Alternus Vera: Fake News Detection

Vignesh Kumar Thangarajan
Dept. Software Egineering
San Jose State University
San Jose, CA, USA
Contribution: Political Bias
vigneshkumar.thangarajan@sjsu.edu

Pooja Patil
Dept. Software Egineering
San Jose State University
San Jose, CA, USA
Contribution: Title vs Body
pooja.patil@sjsu.edu

Sivaranjani Kumar
Dept. Software Egineering
San Jose State University
San Jose, CA, USA
Contribution: Clickbait
sivaranjani.kumar@sjsu.edu

Akshaya Nagarajan
Dept. Software Egineering
San Jose State University
San Jose, CA, USA
Contribution: Node Rank
akshaya.nagarajan@sjsu.ed

***Abstract*— Social media and its splurge in news consumption is said to be a double-edged sword. As the news content gets bigger, the problem of detecting fake news becomes more relevant than ever. Easily accessible news on social media on one hand and enabling wide spread of fake news on another, for example the impacts on Great Britain's exiting out of European Union. In this paper, we are proposing new features to consider for building the classification model. The new features will help the model to be resilient. Training the model with limited dataset makes the model to overfit the data. To avoid that and to enrich the dataset, data amalgamation is applied to the primary PolitiFact data with two other external news datasets. Demonstrating Latent Dirichlet Allocation on the dataset to generate topic naming which also helped to extract the latent variables in the dataset. A polynomial equation of accuracy and the factors is formed which will be helpful in evaluating the overall model's metric. One cannot afford the model to classify fake news and real news and vice versa. Thus, while training the model, both the false positive and false negatives are treated with equal importance to strike the balance in evaluation metric.**

Keywords*— *Data amalgamation, Distillation Process: Data cleaning and text preprocessing, Visualization

I. Introduction

The problem of fake news is prevalent in all media. It has taken toll especially in Social Media. The reason being news can be created by anyone who has access to mobile and internet. On a positive note, there are lot of coverage for every single event. Places that are remote where it is impossible for the Satellite television could reach, civilians having access to internet are able share the news Worldwide in a matter of minutes which is essential and need of the hour. Unfortunately, the volume of those news is getting higher and higher day by day. Some of that news have no credibility and are fake news. Apart from that, observations are made that there are various paid medias that do false propaganda against any organization or entity without any basis.

This has opened a new arena for detecting Fake News. This is a problem of big data which has both high volume of data and solving the problem of veracity. The model also needs to be updated as the news types are evolving which adds another dimension of data velocity to it. In this project, the primary data is collected from PolitiFact website which rates the authenticity and veracity of the news getting published in each media.

II. Dataset, Scraping and Enrichment

A. *Datasets:*

Considering the recent manifestation of Fake News detection and the factors that affect the notion of knowing which news is fake and which is real, not many resources are available to predict a news to be fake or real. Few of the possible datasets which were considered for this project are explained below.

Liar Liar Dataset:

The LIAR dataset is a dataset which includes 12.8K human labeled short statements from PolitiFact.com. Each statement in the dataset is evaluated by PolitiFact's editor for its truthfulness. The labels indicate the degree of falsity and are as follows:

True	Mostly-true	Half-true	Barely-true	False	Pants-fire

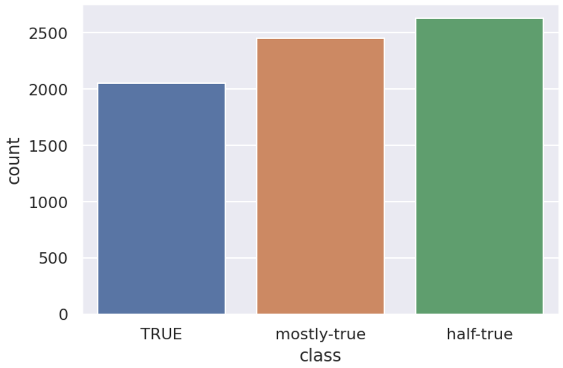


Fig – All true Liar-Liar dataset

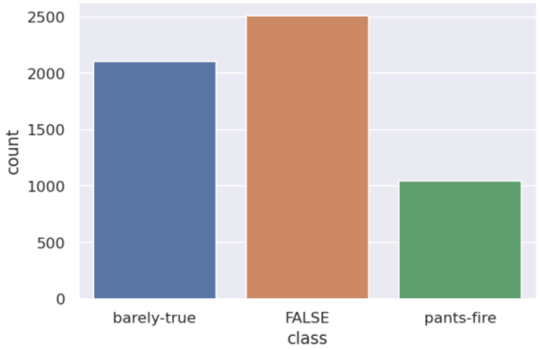


Fig – All false Liar-Liar dataset

Kaggle Fake Dataset:

