# Introduction

Beginning with the contentious 2016 US election cycle, the public has become more aware of increasingly widespread proliferation of fake news on the internet and specifically social media. Potential solutions to this problem have been stymied by the fiduciary interests of companies and politicians in the limitation or furtherance of fake and slanted information. Equally confounding is the inconstancy of the layperson’s understanding of what constitutes fake news, or even what the term *means*. In these respects, this blight is self-perpetuating and its cures are self-sabotaging.

But while human beings may find it difficult to escape the bias of opinion, perhaps machine learning can discern between what is simply a different viewpoint, and what is unobjective or intentionally misleading. Modern texting mining methods may not be able to know or understand truth and falsehood, but they very well might be able to identify the hallmarks of those who peddle in disinformation using only the text of that disinformation.

This study seeks to build a corpus and set of models by which the text of publications known to consistently publish fake and misleading news can be differentiated from the text of publications that maintain accepted journalistic standards of integrity. Steps will be taken to prevent the models from conflating differing political views with truth or fiction. The study will produce a composite model that combines the disparate predictive strengths of each individual model for practical use as the *“Is It Real News?” Engine.*

# Analysis and Models

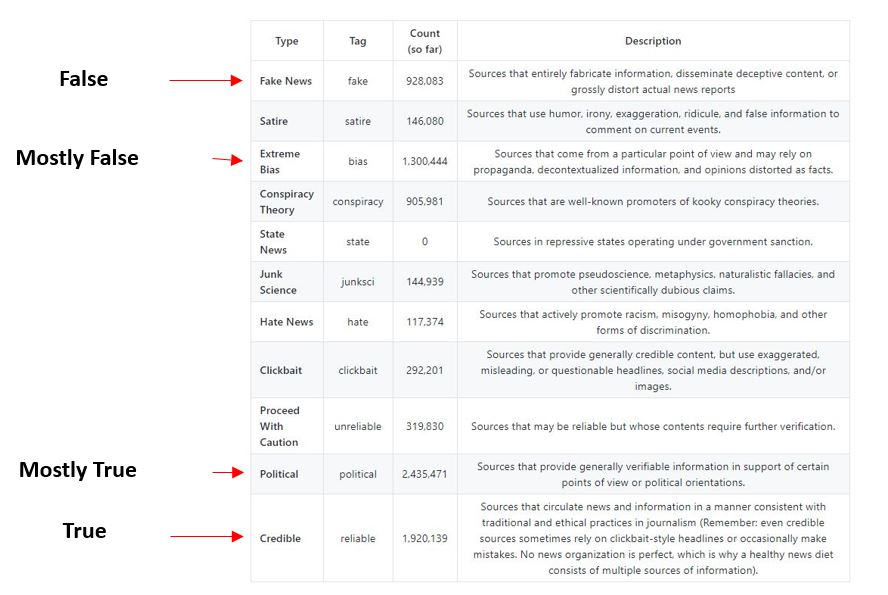
## About the Data

### Source Data

Initial exploratory analysis for this study was executed on the Liar Liar dataset, which sources short statements from Politifact.com and labels them on a 6-level ordinal scale of truth vs. falsehood. However, the shortness of these statements and their frequent deviation from news content made the dataset more practical for simple lie detection.

The dataset that this study used to build its models came primarily from the Several27 Github account’s FakeNewsCorpus dataset. This dataset has many labels that are not necessarily in ordinal format. To build a dataset with ordinal and binary labels, the “Reliable” label became “True”, the “Political” label became “Mostly True”, the “Bias” label became “Mostly False”, and the “Fake” label became “False”. When discretized to binary, “True” and “Mostly True” became “True”, while “False” and “Mostly False” became “False”.

Due to a lack of diversity in the “Reliable” label in FakeNewsCorpus, articles from reliable sources of disparate political leanings were taken from the AllTheNews dataset and added to the “Reliable”/”True” label. All the data was then cleaned of residual special characters left over from the scraping process (e.g. ‘\r’ and ‘\n’) and articles that were far too short in character length to be valid or useful. For the training and testing of each model, the dataset was to be stratified by label and then randomly sampled and shuffled to avoid bias.



A sample of the raw FakeNewsCorpus data is shown below:



### Transformations, Quality and Sampling

#### Transformation

For Document Term Matrix (DTM) Creation, see Model Overview section.

#### Author Type Transformation

Using the author data, a name recognition routine was built (using the NLTK POS\_TAG and NE\_CHUNK routines) to transform the author data into a labelled feature comprising one of the following three labels:

AUTHOR – meaning the author name actually corresponds to a name

MAYBE – meaning that part of the author name corresponds to a name

UNKNOWN – meaning that the author name provided does not contain a name in the wording

#### Quality

The training and testing data that are used are derived from the source dataset on a 60% train, 40% test split basis.

