

Fake News Detection using Natural Language Processing Techniques

Abhay Jagannath 2019A8PS0648H

Anany Prakhar 2018A7PS0211H

Chowhan Tanmay Tushar 2018B5A71056H

Arabelly Abhinav 2018B4AA0868H

Mushkan Sureka 2018B5A70477H

Chinmay Madan Bidarkar 2020B5A71863H

Table of content

Table of content	2
INTRODUCTION	3
DATASETS	3
EXPERIMENTS CONDUCTED	4
Baselines	4
KNN	4
XGBOOST	4
SVM	5
DECISION TREE	5
NAIVE BAYES	6
RANDOM FOREST	6
Improvements	7
Neural Network Based Classification	7
Main Findings And Accomplishments	8
References	9
Work Done by Each Member	9

INTRODUCTION

Fake News contains potentially false information that can be verified. This perpetuates a deception about a country's statistics or exaggerates the expense of specific services for a government, causing unrest in some countries, such as the Arab Spring. Organizations such as the House of Commons and the Crosscheck project are attempting to address concerns such as author accountability. However, because they rely on human manual detection, their scope is severely limited. In a world where millions of items are removed or published every minute, this is neither accountable nor possible. A solution could be the creation of a system that provides a trustworthy automatic index scoring, or rating, for different outlets' reliability and news context.

Here we propose a method to create a model that will detect if an article is authentic or fake based on its words, phrases, sources and titles, by applying supervised machine learning algorithms on an annotated (labeled) dataset, that are manually classified and guaranteed. We aim to investigate the application of text classification problems for domains such as mental health and fake news. In the paper, we have explored neural networks for classifying Fake News.

DATASETS

We used the **LIAR-PLUS Master** dataset for our algorithms but initially we cleaned and extracted useful information from the data set, and the algorithms are applied. This dataset has automatically extracted proof sentences from the full-text verdict article published in Politifact by journalists. We used the features of the Truth values, in addition we applied part of speech on the statement to get another 4 features (nouns, verbs, preposition and sentences) and each record is labeled by class label as (True(mostly-true, partially-true), False(false, pants on fire) to be used in training the model.

The following steps have been used to evaluate the precision of the news.

1. Liar-dataset is preprocessed 12.8K.(size)
2. The texts in multiple contexts are taken from POLITIFACT.COM and are labeled manually.
3. We then labeled each column with its respective definitions (like `'ID'`, `'label'`, `'Statement'`, `'Subjects'`, `'Speaker'`)
4. The next step is to clean the noise using NLP NLTK libraries and SAFAR v2 library. The noise involves ids, dots, commas, quotations, and by stemming terms, delete the suffix. In the next step, we use POS (Part of speech) which will turn the dataset into tokens and statistical values.

5. Then we perform feature extraction by choosing lexical features, Such as word count, average word length, length of article, number count, number of sections of speech (adjective). Again, NLTK libraries were used.
6. We then divided the dataset into 75% for train and 25% for test using python sklearn.
7. We then test the model precision on the test portion of the dataset and evaluate the accuracy for fake and real news.

EXPERIMENTS CONDUCTED

Baselines

The following figure shows the attributes corresponding to each sentence and label corresponding to fake or real news. Features such as nouns, verbs were extracted using nltk libraries while barely true, false count etc were a part of the dataset.

Statement	Label	barely true	false count	half true count	mostly true	pants on fire	nouns	verbs	prepositions	Sentences
Over the past five years the federal government...	True	1.0	2.0	1.0	1.0	0.0	6	1	2	1
ys that Tennessee law requires that schools ...	True	0.0	0.0	0.0	0.0	0.0	11	2	6	1
onald Trump is against marriage equality. He ...	True	0.0	0.0	0.0	0.0	0.0	4	0	1	2

KNN

KNN is a supervised type of learning algorithm that can be used for both classification and regression. It uses feature similarity to find out the values of test data points, which implies that the new data points will be assigned values based on how close they are to the points in the training set. The position assigned in the class is highly mutually exclusive between the nearest neighbors K, as measured by the role of the distance.

XGBOOST

Xgboost is an enhanced version of an edge detection algorithm. It is a highly performant algorithm and is trending in competitions. Library implementations can work in parallelised environments, and even under resource constrained environments. The dataset for fake text detection is ideal for xgboost as it is optimized for training on tabular and structured datasets for classification problems. The xgboost boost algorithm is developed on top of the decision tree algorithm which takes residuals of prior models in prediction. It uses a gradient descent algorithm to minimize errors.

SVM

The Support Vector Machine Algorithm for Classification derives from the layout of data in the form of a point in a range of dimensions defined by the number of properties the data has. The value of a given property is the number of specified coordinates.

