our limited computation power, we are going to limit the number of features, i.e. words, considered.

   Applying the constraint mentioned above, we would still have more than a few thousands features, and to reduce the risk of overfitting, we are going to further reduce the number of features with principal component analysis. The exact number of components will be determined with the 'elbow' method.

2. **BERT/DistilBERT**
   Bert stands for Bidirectional Encoder Representation from Transformers, which is introduced in 2018 and trained through bidirectional transformers. One could fine-tune the model parameters, or just use it as pre-trained feature generator. In addition, Bert can take punctuation, which could be important in the classification of fake vs. true news.

   In this project, we use pre-trained DistilBERT[5], which is designed to be "smaller, faster, cheaper and lighter". This model effectively combined the step of PCA on Bert. This model works well with our dataset, given article length and sample size. The specific pre-trained model we used is "distilbert-base-uncased". This model accepts punctuation but would require lower case letters. Finally, the max number of words + punctuation is 512 (so some articles are cut off), and the dimension of embedding is 768.

   We created distilBERT embeddings for two sets of features: 1) article titles, and 2) article text/bodies. There are two steps in getting the embeddings: 1) tokenize the cleaned text input and make them encoded inputs, and 2) obtain the embeddings from the model. Since it is inefficient to repeatedly run BERT to obtain word embeddings, we saved the outputs of the distilBert embedding in numpy formats for further use as inputs in models.

5.2 Modeling

Given that we have two distinct ways of encoding text information, we will perform the following machine learning models on two sets of features separately, and compare the performances of different models as well as between different sets of features. As mentioned earlier, after obtaining the numeric text features, our question becomes a relatively straightforward classification. We'll briefly discuss the models we intend to implement. The results will be presented in the next section.

1. **Naive Bayes (Baseline)** is commonly used in text classification, especially in spam classification. Compared to other models, Naive Bayes is extremely fast in training and prediction, and it requires no additional parameter tuning. We use Multinomial Naive Bayes with TF-IDF as it is typically used with frequency features, which makes it suitable for text classification with TF-IDF representations[8]. Since Multinomial Naive Bayes does not accept negative values, for Bert we use Gaussian Naive Bayes for a general classification task.

2. **Logistic Regression** is a simple yet powerful classification tool. The intuition behind Logistic regression is that it separates observations with a linear hyperplane in the feature space. So if our selected features can satisfy the linearity assumption, then logistic regression would yield great performance. We use "elastic net" and "L2" regularization methods.

3. **Linear SVM** is also commonly used in text classification. This is the best model for Ahmend H's 2018 paper [1]. It is an interesting alternative to logistic regression, since both attempts to draw linear hyperplane to separate two classes. Given this characteristic, we might expect similar performance from linear regression and linear SVM.

4. **Random Forest** is an ensemble tree model that does not subject to linearity assumption. Since tree models make hierarchical assumption on the features, it would be interesting to compare the result with the previous models. Notably, we understand that it is more sensible for decision trees to make splits on features with realistic significance, rather than embeddings. Therefore we would be using the result from TF-IDF vectorization directly, we expect to perform searches on hyperparameters to avoid overfitting.

5. **Neural Network** is suitable for our data size and number of features. We use binary cross entropy as loss function, tanh as activation layers, batch normalization, dropout to prevent overcasting, and a sigmoid function to translate the last layer into probabilities. When making predictions, we predict true (1) if the probability is greater than 0.5, and predict fake (0) if the probability is less or equal to 0.5. We also perform cross validations to assess overfitting, and use the test accuracy as model evaluation.

5.3 Implementation Approach

During the stage of feature extraction, we used TF-IDF vectorizer and PCA packages from scikit-learn. For BERT, we used "distillbert-base-uncased" model. We obtained this from package transformers' DistillBert-Model.

As for modeling (Multinomial and Gaussian) Naive Bayes, Logistic Regression, and Linear SVM, we also employ packages from sklearn. For logistic regression, sklearn supports different penalty term and allows us to choose regularization strength. The default LogisticRegression() has L2 regularization with C=1 built-in. For TF-IDF models we experimented with elastic net regularization (since TF-IDF is expected to have slightly more multicolinearity among features). For our BERT models we used L2 regularization. Hyperparameter tunings are through cross validation, where we used cross_validation_score package from sklearn.

To build neural network, we created a class "Dataset" for the dataloader in order to train models through batch. We used PyTorch to create the actual model, and used BCELoss (binary cross entropy) as the loss function.