#### Sampling

For the purposes of this analysis, a subset of documents was used

## Model Overview

The objective of our “Is It Real News?” Engine is to receive a textual document as input and then to deliver a prediction or a score which tells whether the document is real or fake.

To deliver the answer, a process has been built that makes use of various machine learning models in either an individual or combined manner to predict or score the results.

The architectural parts of this model (aka “Is It Real News?” Engine) consists of 8 distint processing models, each described separately in the Model section:

1. Model 1: A **Veracity Classifier** that classifies documents with one of four labels
   1. Labels are: False, Mostly False, Mostly True or True.
2. Model 2: A **Binary Predictor** that classifies documents with one of two labels
   1. Labels are: False or True.
3. Model 3: An **Author Type Classifier** that classifies documents with one of three labels
   1. Labels are: Known, Maybe known or Unknown.
4. Model 4: A **Sentiment Classifier** that classifies documents with one of two labels
   1. Labels are: False or True.
5. Model 5: A **Topic Model** that categorizes documents
6. Model 6: A **Composite Model** that uses the individual Predictors and Classifiers as described earlier, together with supplied text, to build a feature set for final prediction or scoring
7. Model 7: A **Supervised Learning Model** that reads the feature set and predicts the final answer
8. Model 8: An **Unsupervised Learning Model** that reads the feature set and scores the final answer

## Model Vectorization

An iterative process was built for parameter driven execution of a selected Vectorizer as well as for multi-model prediction of labels given a set of labelled text as input. Objective was to identify the most efficient vectorizer / model combination for each classifier described above – and then to use that vectorizer / classifier as input to the composite model.

The iterative process is designed to work as follows:

Given a training dataset containing parsed text and corresponding labels:

* Split the train-data into a labeled train and a test set, on a random 60% for train and 40% for test split basis, and then iterate through each rule one-by-one doing the following
  + Execute selected Vectorizer (CountVectorizer or TfidfVectorizer) using rule parameters, ultimately creating a document term matrix (DTM)
  + Fit the test set to the trained vectorizer, primarily to use the same dictionary
  + Train each of the programmed models (linear SVC, multinomial NB and RF) with the vectorized training set
  + For each trained classifier, classify the test set.
  + Calculate accuracy of results as well as confidence factor
  + Once all Vectorizer and Model execution is complete - determine which vectorizer and model combination produced the most accurate result.
  + Classify the unlabeled test-sourced set against the trained classifiers and choose one to submit to Kaggle

To run the Vectorizer (either CountVectorizer or TfidfVectorizer), different combinations of rule parameters were used. The rule parameters used were combinations of the following:

1. Max\_df – which were used for removing terms that appear too frequently. The max\_df values that were used were .2, .4 and .6. What this means is “ignore terms that appear in more than 20%, 40% or 60% of the documents”.

2. Min\_df – which were used for removing terms that appear too infrequently. The min\_df values that were used were 4, 8, 12 which means “ignore terms that appear in less than 4, 8 or 12 documents”.

3. Stopwords – use either None or “english” as the input parameters. This means that any words on the stopwords list will be removed.

4. nGram length – 1,1 or 1,2 or 1,3 were used. 1,1 means to use individual words only, 1,2 means to use individual words and bigrams, 1,3 means to use individual words as well as bigrams as well as trigrams.

5. Make Lower Case – must the process convert all text to lowercase? True means convert, False means do NOT convert to lowercase.

6. Analyzer – an analyzer value of word was used to enable the Token Pattern parameter. There are other value for this parameter but they were not used here.

7. Token Pattern – Two token patterns were used with slightly different results. They are both slightly different variations on the theme “keep all words, but not numeric or partial numeric words”.

8. Max features – this limited the number of features in the dictionary – and this is primarily necessary due to “Memory Errors” when running with large datasets containing large dictionaries.

All-in-all, for the purposes of this study, 11 iterations of this process was run with different parameter combinations as per the description above. The table below shows the input parameter combinations.

In addition, the following 3 classifiers were run, per DTM:

1. Multinomial Naïve Bayes

2. Linear SVC (aka SVM)

3. Random Forest

For each classifier process, (assuming T+T), a confidence matrix was created and the accuracy was determined. Then, the best accuracy together with the model that achieved that accuracy was derived.

In total, per label (Veracity, Binary, AuthorType, Sentiment)

* 11 iterations were run
* 3 classifiers were run

Meaning there are 33 vectorizer/classifier combinations to choose from per label.

