# Fake News Detection Using Machine Learning

A PROJECT REPORT

*Submitted by*

Somesh P. Panchal

180410107052

***In fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

Computer

SVIT, VASAD



**Gujarat Technological University, Ahmadabad**

**[May 2022]**

# Acknowledgment

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them. We are highly indebted to all the mentors, i.e. internal mentors, Keyur Suthar, external mentor, Dr. Neha Soni for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project and all the other people that came forward to help us whenever we got stuck at some place. We would like to express our gratitude towards them for their kind cooperation and encouragement which helped us in the completion of this project. Our thanks and appreciations also go to our colleagues in developing the project and people who have willingly helped us out with their abilities.

# Abstract

As more people become reliant on social media for their news, the spread of unreal or fake news has been more prevalent in society than ever before. Using both linguistic analysis and machine learning, we engineered a program that can classify news articles into fake or real. To create a fake news classifier, we had to split the problem into two parts: extracting features and creating a classifier. In my first prototype, I used Countvectorizer to extract features and the Passive-Aggressive Classifier (PAC) and multinomial naive bayes to classify based on the features. In our second prototype, we decided to use a Term Frequency-Inverse Document Frequency (TFIDF) vectorizer instead of countvectorizer to help our program run faster and make it more scalable to larger datasets. In our refined prototype, I used LSTM classifier to help our program run faster. To test the effectiveness of each prototype, 20 trials were run on each program under different training-testing data ratios. The first prototype achieved an accuracy of 92.317284 percent with an average runtime of 14.4118 seconds under its best conditions. The second prototype reached an accuracy of 91.25967 percent with an average runtime of 7.8138 seconds under its best conditions. The refined prototype reached an accuracy of 91.03943 percent with an average runtime of 4.5 seconds under its best conditions. While the refined prototype is less accurate than the initial prototype, there are significant speed increases. The refined prototype is only 1.1 percent less accurate, but it is also almost 3 percent faster. These findings have significant implications for social media sites because automating the detection of fake news can help inhibit the spread of false articles.

# Table of contents

# 1 Introduction

The problem of unreal or fake news has been around since the creation of the free press. According to John Adams, the second president of the United States of America,"There has been more new error propagated by the press in the last ten years than in an hundred years before 1798." As innovations such as the Internet and social media allowed people to easily connect, it also helped spread news that is misleading or untrue. This term was put into popularity in the 2016 election, when articles spreading over social media had no factual standing. One of the most prominent examples of this is when the article"Pope Francis shocks world, endorses Donald Trump for president," spread over Facebook and had over 900 thousand views. While this article simply wasn't true in any way, it still managed to influence the election. Fake news sites such as the Onion or propaganda made by the government are other examples of modern fake news. Certain social media websites have come under fire because of fake news and how they aren't reacting to the spread of it. Facebooks CEO Mark Zuckerberg has stated that his company plans on utilizing machine learning to help prevent the spread of fake news. However, he wasn't specific on what his approach would be or how effective that approach is. For decades, machine learning has been used to help find spam emails or bots, but detecting fake news requires a much different approach. In this project, we developed a model for detecting fake news that utilizes both text feature extraction and binary classification algorithms. We will also present a database that contains thousands of examples of news articles classified as REAL or FAKE. Using this dataset, we will create a program that will extract linguistic features and then create a classifier that learns from these features to effectively predict the accuracy of news.

# 2 Literature Review

Fake news is a term that refers to news that are false or cannot be verified since it has no reputable facts or quotes. Articles that are often called fake usually have some truth within them, but they fail to give any context. Fake news can be split into seven categories: satire, misleading content, imposter content, fabricated content, false connections, false content, and manipulated context. With technology, the authors of fake news can now use bots to seed articles to gain readers of the article. According to researchers from University of Michigan, fake news spreads much faster than real news. Social media sites such as Facebook and Twitter are common places to spread fake news. These sites use human fact-checkers who are now struggling to keep up. According to a University of Michigan survey, humans could only detect fake news at a 70 percent success rate.