[8] 1.9.2, Multinomial Native Bayes https://scikit-learn.org/stable/modules/naive$_b ayes.html$

6 Experiments and Results

6.1 TF-IDF Results

Before jumping into the discussion of model results, we'll touch upon our preprocessing procedure. We understand that TF-IDF encodes each vocabulary and without restriction, it would produce a numeric value for each unique word in the corpus. We expected there to be a diverse set of words in our news dataset, and possibly some typos - including some words with lower frequency and low significance. Therefore, we limit the number of features, i.e. word, to consider in TF-IDF vectorizer for news title to 1,000, and that for news text to 5,000. This results in a feature space dimension for 6,002 (6,000 text features and 2 date features). We then proceed to perform feature reduction with Principal Component Analysis.[9] To choose an appropriate number of components to keep, we graphed the number of variance explained and the number of components included, shown in **Figure 7**. By inspection, we decided to use the first 750 components, which captures about 99% of variance in the original feature space.

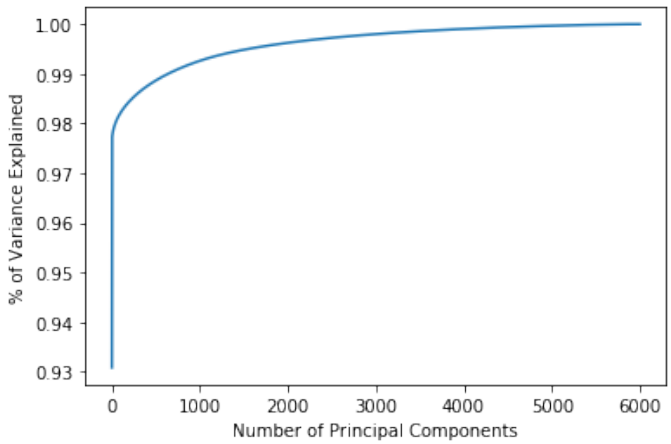


Figure 7: Principal component analysis - variance explained

Moving on to model performance. We first evaluate the performance of **Naive Bayes**. It should be noted that since Naive Bayes assumes strong conditional independence between features and doesn't allow negative values, we will use the original pre-PCA features as predictors. The test accuracy is **94.6%**. It's interesting that this performance is much better than previous work done on the same dataset.[2] We suspect the choice in number of vocabularies to include plays a part in model performance.

The performance of **Logistic Regression** is in par with previous documentation, although still exceeds our expectation. To avoid overfitting, we used "elastic net" regularization with equal weights on L1 and L2 penalties. We experimented with a range of regularization strength, and compared the performances based on validation accuracy. Using the best parameter found, we arrived at a test accuracy of **98.2%** - better performance than Naive Bayes. In terms of computation efficiency, logistic regression takes notably more time than Naive Bayes to tune hyperparameters, train, and converge, which is expected given the size of data and feature space.

Next model we trained was **Support Vector Machine**, specifically we used linear kernel in this case. We observe that training was rather fast for linear SVM; searches for the best regularization parameter, C, was quite efficient as a result. We chose the best C given validation accuracy shown in **Figure 8**, although the distinction was not very significant. The resulting test accuracy was 98.2%, which is comparable to the logistic regression performance.

[9]We first split the data between training and test set, then we found the singular vectors of the training set matrix, and then transformed both training and test data with the chosen dimension of singular vector matrix.

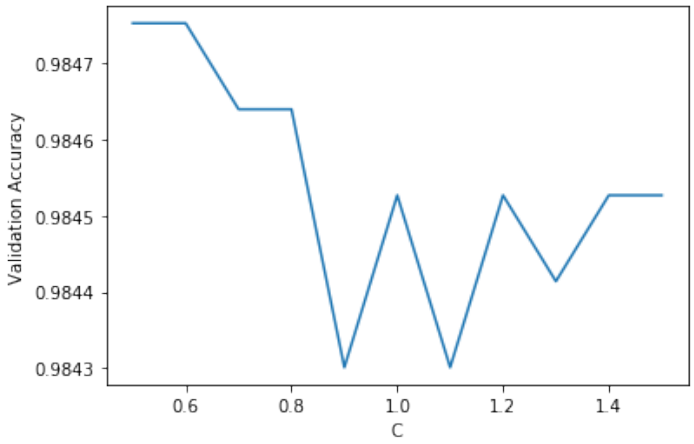


Figure 8: SVM - regularization parameter search

Additionally, we ran **Random Forest Classifier** on the original TF-IDF, with each word being a feature.[10] We understand that single decision trees are prone to overfitting and tree ensembles, such as random forest, is designed to resolve this issue. Initially, we achieved perfect training accuracy with a slightly lower test accuracy. With appropriate regularization parameter, we were to able to achieve test accuracy of **98.9%**. We show below in **Table 2** an excerpt of our hyperparameter search process.[11] We noticed something interesting in tuning random forest - the gap between training and validation accuracy doesn't monotonically decrease as we increase regular strength - on the contrary, it tends to bounce around, and sometimes stay the same. We suspect this could be related both to the randomness in ensemble tree models as well as how trees are constructed.