## Model 1: Veracity Prediction and Accuracy

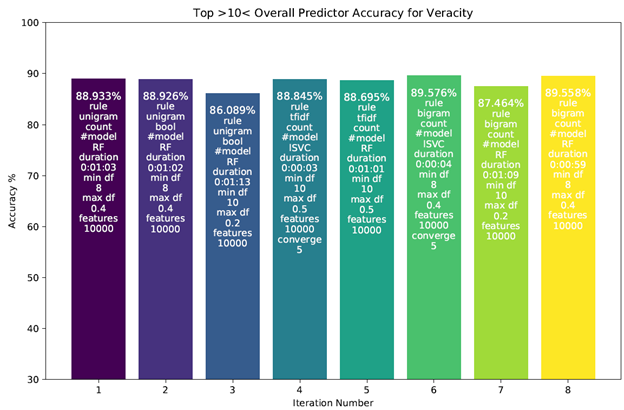
For each of the 8 iterations of the Veracity Modeling process, with each iteration having a distinct word vector, 3 classifiers were trained and tested to determine the prediction accuracy per model per label. And then the most accurate model was chosen.

The classifiers are:

1. A Multinomial NB model using the test-and-train set to predict accuracy

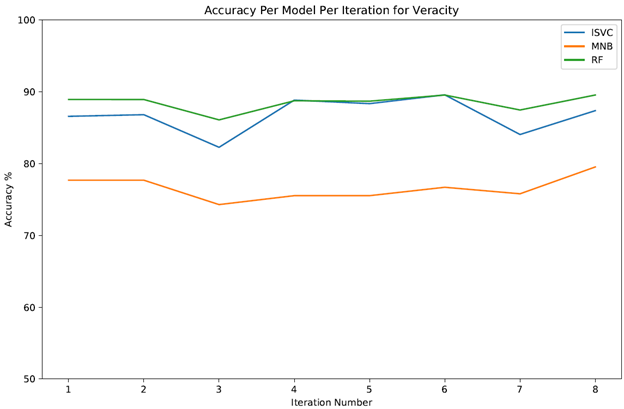
2. A linear SVC model using the test-and-train set to predict accuracy

3. A Random Forest model using the test-and-train set to predict accuracy

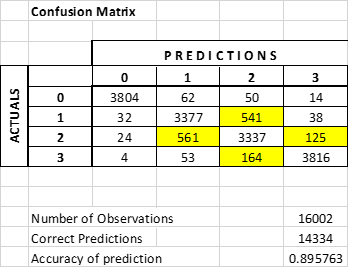


Looking at the results of this process where the accuracy of the best model is shown per iteration, it is observed that the most accurate model is the iteration 6 linear SVC process. This is the vectorizer/model set that was chosen and used in the composite modeling process.

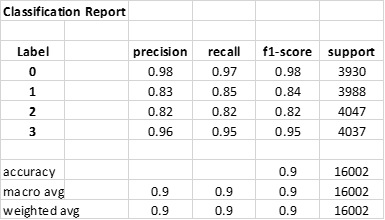
The chart below shows the accuracy results by modeling algorithm f(linear SVC, Multinomial NB and Random Forest) for each iteration where the Veracity accuracy is predicted. As can be seen, the RF model is generally more accurate than either the linear SVC or the Multinomial NB model. What is unusual here is that the linear SVC is the most accurate model.



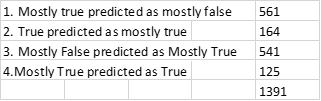
For the linear SVC classifier selected as the most accurate model, the Confusion Matrix is shown below:



And the Classification Report is shown here:

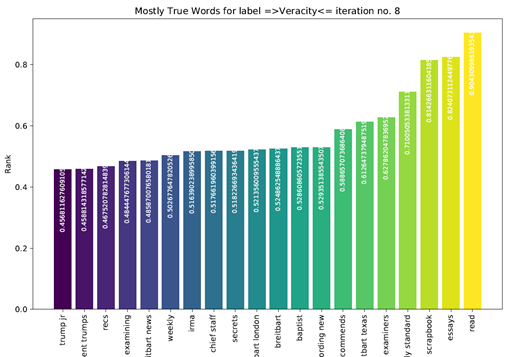


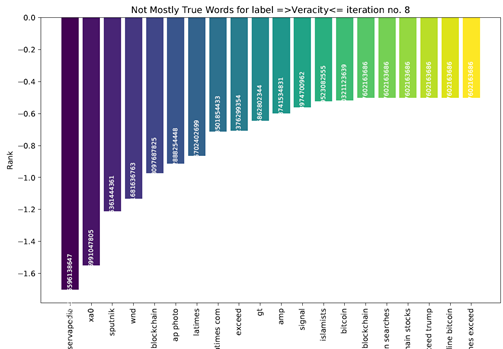
What is obvious from this model is that there is a definite opportunity to look at words that are classified as follows:



An improvement with these classifications would increase the accuracy percentage by about 8%.