The Kaggle's Fake Dataset contains the text and metadata from fake and biased news source around the web. The dataset contains 12,999 posts in total collected from 244 websites in a period of 30 days. As the name suggests, it contains all fake news posts. The dataset contains following columns:

uuid	Unique id of news posts
ord_in_thread	Thread order
author	author
published	Published date-time
title	Title of the posts
text	Body of the posts
language	lanaguage
crawled	Creawled date-time
site_url	Site url from where the posts were crawled
country	Country of posts
domain_rank	Rank of the sites crawled
thread_title	Title of thread
spam_score	Spam score of posts
main_img_url	Actual url
replies_count	Count of replies
participants_count	Count of participants
likes	Likes on posts
comments	Comments on posts
shares	Counts of shares
types	Type of posts

News Category Dataset from Kaggle(HuffPost)

The News Category Dataset is about categorizing the type of news based on headline and short description. The dataset has around 200l headlines and short description from year 2012 to 2018. The dataset is available as a JSON file. Each JSON record contains the following attributes:

Category	Category the article belongs to
Headline	Headline of the article
Author	Person authored the article
Link	Link of the Post
Short Description	Short Description of the post
Date	Date when the article was published

For the purpose of this project, we converted this JSON file into CSV file format.

B. Data Amalgamation:

The process of data amalgamation is one of the important tasks. The datasets are from different source and they follow different probability distribution. So, to amalgamate them is not an easy task. Initially, some proposals are considered, like computing a new feature using other features, but the variety of the datasets doesn't allow that. The text content of the news in datasets are similar in terms of distance similarity. As already discussed, the news category dataset is filtered based on only the political news category. This helps to compliment the PolitiFact dataset.

Finally, only the useful and common features are filtered among the all three datasets. They are shown in the table below:

Author	Title	News Text	Site URL

III. Implementation Details

A. Pre-processing:

Removing stop words, there are some words in english language which does not contribute meaninig to the phrase, words such as before, had, when are called as stop words and they are filtered using nltk library stop words corpus. Numbers and whitespace characters are removed. The leading and trailing whitespaces are of no use. Lemmatization is a process of grouping together the different inflected forms of a word so they can be analysed as a single item. This is similar to stemming but it brings context to the words. So it links words with similar meaning to one word. These steps are standard in all NLP tasks and it is followed and applied on the news text.