## 2.1 Feature Extraction

TF-IDF is an algorithm that gives weight to how important a word is in a document. TF-IDF first calculates term frequency (number of times word appears in document vs. number of words in document). But because words such as"the" and"or" appear frequently in many documents, TF-IDF devalues words that show up in more documents. This leaves only the relevant words in our text analysis. All of the TF-IDF values will add up to 1. While this feature may sound very simple, it is advantageous and it has a hand in Google searches. By using tfidf, researchers can determine what each article was about and what the keywords of the article are. Below is the formula for tfidf value. tfi,j stands for number of occurrences of i in j. N stands for the total number of documents in the data set. dfi stands for the number of documents containing i.

$$\text{tf-idf}_{i,j} = \text{tf}_{i,j} * \log( N/\text{df}_i )$$

According to an experiment run by Juan Ramos of Rutgers University, TF-IDF was able to find terms more relevant to documents than a naive brute force approach or simply using a bag-of-words model.

**CountVectorizer** tokenizes(tokenization means breaking down a sentence or paragraph or any text into words) the text along with performing very

basic preprocessing like removing the punctuation marks, converting all the words to lowercase, etc. The vocabulary of known words is formed which is also used for encoding unseen text later. An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document. The Bag of Words(BoW) model is a fundamental(and old) way of doing this. The BoW model is very simple as it discards all the information and order of the text and just considers the occurrences of the word; in short it converts a sentence or a paragraph into a bag of words with no meaning. It converts the documents to a fixed-length vector of numbers**.**

## 2.2 Machine Learning

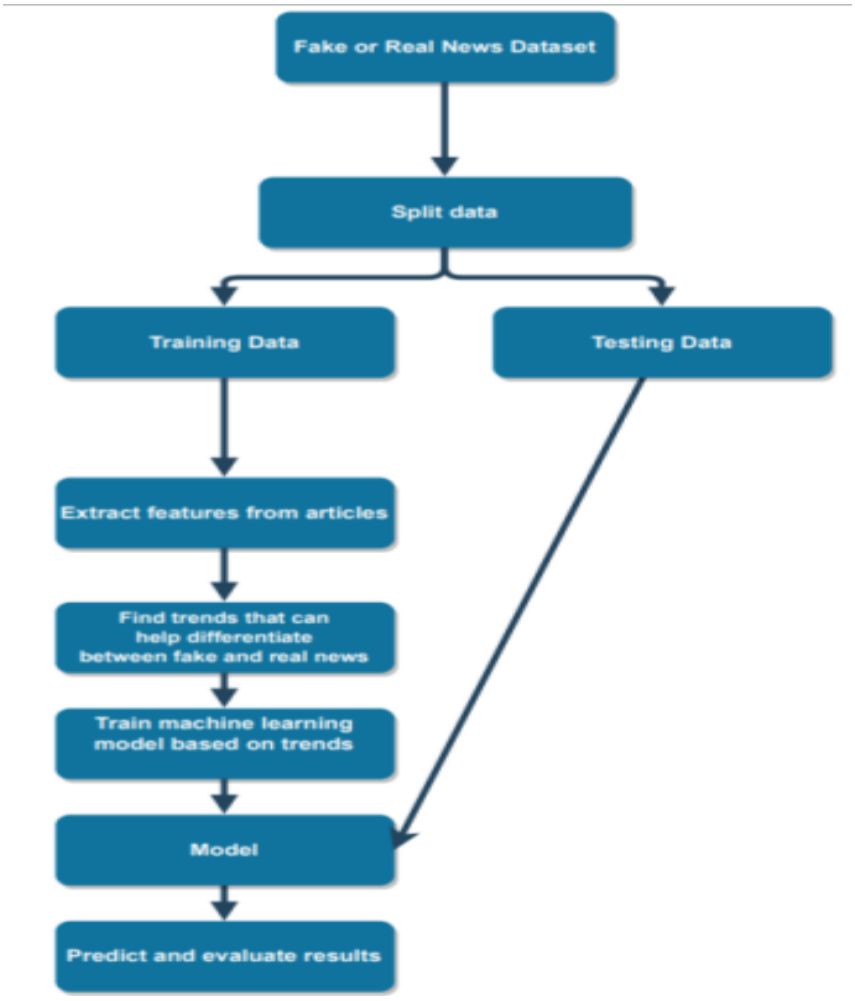
While TF-IDF and hashing simply extracts relevant features from documents, classifying news into fake or real is a much different problem. One way to approach this is using Naive Bayes(a Bayesian Network Algorithm). This classifier assumes that all features are unrelated and it chooses which class a document is most likely to fall in based on how often the feature showed up in the training data in each class. The classifier will then choose which class the document belongs to by looking at the highest probability part. According to an experiment conducted by Carnegie Mellon University, Naive Bayes can reach accuracy rates up to 85 percent. The formula below determines the likelihood whether a document is in a certain class.