The 2 charts shown below have words derived from the rule “bigram\_count\_3”, and come from the Veracity set with the top20 Mostly True and the bottom 20 Mostly True words for each label shown. Other Most Important Word Charts are available, but are not shown here.





Lastly, the WordCloud shown here was also generated as part of the Veracity modeling process using words from the rule “bigram count 3” vectorizer.



## Model 2: Binary Prediction and Accuracy

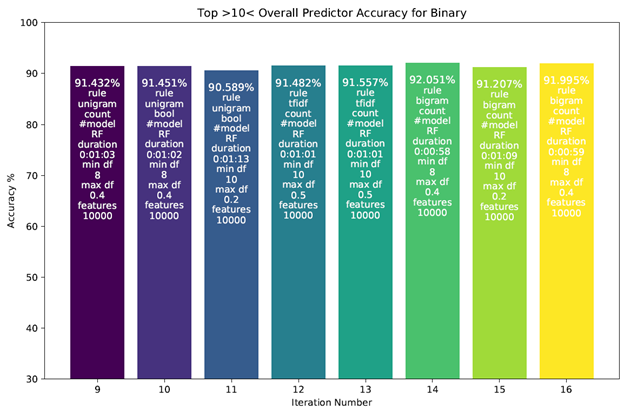
For each of the 8 iterations of the Binary Modeling process, with each iteration having a distinct word vector, 3 classifiers were trained and tested to determine the prediction accuracy per model per label. And then the most accurate model was chosen.

The classifiers are:

1. A Multinomial NB model using the test-and-train set to predict accuracy

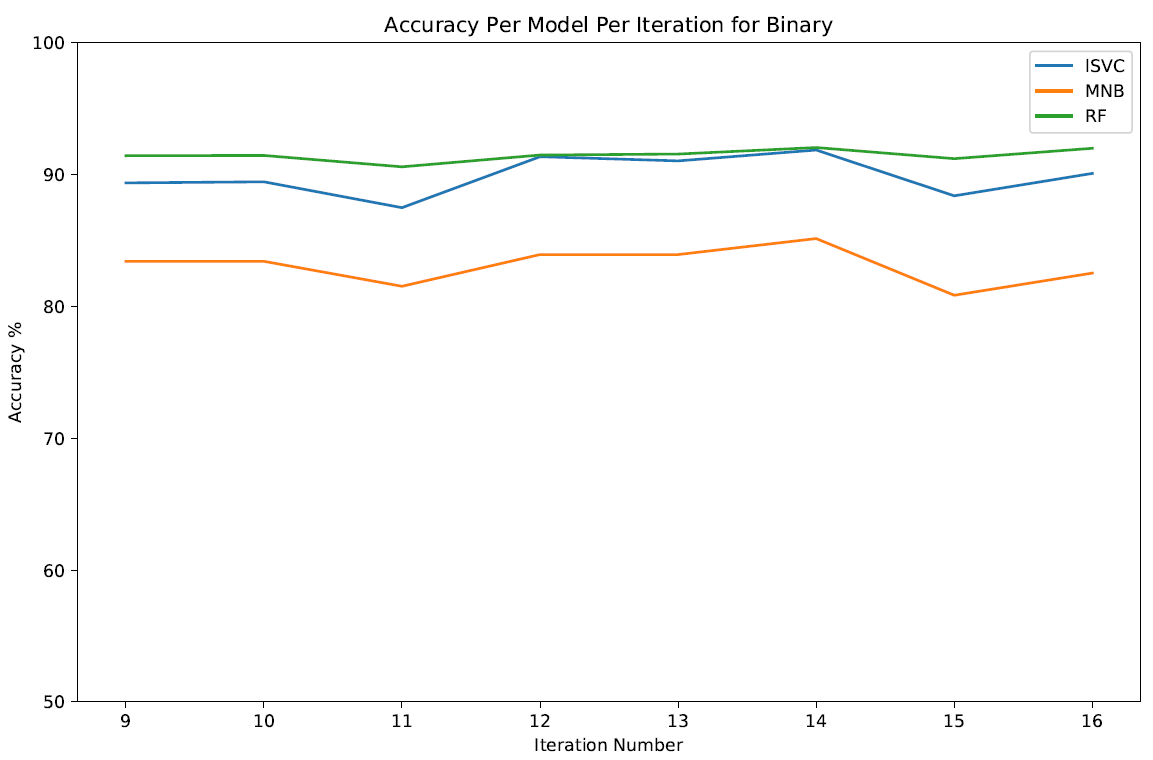
2. A linear SVC model using the test-and-train set to predict accuracy

3. A Random Forest model using the test-and-train set to predict accuracy

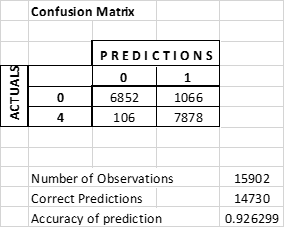


Looking at the results of this process where the accuracy of the best model is shown per iteration, it is observed that the most accurate model is the iteration 6 RF process (in this case #14 is shown due to the iterative counting method used). This is the vectorizer/model set that was chosen and used in the composite modeling process.