B. Visualization:

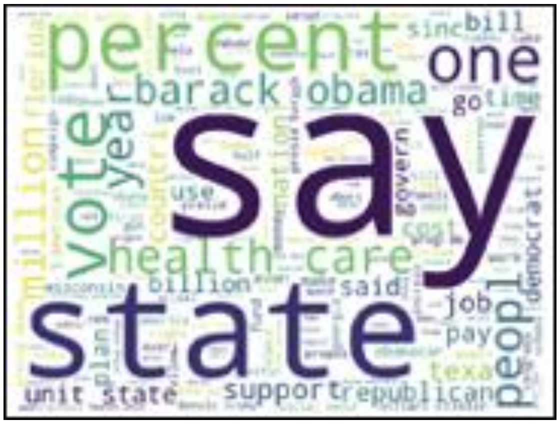


Fig – Word Cloud Visualization

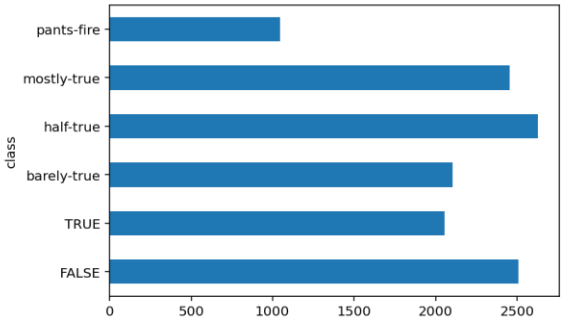


Fig – Liar –Liar Dataset

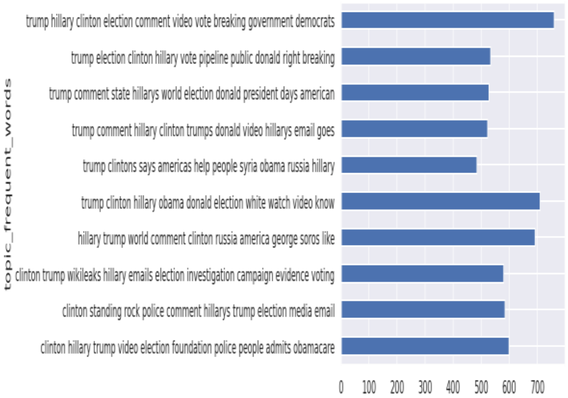
C. TF-IDF

Term Frequency-Inverse Document Frequency is another vectorization technique used to vectorize text using word frequency. TF-IDF assigns lower weights for common words. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. TF-IDF Vectorizer converts a collection of raw documents to a matrix of TF-IDF features. This is applied in most of the individual factor analysis. The vectorized words are used to train the model to classify the news to correct category. The train data is from liar-liar dataset. Using this model, the labels are re-assigned to the appropriate classes in other two dataset.

D. Topic Modeling:

Latent Dirichlet Allocation (LDA) is a Probabilistic, generative model which uncovers the topics latent to a dataset by assigning weights to words in a corpus, where each topic will assign different probability weights to each word. In LDA, the modelling process revolves around three things: the text corpus, its collection of documents, D and the words W in the documents. Therefore, the algorithm attempts to uncover K topics from this corpus. The LDA algorithm first models' documents via a mixture model of topics. From these topics, words are then assigned weights based on the probability distribution of these topics. It is this probabilistic assignment over words that allow a user of LDA to say how likely a word falls into a topic. Subsequently from the collection of words assigned to a topic, are we thus able to gain an insight as to what that topic may represent from a lexical point of view. From a standard LDA model, there are really a few key parameters that we must keep in mind and consider programmatically tuning before we invoke the model.

In the dataset, the news statements are preprocessed. The statements are converted to word tokens. The list of tokens is converted to list of token ID and token counts using doc2bow. The result of doc2bow is fed into LDAMulticore in Genism Library to get an LDA model specific to our dataset. The topic number and topic scores are calculated from the LDA model created. These two are calculated for each news statement and included.in the dataset. Sample topic generated - ['says', 'percent', 'state', 'states', 'years', 'year', 'said', 'trump', 'united', 'president'], ['says', 'percent', 'said', 'state', 'health', 'trump', 'wisconsin', 'campaign', 'country', 'obama'] for one specific news statement.



E. Ranking:

The Kaggle news category and Liar-Liar datasets contains author and speaker respectively. Ranking for the dataset is calculated based on the frequency of occurrence of speakers and authors. The frequency count for each author is taken and the top 10 authors/speakers are considered and ranked accordingly. The speaker/author with least frequency is given a value of 1. This ranking is added as an attribute in the dataset.

Individual Contribution:

Based on the above data explorations, we as a team decided to compute factors using three datasets. The datasets were viewed from different angles to ensure the important factors that affect the dataset to swing between fake and real news were identified and divided amongst the team. This approach helped us with different polynomial equations for each factor, building towards a model:

Feature	**Author**
Political Bias	Vignesh Kumar Thangarajan
Title Vs Body	Pooja Patil
Clickbait	Sivaranjani Kumar
NodeRank	Akshaya Nagarajan

IV. Feature Vector

A. Political Bias:

Political bias is when a reporter, news organization, or TV show slants or skews facts in order to make their personal political position look more attractive. It can take many forms, from ignoring contradictory evidence, asking unbalanced questions, or cleverly framing facts and stories to change the public perception. There is a thin line between reporting the fact and reporting the fact for some personal gain. So, the news text is important and is subjected to apply text mining on it. There are several ways of doing text vectorization. Tf-Idf method of vectorization is better than simple count-vectorization as it attenuates the word that is frequent in overall corpus. We also used random forest classifier to reassign the labels in the other two datasets.

Assumption: The Liar Liar dataset has Party Affiliation column. It is a categorical column containing 24 unique values. Using domain knowledge, we assigned arbitrary values indicating political bias. The range of values are between 0 and 1. News articles affliated to parties like democrat, republicancs, libertarian are considered to be more bias and hence assigned values that are greater than 0.5. Whereas, articles related to Journalist, columnist, or independent candidates are tend to be less affliated to any party. So, those categories got value between the range of 0 to 0.5.