$$P(\text{class—data}) = P(\text{data}|\text{class}) * P(\text{class}) \ / \ P(\text{data})$$

Another approach to classifying fake news is to use an on line algorithm, more specifically a Passive Aggressive Classifier (PAC). PAC observes instances in a sequential manner and predicts an outcome (in our case it would be fake or real). Once the algorithm makes a prediction, it receives feedback on how it did. Then, if the mechanism was incorrect, it will modify itself to help make more accurate predictions. It will change the weight vector to match the incorrect prediction as well as the previous data points.

# 3 Goal

The goal of my project is to create a model that can test for what i defined as fake news based on previous examples of fake and real news(in a dataset). This model will essentially learn how to differentiate between fake and real news and make predictions on articles on which the model doesn't know the classification. The model is implemented using a program created in Python. The program should extract a feature set from the text of the articles. Furthermore, the program should also be able to differentiate between fake and real news with an accuracy of 85 percent or more. The program should also be able to run the model in under 15 seconds so that it can be scalable to larger sets of data.




This figure shows the process that the Fake News Classifier will go through

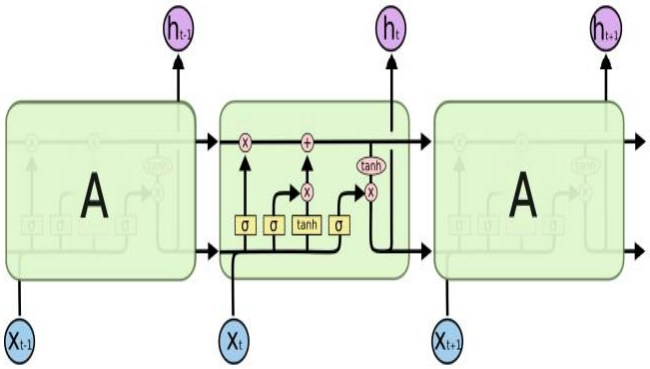
# 4 Used classifiers

**Naïve Bayes:** Naïve Bayes uses probabilistic approaches and are based on Bayes theorem. They deal with probability distribution ofivariables in the dataset and predicting t he response variable of value. They are mostly used for text classification.

Bayes theorem is **P(a|b) =p(b|a)p(b)/p(a)** There are mainly 3 types of naïve base model as - Gaussian Naïve Bayes, Multinomial naïve Bayes and Bernoulli Naïve Bayes. We have used **Multinomial Naïve Bayes** model for our project to detect fake news. An advantage of naïve Bayes classifier is that only requires less training data for classification.

Received Result: If classifier fined news article as fake, then: The true positive are the correctly classified as fake news articles. The false positive examples are the incorrectly classified as fake articles. The true negative are the correctly classified as real news articles. The false negative examples are incorrectly classified as real news articles.

**LSTM Model:** Long Short Term Memory networks – usually just called "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies.




Architecture of LSTM

They work tremendously well on a large variety of problems, and are now widely used. LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

# 5. Overview of Dataset

Dataset is taken from Kaggle platform. It has the following attributes: id: unique id for a news article, title: the title of a news article, author: author of the news article, text: the information of news article. Dataset consist of total 18285 news articles for training and testing of model. Dataset is formed with combination of real and fake news.

**Dataset:** https://www.kaggle.com/c/fake-news/data#

# 6. Data Preprocessing

To transform data into the relevant format the data set needs preprocess. Firstly, we removed all the NAN values from the dataset. Vocabulary size of 5000 words is decided. Then NLTK (Natural Language Processing) Tool Kit is used to remove all the stop words from the dataset. Stop words is list of punctuations + stop words from nltk toolkit i.e. Words such as ‘and’ ‘the’ and ‘I’ that don’t convey much information converting them to lowercase and removing punctuation. For each word in documents if it is not a stop word then that words tag is taken from postag. Then, this collection of words is appended to document.