Max Depth	Training Accuracy	Validation Accuracy
30	99.9%	98.8%
35	99.9%	98.7%
40	99.9%	98.9%
45	99.9%	98.7%

Table 2: Random Forest Hyperparameter search example

The list of model performances can be found in **Table 3**. We found that using TF-IDF vectorization, along with dimensionality reduction technique when necessary, we were able to achieve test accuracy of 98.9% with random forest.

Model	Test Accuracy
Naive Bayes	94.6%
Logistic Regression	98.2%
Linear SVM	98.2%
Random Forest	98.9%

Table 3: Model performance with TF-IDF features

[10]We also tried running random forest with PCA transformed features. The test accuracy trails that of model with word features by more than 2%.

[11]The table only shows a small subset of our hyperparameter search. For other combinations of hyperparameter, we at best reduced the difference between training and validation accuracy to about 1%. We chose the hyperparameter combination that produced the highest validation accuracy.

6.2 DistilBERT Results

DistilBERT creates word embeddings for a max of 512 words and punctuation for each article. Padding is added in case there are articles that are shorter than 512 words. Each word has 768 dimensions of representation, pretrained from the model. The first dimension of the BERT output for each article is the embedding on the article level (aka the hidden states). Our final output from BERT are two sets of features for title and text, each of dimension (n = 44,271[12], p = 768).

For this part of the work, title and text are first separately used as inputs for the models as we are interested in seeing their individual effects using BERT as a embedding. Finally, we also trained a version of the models with both Title and Text (n = 44,271, p = 768*2). Our results for distilBert are below:

	Test Accuracy		
Model	**Title**	**Text**	**Title + Text**
Naive Bayes	85.0%	92.4%	92.2%
Logistic Regression	96.8%	99.1%	99.6%
Liner SVM	96.8%	99.0%	99.6%
Neural Network	96.1%	98.8%	99.5%

Table 4: Model performance with distilBERT features

Similar to the TF-IDF version, **Naive Bayes** model performed the worst among the models for the distilBERT embedding as well. Using Title alone, the test accuracy is only 85.3%. Using text alone, the accuracy increased to 92.7%, and the test accuracy for combined is 92.3%. It seems adding Title to text did not lead to a higher accuracy.

Logistic Regression and **Linear SVM** perform the best among the four models. For LSVM, due to risk of overfitting, we used cross validation to select the best slack variable. For these two simple models, the test accuracy improved when using both Title and Text as inputs. For each of the combinations for these two models, we performed separate cross validations to control for overfitting. As expected, title requires less regularization than text or title-text combination. An example figure is below for the Logistic Regression model:

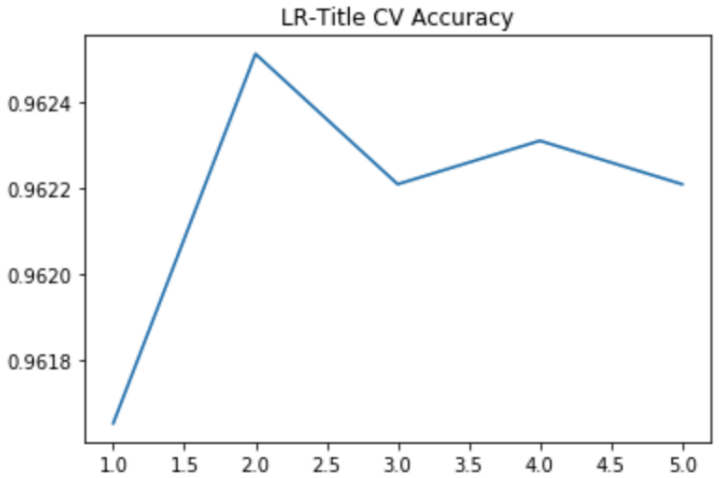


Figure 9: Logistic regression cross validation accuracy example

Neural Network Classifier is ran instead of random forests, as in the case of TF-IDF, because bert