An SVM generates a set of hyperplanes in an infinitely high-dimensional space used for classification and regression. Given a set of n-features, the SVM algorithm uses an n-dimensional space to plot the data item with the coordinates representing the value of each feature. The hyperplane achieves a good separation with the most significant distance to the nearest training-data point of any class, defined as the functional margin.

We then achieve classification based on the hyper-plane obtained to separate the two classes by the SVM algorithm.

```
F[0..N-1]: a feature set with N features that is sorted by information gain
in decreasing order
accuracy(i): accuracy of a prediction model based on SVM with F[0...i] gone
set
low = 0
high = N-1
value = accuracy(N-1)
SupportVectorMachine(F[0...N-1], value, low, high) {
  If (high } > low)
  Return F[0...N-1] and value
  mid = (low + high ) / 2
  value_2 = accuracy(mid)
  if (value_2 < value)
  return SupportVectorMachine(F[0...mid], value_2, low, mid)
  else (value_2 > value)
  return SupportVectorMachine(F[0...high], value, mid, high)
}
```

Pseudocode for SVM Classifier¹

DECISION TREE

It is a tool with a tree-like internal structure. It is created using gini coefficients and these coefficients determine the decision nodes. Decision trees are useful in classification problems. Input is processed through the tree and the result is the final class label the input is attached to. The distance from the root to leaf represents the classification rule. Decision trees are good at selecting the impactful variables. It is also able to learn correlation among the variables.

¹ Khanam, Z & Alwasel, B & Sirafi, H & Rashid, Mamoon. (2021). Fake News Detection Using Machine Learning Approaches. IOP Conference Series: Materials Science and Engineering. 1099. 012040. 10.1088/1757-899X/1099/1/012040.

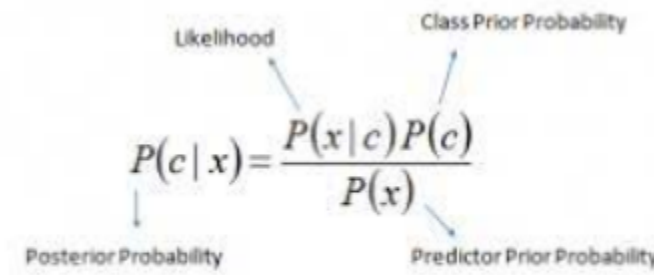
Decision tree algorithms can be accurate when used with supervised predictive learning models and can map non linear relationships.

NAIVE BAYES

This classifier works on the famous probabilistic algorithm of Bayes theorem. Considering each factor as an independent contributor to the classification it gives the results.

For applying this classifier to our dataset we perform following steps:

- 1] uploading dataset in the system
- 2] we take out mean of predictor's variables
- 3] we take out magnitude of predictor's variables and repeat it again
- 4] using the gauss density function we calculate likelihood
- 5] calculating the likelihood of all classes
- 6] and getting the highest likelihood



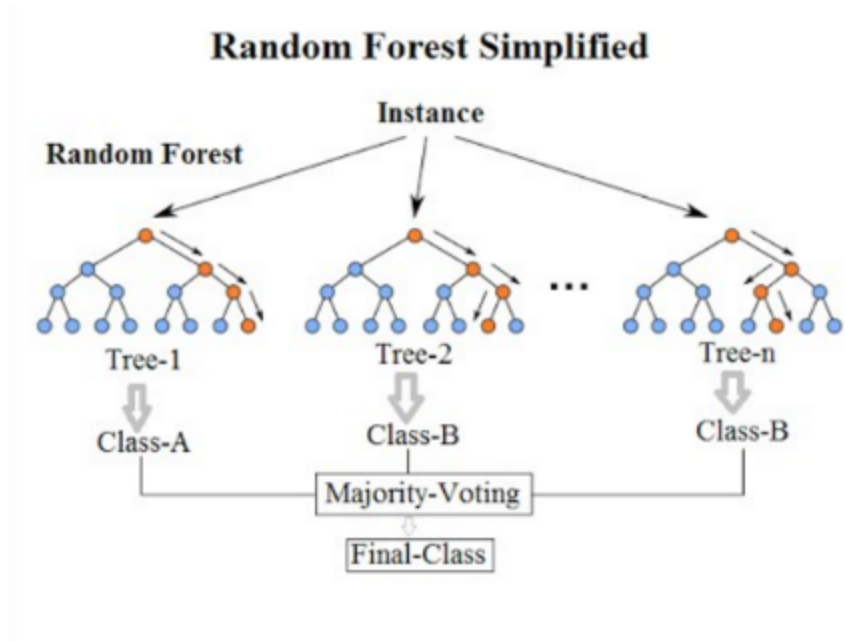
The diagram shows the formula for Bayes' Theorem: $P(c | x) = \frac{P(x | c)P(c)}{P(x)}$. Arrows point from labels to parts of the formula: 'Likelihood' points to $P(x | c)$, 'Class Prior Probability' points to $P(c)$, 'Posterior Probability' points to $P(c | x)$, and 'Predictor Prior Probability' points to $P(x)$.