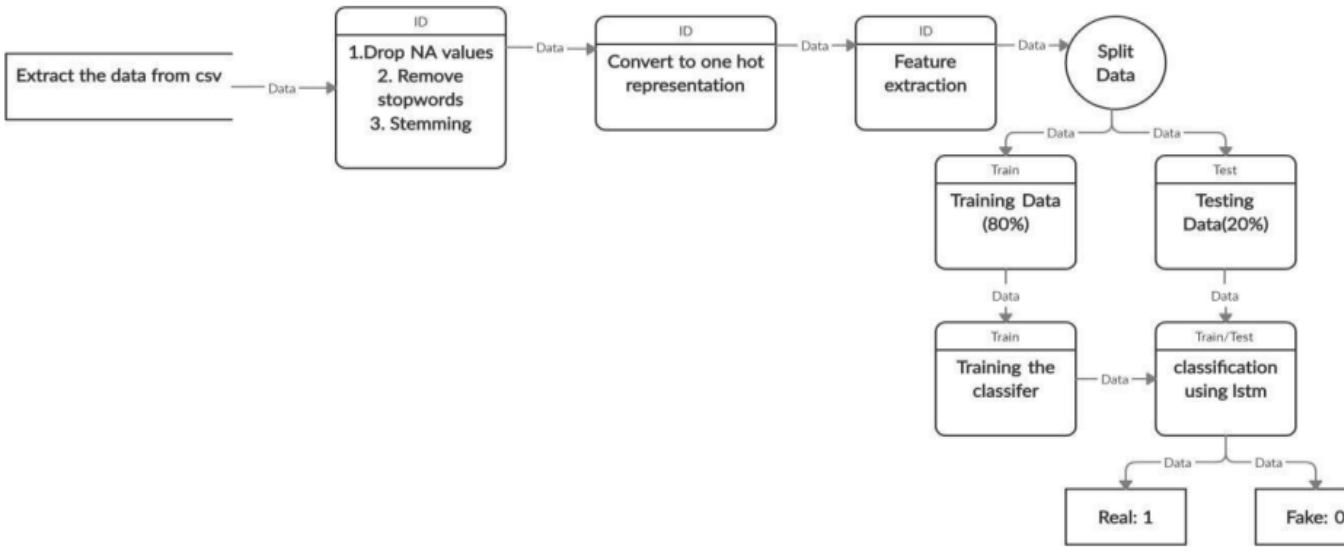
# 6.2 Word Embedding

Onehot Representation: We cannot give input in the form of text format to the algorithm so we have to convert them into the numeric form, for which we are using one hot representation. In onehot representation each word in the dataset will be provided its index from the define vocabulary size and these indexes are replaced in sentence. While giving input to t he word embedding, we have to provide it with the fix length. To convert each sentence into the fix length padding sequences is used. We have considered max length ofi20 words while padding title. Either we can provide padding before the sentence (pre) orafter the sentence (post), or then these sentences pass as input to the word embedding. Word embedding applies feature extraction on the provided input vector. In total 40ivector features are considered.

# 6.3 Model

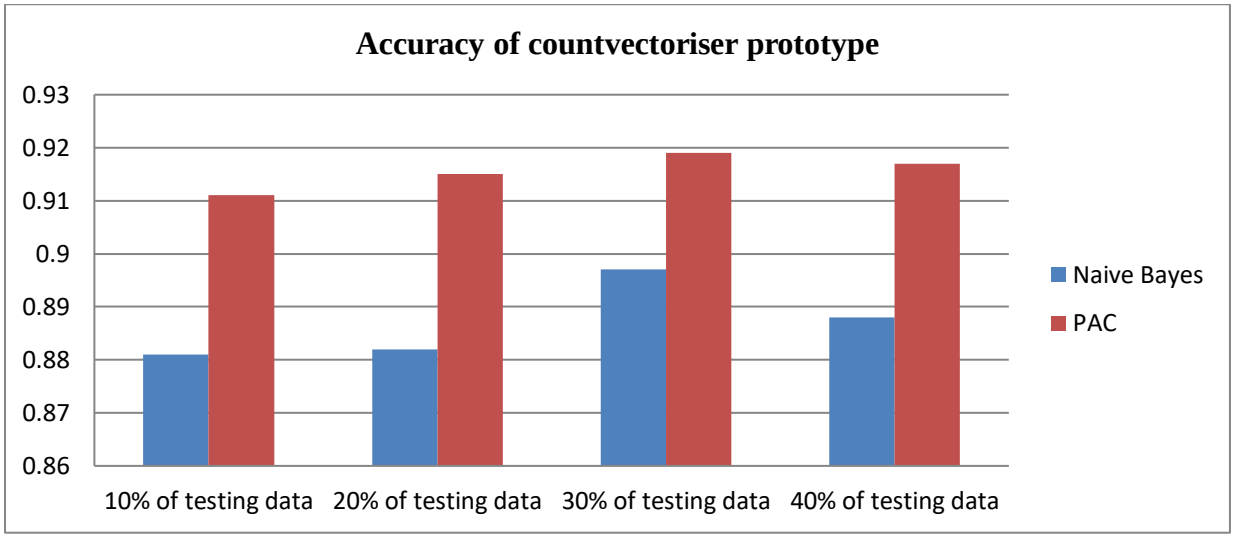
Output from the word embedding is provided to the model. The machine learning model implemented here is a sequential model consisting of embedding as first layer which consist of values vocabulary size, number of features and length of sentence. The next is LSTM with 100 neurons for each layer.