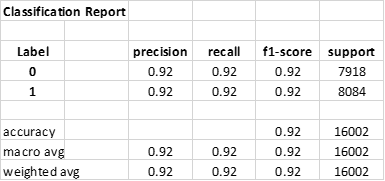
The chart below shows the accuracy results by modeling algorithm f(linear SVC, Multinomial NB and Random Forest) for each iteration where the Binary label accuracy is predicted. As can be seen, the RF model is generally more accurate than either the linear SVC or the Multinomial NB model.



For the RF classifier that was selected as the most accurate model, the Confusion Matrix is shown below:



And the Classification Report is shown here:

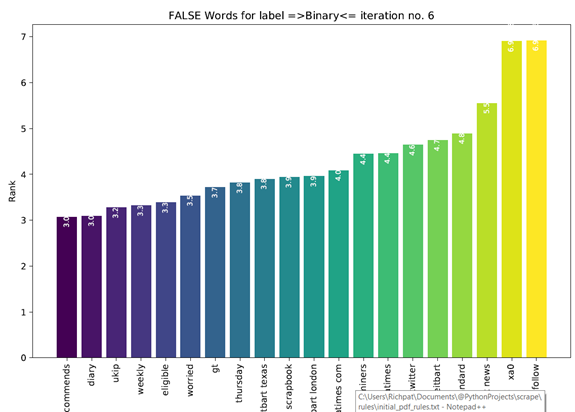


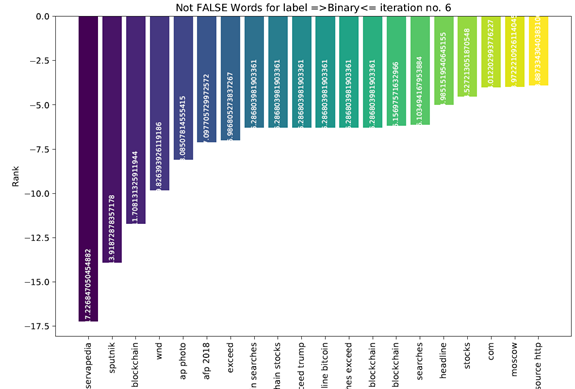
What is obvious from this model is that there is a definite opportunity to look at the words that are incorrectly classified.

· 1066 negative phrases are predicted as positive

· 106 positive phrases are predicted as negative.

The 2 charts shown below have words derived from the rule “bigram\_count\_1”, and come from the Binary set with the top20 False and the bottom 20 Not False words for each label shown. Other Most Important Word Charts are available, but are not shown here.





Lastly, the WordCloud shown here was also generated as part of the Binary vectarization process using words from the rule “bigram count 1” vectorizer.



## Model 3: Author Type Prediction and Accuracy

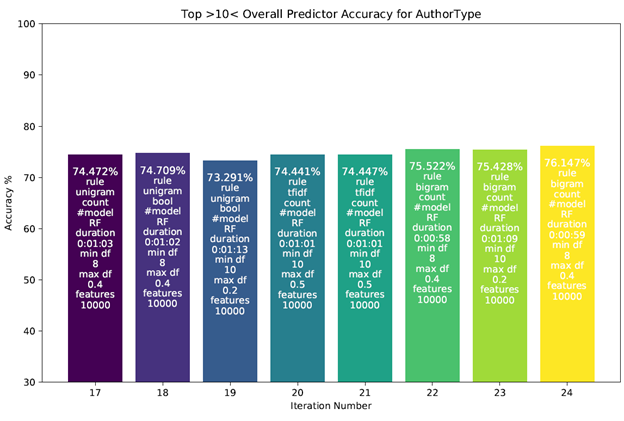
For each of the 8 iterations of the Author Type Modeling process, with each iteration having a distinct word vector, 3 classifiers were trained and tested to determine the prediction accuracy per model per label. And then the most accurate model was chosen.