$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$$

RANDOM FOREST

This classifier is the extended version of the decision tree model. We build many decision trees in this model hence named forest. Its results are those which are given by most of the trees. The ability of random forest to adapt variation in dataset values puts it ahead of the decision tree model. It's called random since it's applied on those decision trees which are not connected with each other's constraints. And we obtain them by bootstrapping.

Using the present library of RandomForestClassifier in sklearn.ensemble we classify our data through this model and get the adjoining results.



Improvements

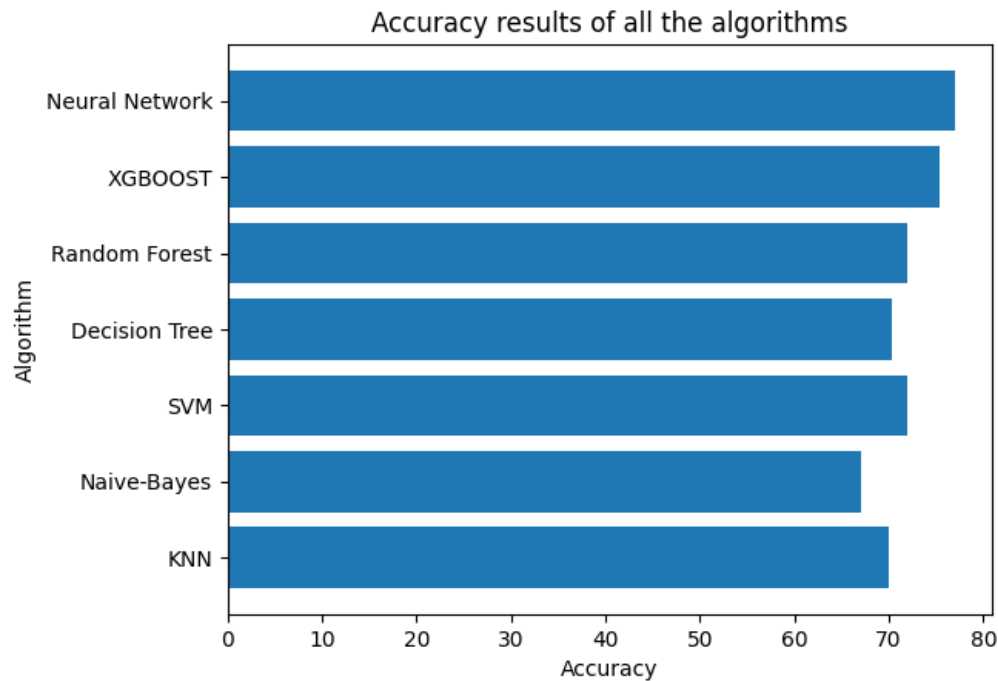
Neural Network Based Classification

Neural Networks are a popular supervised machine learning algorithm for both classification and regression tasks. They help to cluster and classify the data. They comprise the input layer, hidden layer(s), output layer and the node layer. Each sentence is represented as a 9 dimensional vector consisting of various attributes like nouns, verbs etc. The goal is to classify the given text as Fake or Real News. We train an artificial neural network consisting of input layer, single hidden layer of size 12 neurons using relu activation function and output layer of size 1 (sigmoid activation function) that gives a score between 0 and 1. The model was trained for 200 epochs with Adam Optimiser, batch size = 32 and learning rate= .001.

We then used the model to predict the scores on test data. The threshold was set to 0.5 and scores ≥ 0.5 were set as TRUE and < 0.5 as FALSE. An accuracy of 77% was achieved which is greater than best performing model XGBOOST (75%) in the baseline paper. The code for the same is attached in the [link](#).

Main Findings And Accomplishments

The following graph shows the results obtained for accuracy on implementation of the models in the baseline and the improvement (Neural Network) based method. Using Neural Networks we could achieve an accuracy of 77% which is more than XGBOOST by 2%.



References

- 1) <https://iopscience.iop.org/article/10.1088/1757-899X/1099/1/012040/pdf>
- 2) <https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/>
- 3) <https://realpython.com/python-ai-neural-network/>
- 4) <https://www.geeksforgeeks.org/random-forest-regression-in-python/>

Work Done by Each Member

Abhay Jagannath	Implemented SVM, Write up on SVM
Mushkan Sureka	Data Pre-processing, Write-up on Introduction
Anany Prakhar	Implemented XGBOOST, Decision Tree; Write up on same
Abhinav Arabelly	Implemented KNN, Neural Network Method, Write up on same, Main Findings
Chowhan Tanmay Tushar	Data Pre-processing, Write-up on Data Pre-Processing
Chinmay Madan Bidarkar	Implemented Naive Bayes, Random Forest. Write up on same