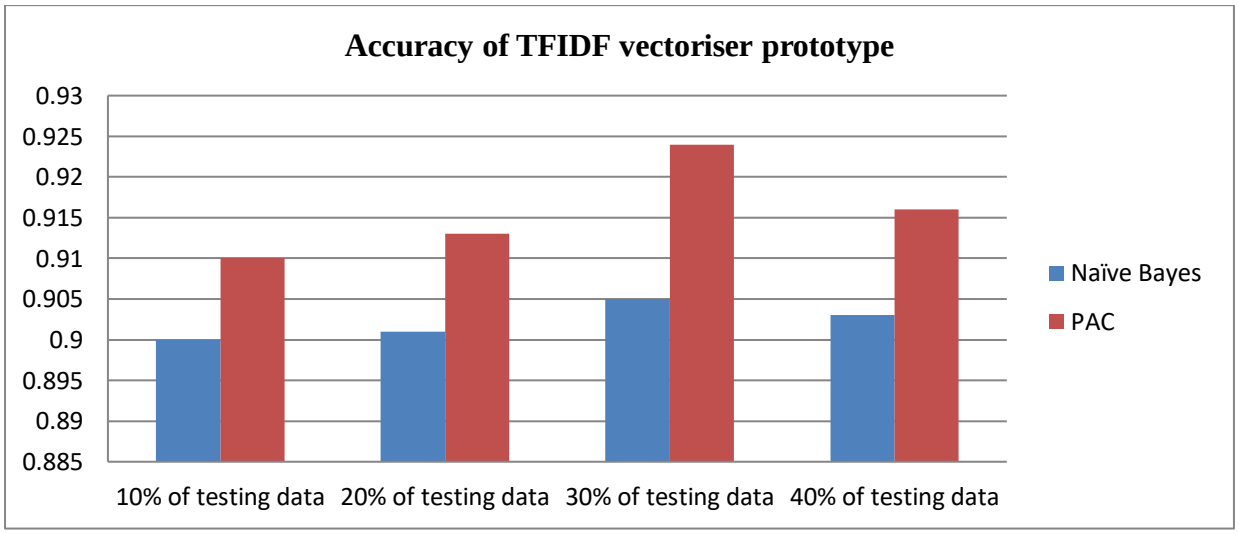
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 20, 40)            200000