The classifiers are:

1. A Multinomial NB model using the test-and-train set to predict accuracy

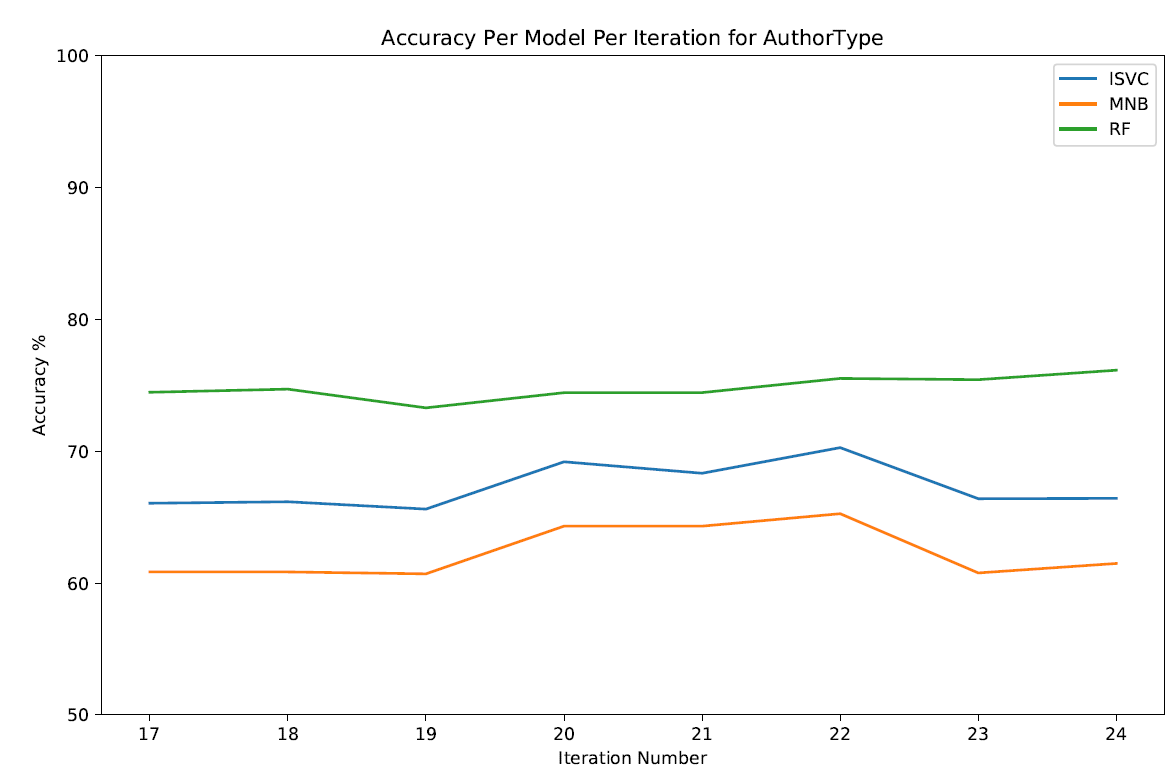
2. A linear SVC model using the test-and-train set to predict accuracy

3. A Random Forest model using the test-and-train set to predict accuracy

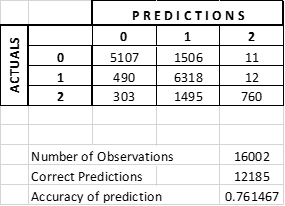


Looking at the overall results of this process where the accuracy of the best model is shown per iteration, it is observed that the most accurate model is the iteration 8 (in this case #24 is shown due to the iterative counting method used) RF process. This is the vectorizer/model set that was chosen and used in the composite modeling process.

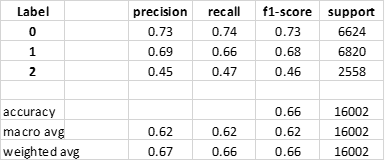
The chart below shows the accuracy results by modeling algorithm f(linear SVC, Multinomial NB and Random Forest) for each iteration where the Author Type accuracy is predicted. As can be seen, the RF model is generally more accurate than either the linear SVC or the Multinomial NB model.



For the RF classifier that was selected as the most accurate model, the Confusion Matrix is shown below:



And the Classification Report is shown here:



What is obvious from this model is that there is a definite opportunity to look at words that are classified as follows:

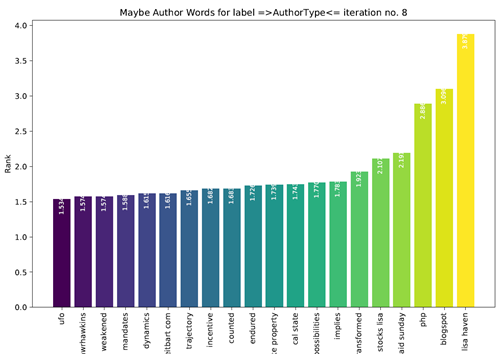
1. Unknown as Maybe Unknown – 1506

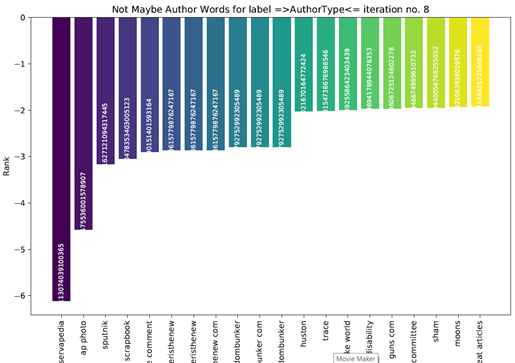
2. Maybe Unknown as Unknown – 490

3. Known Author as Maybe Unknown 1495

Improving the predictions for these incorrect ones would increase the accuracy by approximately 20%.