Data: Now filtered the Liar-Liar dataset by rows containing labels related to all fake news and another dataset related to all true news.

Model: Used these two datasets to train two different Feed Forward Neural Network models. The feature is the vectorized data of news text and label is the political bias. The trained model is then used to predict the political bias for both Kaggle fake news dataset and Kaggle News Category dataset.

Solution of feature size mismatch: The number of features in these datasets are varying. Padding the array that has less features to match that of higher feature array helps to solve this problem. The test data can then be fed to the same model without facing the input dimension mismatch error.

Comparisions of models after predicting Political Bias: The regression values from the neural network model, are now the feature values to our classifier models. Random Forest, XGBoost and Decision Tree classifiers are used to classify the political bias indices. The performance of these models are more or less similar in terms of accuracy score and f1-score. Random forest performed slightly better than other models in overall metrics.

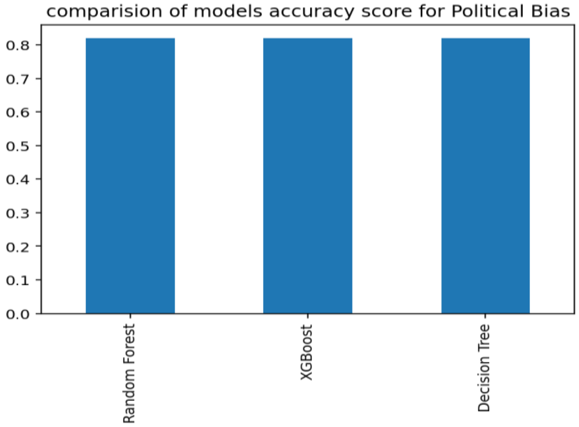


Fig – Comparison of model performance using accuracy score

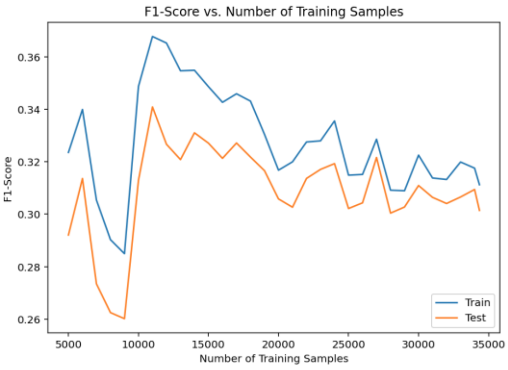
B. Title Vs Body:

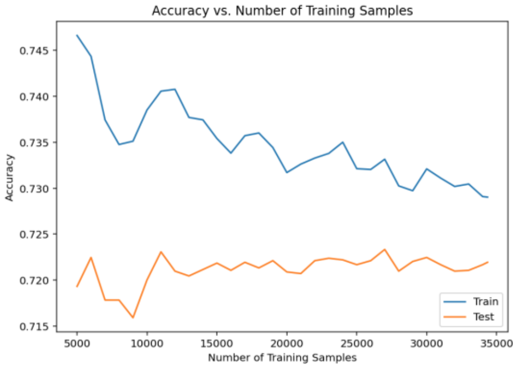
Title and Body of any news article can be used to assess news article as first step in determining whether the news article is Fake or Real. Titles are strong differentiating factor between real and fake news. The incongruence or dissimilarity between the Title and the Body can help in detecting the fakeness of a news articles. To detect similarity and the fakeness labels, two approaches are followed.

Following approach is used to model the 'Title vs Body' factor.

1. Get the Liar Liar Dataset, Kaggle Fake News Dataset, and News Category Dataset.
2. Initial Data Exploration and Data Cleaning.
3. Removed unwanted columns from all the three datasets and retained only the 'Title', 'Body' and 'Fakeness' label (Classification label) column.
4. Cleaning Textual data using NLP's stopwords removal, lemmatization, stemming, etc.
5. Create Corpus of text to find Similarity between 'Title' and 'Body'.
6. Feature Engineering: Tokenization of 'Body' into sentence tokenization and word tokenization.
7. Train the multiclass classification algorithm using the similarity generated between 'Title' and 'Body' to predict fakeness of the news articles.