 lstm (LSTM)                 (None, 100)               56400

 dense (Dense)               (None, 1)                 101

=================================================================
Total params: 256,501
Trainable params: 256,501
Non-trainable params: 0
_________________________________________________________________
None
```

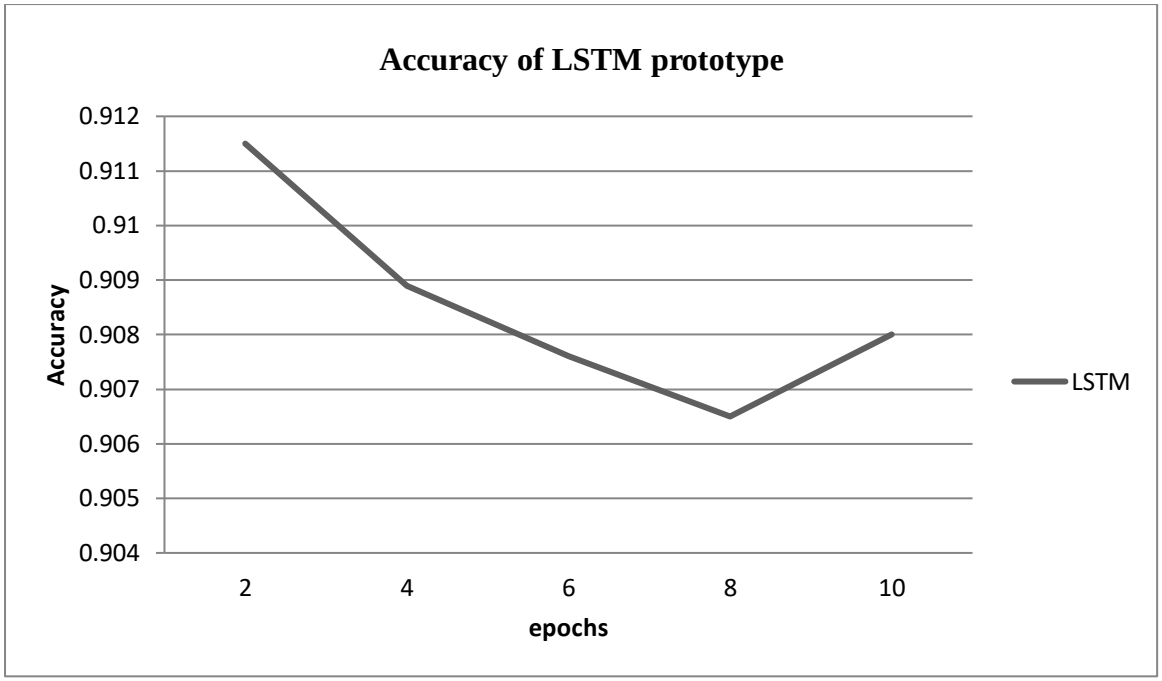


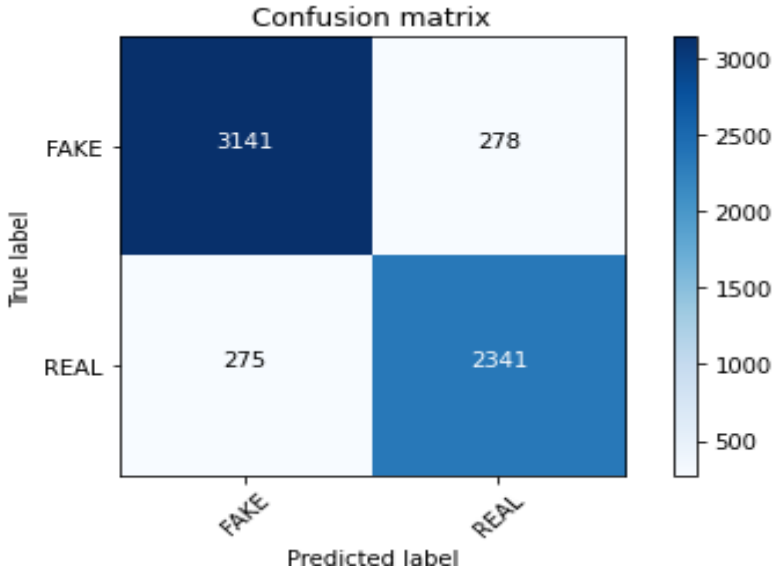

Proposed system module

# 6.4 Accuracy


Accuracy of countvectoriser prototype
0.93
0.92
0.91
0.9
0.89
0.88
0.87
0.86
10% of testing data
20% of testing data
30% of testing data
40% of testing data
Naive Bayes
PAC


Accuracy of TFIDF vectoriser prototype
0.93
0.925
0.92
0.915
0.91
0.905
0.9
0.895
0.89
0.885
10% of testing data
20% of testing data
30% of testing data
40% of testing data
Naïve Bayes
PAC




Confusion matrix of LSTM prototype

| | Countvectorizer accuracy | | TFIDF accuracy | |
|---|---|---|---|---|
| Test size | **Naïve bayes** | **PAC** | **Naïve bayes** | **PAC** |
| **10%** | 0.881 | 0.917 | 0.902 | 0.912 |
| **20%** | 0.888 | 0.917 | 0.901 | 0.921 |
| **30%** | 0.897 | 0.919 | 0.952 | 0.924 |
| **40%** | 0.880 | 0.915 | 0.940 | 0.916 |

| epochs | LSTM accuracy | Loss |
|---|---|---|
| **2** | 0.911 | 7.265e-04 |
| **4** | 0.909 | 4.603e-05 |
| **6** | 0.908 | 1.664e-05 |
| **8** | 0.905 | 1.660e-05 |
| **10** | 0.908 | 1.450e-05 |

# 7. Results

In order to evaluate the effectiveness of our classifier, we used four attributes. The first one, True Positive (TP) is the amount of articles correctly classified as real. The second one, True Negative (TN) is the amount of articles correctly classified as fake news. Next, False Positive (FP) is fake news incorrectly classified as real news. Finally, False Negative is real news incorrectly classified as fake. Using these four attributes the accuracy and recall for both classes. To further evaluate the results, the program runtime was also calculated.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Recall(P) = \frac{TP}{TP + FN}$$

$$Recall(F) = \frac{TF}{TF + FP}$$

For out of all of the 20 trials run with this program, the best training to testing data ratio was 67 percent training data and 33 percent testing data. The classifier was able to achieve an accuracy of 91.317284 percent and an average program runtime of 4.4118 seconds.

# 8. Introspection

Although seeing how well our model did is important, what is equally as important is to see what it learned. For each trial of the program, these terms have often come up as predictive factors for news being fake or real.

Most real words for real News

1. Trump
2. hillari
3. clinton
4. elect
5. new

Most fake words for real News

1. abe

2. abroad
3. abus new
4. abus new york
5. act new

# 9. Limitations

While the results discussed herein suggest for model some external features like source of the news, author of the news, place of origin of the news, time stamp of news were not considered in our model which can be influence the outcome of the model. Availability of datasets and literature papers are limited for fake news detection. The length of the news that is heading or whole news is less which affects the result in terms of accuracy. In Fake News with increasing in layer of module training time increases.