[12]Note that the number 44,271 excludes articles with no body text.

embeddings "separated" the meaning of each word in the article and are aggregated on the article level. In other words, the significance of each bert embedding is not as well defined as TF-IDF features, which makes BERT less suitable for random forest. Due to how well our simple models performed, we decided to use a simple architecture with one hidden layer, setting bias = true. Between the hidden layer and the output layer, we will also add a dropout layer to prevent overfitting. The structure and hyper parameters are detailed below:

Model Structure
Linear Layer 1
Tanh Activation
Hidden Layer
Batch Normalization
Dropout
Linear Layer 2
Sigmoid

Table 5: Neural Network Structure

	Title	**Text**	**Title + Text**
Hidden Neuron	80	300	500
Dropout Rate	0.3	0.5	0.5
Learning Rate	0.3e-3	0.5e-3	0.5e-3
Num Epochs	7	7	7

Table 6: Neural Network Hyper Parameters

We decided to use early stopping and run only 7 epochs because of how the model overfits easily. Partially due to strict early stopping rule, the model performed slightly worse than Logistic Regression and Linear SVM.

6.3 Sensitivity using subset of subjects

As mentioned earlier, our models are performing very well for simple models like logistic regressions and LSVM. We explored how the data input structure could have produced this result. First of all, our "true" news source is Reuters and "fake" news source is from PolitiFact. The news obtained from these two sources are likely to be fundamentally different. An visible example is the "subject" variable and general topics.

In our discussion of feature exploration earlier, we mentioned that the data provides us with a subject variable that we decided not to use as a feature. From the categories of "true" and "fake" news, we noticed that the subject topics for "fake" news include government news, middle-east, US news, left news, politics, and News. This appears to be very different with the those of the "true" news, which is only politics and world news. Therefore, we decided to restrict the topics of the news, so as to reduce structural difference between our "true" and "fake" data. For "fake" news, we kept the articles with subject ,"politics" and "Government News". For "true" news, we kept articles with subject "politicsNews". The class remains quite balanced after selection, which allows us to continue use accuracy score as evaluation metrics, although we also computed the f-scores. Our results are below for both unrestricted (n = 44,271) and politics restricted (n = 19,205) versions.

TF-IDF Results

Model	Test Accuracy For *All* News	Test Accuracy For *Politics* News
Naive Bayes	94.6%	94.8%
Logistic Regression	98.9%	98.7%
Linear SVM	98.2%	98.0%
Random Forest	96.4%	97.7%

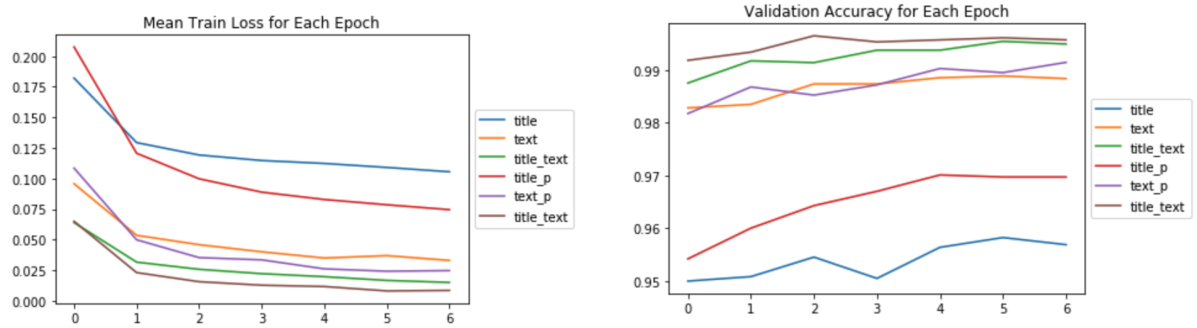
Table 7: Model Performance with TF-IDF features - All news vs Politics news

distilBERT Results

	Test Accuracy For *All* News			Test Accuracy For *Politics* News		
Model	**Title**	**Text**	**Title + Text**	**Title**	**Text**	**Title + Text**
Naive Bayes	85.0%	92.4%	92.0%	87.4%	92.5%	93.7%
Logistic Regression	96.8%	99.1%	99.6%	97.6%	99.3%	99.8%
Liner SVM	96.8%	99.0%	99.6%	97.6%	99.2%	99.8%
Neural Network	96.1%	98.8%	99.5%	97.3%	98.9%	99.6%