Initially, Training is performed using classification algorithms separately on Title and Body of the news articles. The models used to train the 'Title vs Body' factor is Random Forest and XGBoost. The models performed well on the Body part of the news articles with accuracy around 81% compared to the Title which has accuracy around 73%.





The Second approach considered finding the Similarity between the Title and the Body. Using Doc2Bow, the embeddings resulted in powerful representation of the raw text data. Using the embeddings generated by Doc2Bow, Similarity is generated based on Jaro-Winkler similarity. This similarity generated is then used to predict the fakeness of the news articles.

C. Clickbait:

When the link of a news article is made appealing along with a flashy headline, it provokes the users to click. Clickbaits are highly attention seeking in any websites. They can be pointed out by the use of certain sentences, images, keywords and symbols. Clickbait (headlines) make use of misleading titles that hide critical information from or exaggerate the content on the landing target pages to entice clicks. As clickbait's often use eye-catching wording to attract viewers, target contents are often of low quality. Clickbait's are especially widespread on social media such as Twitter, adversely impacting user experience by causing immense dissatisfaction. Hence, it has become increasingly important to put forward a widely applicable approach to identify and detect clickbait's

1. Feature Engineering:

As part of feature engineering, I did tokenization of text using gensim library from gensim.utils.simple_preprocessing. Also collapsed all the white spaces into single spaces along with elimination of leading and trailing spaces.

Stemming: Stemming is about getting the words by removing suffixes. With stemming, words are reduced to their word stems. A word stem need not be the same root as a dictionary-based morphological root, it just is an equal to or smaller form of the word. So I applied stemming on the document to get the root word.

Dictionary words: Dictionary encapsulates the mapping between normalized words and their integer ids by making use of tokenized words from previous step.

Doc2bow: This will convert the documents into the bag of words format (token_id, token_count). Each word is assumed to be tokenized and normalized before converting to doc2bow. I used a gensim library to convert the documents to doc2bow format.

TF-IDF: Computing the tf-idf by multiplying term frequency as a local component and inverse document frequency as a global component and normalizing the resulting documents to unit length. TF gives us the frequency of the word in each document while IDF give us the weight of all rare words present in the document.

LDA: Using tf-idf to compute LDA and built a topic per document model and words per topic model, modeled as Dirichlet distributions. In LDA we divide the document on the basic of topics and classify words as per topic.

2. Visualization:

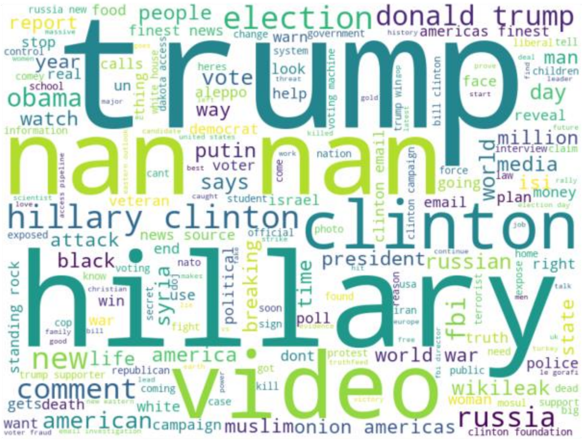


Fig – Word cloud visualization for clickbait

3. Feature Extraction:

- First Approach:

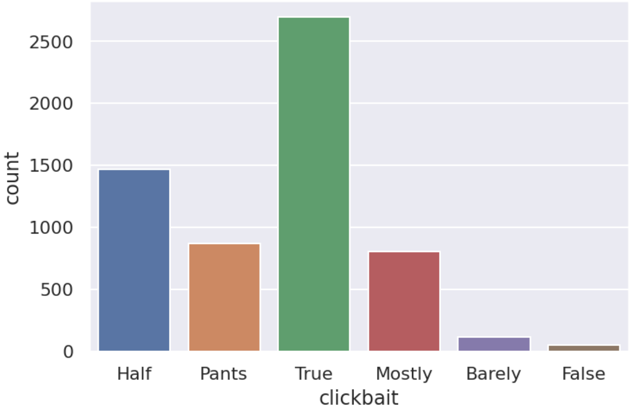
We have considered three columns from the fake news dataset to compute clickbait feature. First, I have created three columns named topic number, topic score and topic frequent words by applying LDA in clean title column of clickbait dataset. Since all the topics are related to politics and the topics generated are very similar in each document, with this column we couldn't compute the clickbait feature as multi class.