# 10. Application

Journalism: The major spread of information and trusted source is through newspapers and news channels, so this detection can be used to verify the news before broadcasting it.

Social Media: In today's world of social media, it is easy to manipulate any information or news. Such manipulated news misguides the readers. It is import ant to identify that news is fake or real. This paper provides various techniques that can be used in detection and classification of information.

# 11. Conclusion

By comparing the three prototypes, it becomes clear that the refined prototype is more applicable to real world scenarios and larger data sets. According to a study from the University of Michigan, humans can only detect news at a 70 percent success rate. The models we have created easily surpass that by reaching accuracy rates of over 90 percent. While our prototypes have very high accuracy rates, there are also major flaw within them. For instance, they do not fact check, so that means that blatantly incorrect articles can be classified as real if they are written convincingly enough. It also cannot detect click bait. These programs and models that

were created shouldn't be used by itself, but instead in a combination with human checkers. What these prototypes can do is that it can help filter out fake news and the news that is classified as real can be evaluated for validity by humans. Also, the people that are making"fake" news are always changing their styles, so the data for training the classifier always has to be updated.

Here in this paper we compared various methods like Bag Of Words(BoW), TF-IDF, Naïve Bayes etc. LSTM to be most effective of all we used various techniques like stop word removal, one hot representation, word embedding. Model mentioned in this paper is very effective, also compared to existing system the model proposed here gives better results with accuracy of 91.05% which is very promising; we can further increase results by increasing training data.

# References


- Crammer, K., Dekel, O., Keshet, J., Shalev-Schwartz, S., & Singer, Y. (n.d.). Online Passive-Aggressive Algorithms. Retrieved January 22, 2019, from crammer06a.dvi (mit.edu)
- Understanding LSTM Networks: https://colah.github.io/posts/2015-08-Understanding-LSTMs
- GENERATIVE AND DISCRIMINATIVE CLASSIFIERS: NAIVE BAYES AND LOGISTIC REGRESSION. (2017, September 23). RetrievedJanuary22,2019,from: https://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf
- Lu, J., Zhao, P., & Hoi, S. C. (n.d.). Online Passive Aggressive Active Learning and Its Applications. Retrieved January 22, 2019, from: http://proceedings.mlr.press/v39/lu14.pdf
- Sklearn.metrics.confusionmatrix. (n.d.). Retrieved January 28, 2019, from: http://scikitlearn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html