Table 8: Model performance with distilBERT features - All news vs Politics news

We performed cross validations similarly to the previous section for the politics features. The graphs will be in the notebooks. It is interesting to note that even after restricting the topic to politics, for BERT, most models' accuracies increased. We interpreted this as the semantic differences between "fake" and "true" news for politics is somehow greater than general topics, due to how emotional people could get over political topics for the "fake" news in particular. The training loss and validation accuracy for all BERT Neural Network Models across the 7 epochs are below:



(a) Mean Training Loss

(b) Validation Accuracy

Figure 10: Neural Network Epoch Results

7 Conclusions and Discussion

Both TF-IDF and Bert are great to represent words, and Bert performs better in general than than TF-IDF. It is great to learn that language models like TF-IDF and pre-trained Bert do a good job to represent articles numerically, and that Bert performed slightly better (as expected) as it included context in the embeddings. If we do not use the words as features but some metadata such as percentage of unique words as a single feature, we would only achieve 0.55% accuracy and 0.64% f-score using a simple logistic regression.

Models that are more "complicated" are not always better. In the TF-IDF version, random forest had the highest performance. Before we perform the model training using BERT as features, we expected

Neural Network to perform better. However, our result shows that it is very easy for Neural Network model to overfit, and we tried to be prudent. We also actually tested RBF SVM for the BERT version, but the results are not as good as the linear SVM. This might suggest that the separating hyperplane in the BERT feature space follows a linear form, which a linear model will fit the best.

Accuracy and f-score works almost equally well for a balanced data. As expected, since both our data restricted on topic "politics" and the full data are fairly balanced, the accuracy and f-score are very close.

The input data collection process likely contributed to the performance of our models, which could be a potential problem. Our main surprise is how well even simple models like logistic regressions and SVM are performing, achieving accuracy of 97-99%. After observing this, we spent some time exploring the reason, and discovered that our input data for true articles and fake articles seem to have very different structures. The true articles are formal articles from a single news source (Reuters), while the fake ones are the news selected to be reviewed by PolitiFact, and typically involves more comments-like sentences with more "extreme" words and emotions (see examples in section 2.2.1). Therefore, even after we restricted topics to only politics, we still got a very high accuracy.

While it is exciting to see how our models can do a great job picking out formal news articles from the rest, it could be more interesting if we had the articles that PolitiFact reviewed but decided to label as true to be the training data for our "true" articles instead. This way, we could potentially create models that can generalize better and save time to classify new articles. In addition, as we learned that PolitiFact originally rated the articles as tiers of "fakeness" as opposed to a binary classification, it would be interesting to use that rating system instead.

To summarize, we do note the **potential problem** in generalizing and evaluating the effectiveness of our models when applying to real world fake news detection, and could do more as next steps. The sources for both fake and real news are quite limited in our dataset, and news subjects are heavily politics centered. While our model could have learned well within this restricted scope of news article, it might not generalize well to additional sources and subjects. Secondly, given that our text feature extraction method has generated a set of embeddings that are hardly interpretable by human, consequently it is a challenging task to understand what information did our model rely on to make decisions. Fortunately, there are methods available that would allow us to explain our model, for instance, LIME[3]. Although we have tried to eliminate tags that are strong indicator of fake or real news, we anticipate that LIME could further augment this process by identifying phrases that were used to, but cannot be generalized, to make predictions. Therefore, in future researches, we expect better data quality, i.e. dataset with more homogeneous sources and more diverse subjects, and employment of techniques such as LIME could push the performance and generalization of our model farther.

Acknowledgments

We appreciate Professor Lyle Ungar and the CIS520 class community's advice and support for this project.

References

[1] Saad S Ahmed H, Traore I. Detecting opinion spams and fake news using text classification. *Journalof Security and Privacy*, 1(1), January/February 2018.

[2] Ishaan Arora. Document feature extraction and classification.

[3] Carlos Guestrin Marco Tulio Ribeiro, Sameer Singh. "why should i trust you?" explaining the predictions of any classifier. February 2016.

[4] N. O'Brien, Sophia Latessa, Georgios Evangelopoulos, and X. Boix. The language of fake news: Opening the black-box of deep learning based detectors. 2018.

[5] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[6] Alexandra Lefevre Rada Mihalcea Veronica Pérez-Rosas, Bennett Kleinberg. Automatic detection of fake news. *COLING*, 2018.

[7] Liqiang Wang, Yafang Wang, Gerard de Melo, and Gerhard Weikum. Understanding archetypes of fake news via fine-grained classification. *Social Network Analysis and Mining*, 9, 07 2019.