The 2 charts shown below have words derived from the rule “bigram\_count\_3”, and come from the AuthorType set with the top20 Maybe and the bottom 20 Not Maybe words for each label shown. Other Most Important Word Charts are available, but are not shown here.



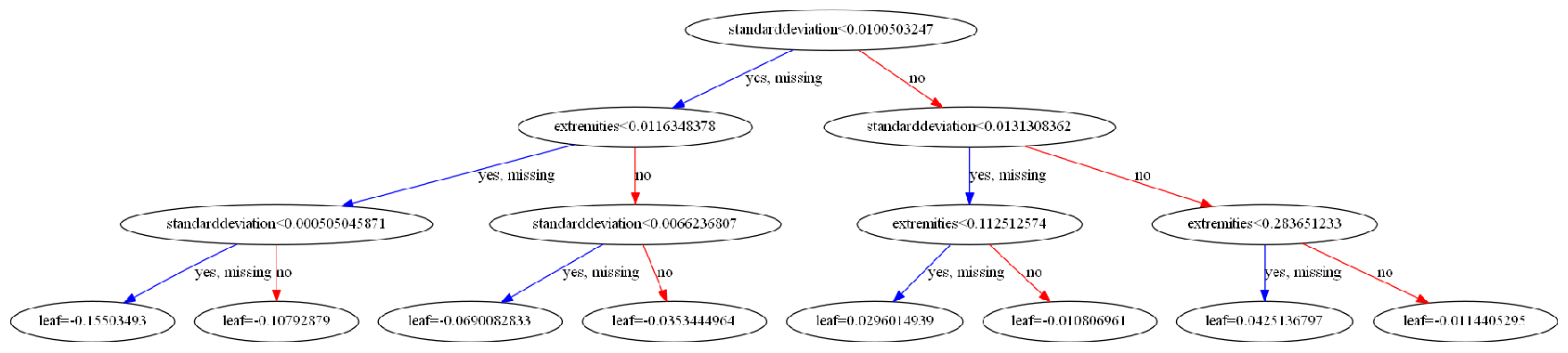


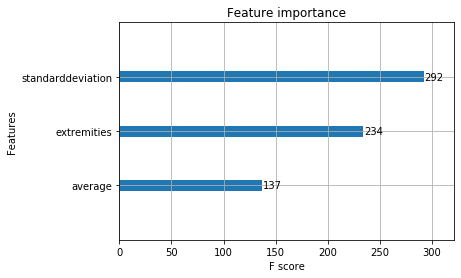
Lastly, the WordCloud shown here was generated as part of the Author Type vectarization process using words from the rule “bigram count 3” vectorizer.



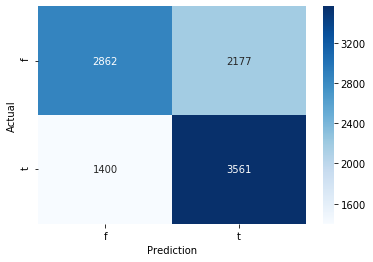
## Model 4: Sentiment-based Prediction and Accuracy

The sentiment-based model used the AFINN sentiment lexicon to assign every word that was vectorized an integer score for sentiment between -5 and 5 inclusive. Three metrics were then taken from each article: the average word sentiment, the standard deviation of word sentiment, and the number of “extreme” (3 through 5 and -3 through -5) words divided by document length. These three metrics were normalized using SKLearn’s MinMaxScaler and fed into a machine learning algorithm. Of the algorithms tested, XGBoost outperformed RandomForest, neural networks, C.45-based decision trees, and all four kernels of SVM offered in SKLearn. The XGBoost tree and its feature importances are shown below.

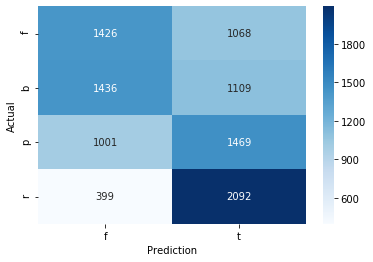




When trained and tested only on the Reliable and Fake labels, the testing accuracy of this model was 77%. However, when equal numbers of all four labels were included in the training and testing sets, then discretized to True and False, the testing accuracy was only 64.23%. This model yielded the confusion matrices below. In both cases, the training accuracy was very similar, suggesting minimal overfitting.