- Second Approach:

In the second approach, we computed the clickbait feature by ranking the clickbait based on the rules [question marks, exclamation, modulo, hash, caps ratio and length of text]. When we tried to compute using all these functions, we ended up getting biased clickbait features as 90% of the data falls under false category out of six classification.

- Third Approach:

In the third approach, we have considered the frequency of unique words which are present in the title column and based on these frequency of words we classified the clickbait feature as [false_clickbait, barely_clcikbait, pants, true_clickbait, mostly_clickbait, half_clickbait]. By doing so, we were able to control the biasness in the dataset and achieve some good accuracy.



4. Neural Network and Classification Model:

With the clickbait feature that we computed, we passed the title and clickbait label column to the Keras regressor model and able to predict the clickbait label for other two datasets [news category and liar-liar dataset]. Later we merged all the three dataset and applied to the three-classification model – Random Forest Classifier, XGBoost, and Decision Tree Classifier. When compared the accuracy of all three classification models, Random forest gave the highest accuracy of 72%.

D.Node Rank:

To assess the authenticity of a news article, we can analyze the source of the news article. Each article published in a domain is given a rank based on the number of its neighbours - outgoing links from the article and incoming links to the article. It can determine how influential a news article can be.

1. Feature Engineering

For Alternus Vera, the Liar-Liar and Kaggle News Category datasets are considered. The Liar-Liar dataset contains Politifact links which holds the information about the article - such as in which particular websites the specific article was published, date of publishing and other facts about the News article.

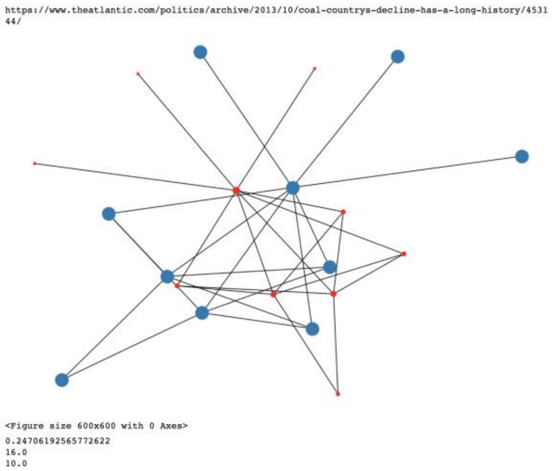
For Page rank/Node rank, we require the actual website in which the article was published. For this, we can either scrape the News publishing website URL from politifact webiste or we can make a Google search to collect the list of URLs in which the News was actually issued. The later approach was taken. Google search for the actual statement was carried out. The search results and the corresponding links with <a> tag and href attributes in the first page of Google search was scraped. This list was added to the dataset.

Considering the first link, from the list of scraped links, using BeautifulSoup Library, all the URLs in the web page was scraped. NetworkX Library in Python can be used to create a graph structure for the network and visualize it. Each web page is appended as a node and the outlink from one page to other is appended as an edge in the graph. NetworkX Library also provides functions to calculate the PageRank, number of edges and number of nodes for a web page. All these attributes are calculated and included as features in the dataset.

Similar approach was taken for Kaggle News Category Dataset. This Dataset contains the web page of the news article as a column. So, all the URLs in each web page were scraped using BeautifulSoup and PageRank, number of edges and number of nodes are calculated using NetworkX Library and appended to the dataset.

Features Used for Classification: NodeRank/PageRank, number of edges, number of nodes, topic number, topic score and author/speaker rank. Both LDA and Ranking was implemented to get features (topic number, topic score and author/speaker rank) as part of distillation.

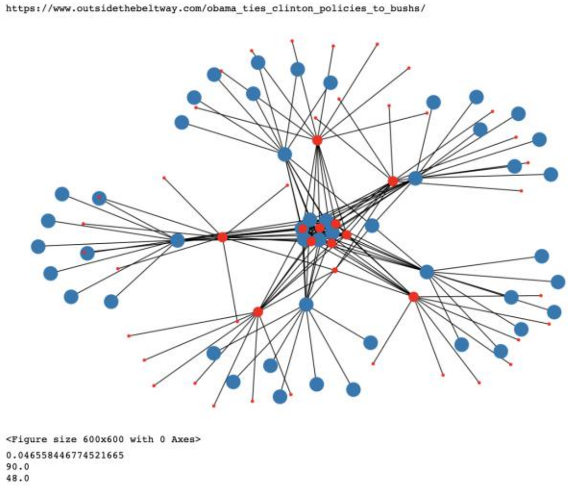
2. Visualization



```
https://www.theatlantic.com/politics/archive/2013/10/coal-countrys-decline-has-a-long-history/4531
44/
```

```
<Figure size 600x600 with 0 Axes>
0.24706192565772622
16.0
10.0
```

Fig – Graph Structure for web page



```
https://www.outsidethebeltway.com/obama_ties_clinton_policies_to_bushs/
```

```
<Figure size 600x600 with 0 Axes>
0.046558446774521665
90.0
48.0
```

Fig – Graph structure for web page

3. Classification

The dataset is fit, trained and tested using Multiclass classifiers like – XGBoost, RandomForest, Decision Tree and Multinomial Naïve Bayes. All these four models are applied at different stages – before amalgamating Liar-Liar and News Category datasets, after amalgamating the same, after distillation process and adding additional features like topics, topic score and ranking using speaker/author columns.

4. Results

Comparision of these model results are analyzed. Before amalgamation and distillation, all the models produced low accuracy of around 20%. After amalagamating (without distillation), the accuracy of the models improved with XGBoost producing 43% highest accuracy. Similarly, after distillation process, the accuracy improved slightly, again with XGBoost producing 43% - 45% accuracy. Clearly, amalgamation and distillation played important roles in improving the accuracy.

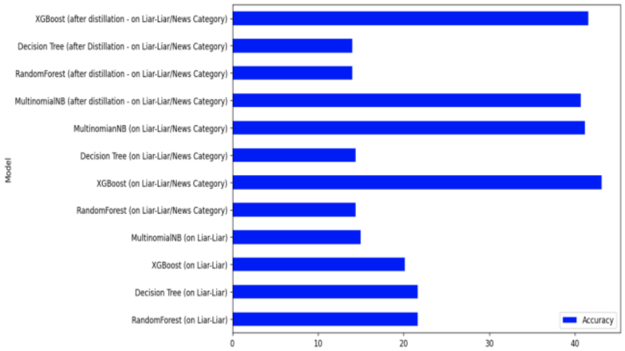


Fig – Classification Models

V. Conclusion

As a team we have collected fake news datasets, news category and liar liar dataset from Kaggle. we decided on the importance of the factors presented in this paper. We brainstormed on the general pre-processing techniques we did want to use. We also had common visualization methods and similar techniques for evaluating the classification model accuracy. Each of us enriched the dataset with individual features and persisted it in a csv file. We also came up with a polynomial equation based on the factors and the accuracy scores we received by classification. The polynomial equation is then used to build a model for fake news classification. The final model that we built is a variation of the ensemble technique. Ensemble method is where the models are combined. The basic idea is to train machine learning algorithms with training dataset and then generate a new dataset with these models. Then this new dataset is used as input for the combined machine learning algorithm. The combined model is then used to predict the fakeness in the corpus. We as a team were able to achieve an accuracy of (percentage) using the various supervised learning techniques specified in this paper

References

[1] https://www.kaggle.com/mrisdal/fake-news/kernels

[2] https://www.kaggle.com/c/fake-news/data

[3] http://www.fakenewschallenge.org/

[4] https://www.kaggle.com/rmisra/news-category-dataset

[5] https://arxiv.org/abs/1705.00648

[6] https://dspace.mit.edu/bitstream/handle/1721.1/119727/1078649610-MIT.pdf

[7] https://arxiv.org/pdf/1708.00214.pdf

[8] S. Gilda, "Notice of Violation of IEEE Publication Principles: Evaluating machine learning algorithms for fake news detection," 2017 IEEE 15th Student Conference on Research and Development (SCOReD), Putrajaya, 2017, pp. 110-115, doi: 10.1109/SCORED.2017.8305411.

[9] M. Sundermeyer, I. Oparin, J. -. Gauvain, B. Freiberg, R. Schlüter and H. Ney, "Comparison of feedforward and recurrent neural network language models," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, 2013, pp. 8430-8434, doi: 10.1109/ICASSP.2013.6639310.

[10] J. Wang, L. Shi, R. Diao and Z. Wang, "Node ranking of multiplex network based on weighted aggregation using AHP method," *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, Changsha, 2016, pp. 1659-1663, doi: 10.1109/FSKD.2016.7603426.