This second confusion matrix shows that although Fake and Bias (f and b) and Political and Reliable (p and r) were discretized together to form False and True respectively, the accuracy of this binary model was not the same for each of the four labels. It is clear that the model significantly overpredicted the True label.



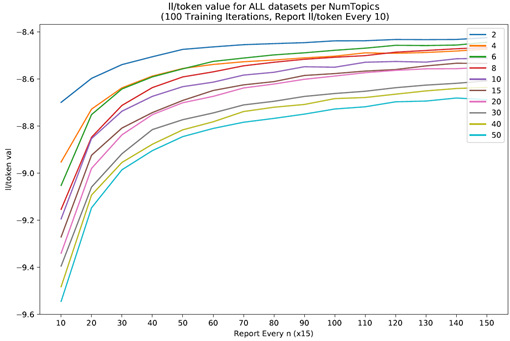
## Model 5: Mallet Topic Modeling

The topic modelling test had two objectives:

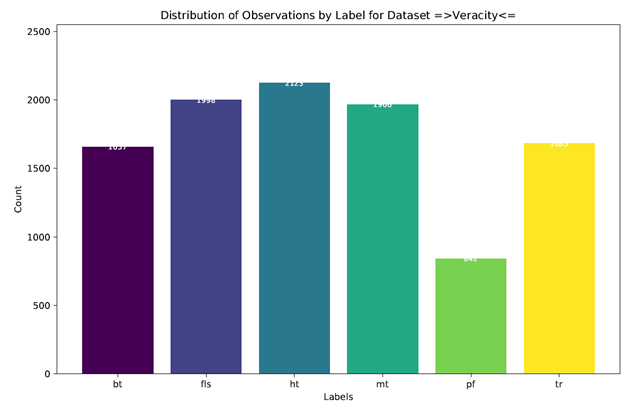
* Objective 1: Test the “Liar Liar” dataset to see whether it would be a useful dataset to add to our “known lies” input
* Objective 2: Determine whether the Mallet Topic Modeling routine could be utilized to extract a meaningful topic set for feature creation for the “Is It Real News?” engine.

The first step was to determine an appropriate number of topics to use. This was achieved by iterating thru Mallet with an increasing number of topics to determine what could be considered an appropriate number of topics.

As can be seen from the chart below, the topic range chosen was between 2 and 50. For each topic range, a topic generation was run with 150 iteration and results reporting every 10. This gave an 15 output/efficiency tokens (mallet ll/token) per iteration and this is what is charted below. The most efficient output is shown at the end of the 50 topics model where the ll/token is the lowest. This is the model that was used for subsequent analysis.



Given 10243 phrases on the “Liar Liar” dataset, distributed as follows:



Bt=Basically True

Fls=False

Ht = Hardly true

Mt = Mostly True

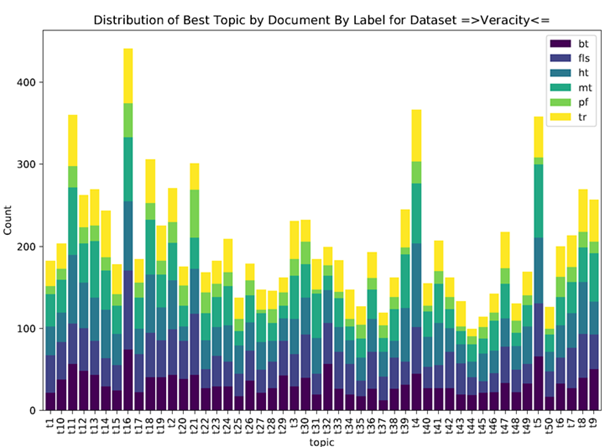
Pf=Pants On Fire

Tr=True

Together with the extracted 1000 topic words (20 words per topic spread across 50 topics)



And a chart showing the distribution of most important topic per document (given that each document has a “relevance to topic” factor) separated by document label



This test produced a satisfactory set of output where it could be determined that:

1. The “Liar Liar” dataset was not usable for our purposes due to the shortness of phrases included in the dataset
2. The topic modeling output required manual analysis and derivation work which made it unusable for our purposes.

## **Model 6: The Composite Model**

## The Composite Model was built to generate a predictable feature set that would be used for subsequent real or fake news prediction.

## 

## The first piece of the Composite Model is the model setup:

## Load the pickled vectorizer, the pickled model and the pickled confusion and classification scores for the Veracity process.

## Load the pickled vectorizer, the pickled model and the pickled confusion and classification scores for the Binary process.

## Load the pickled vectorizer, the pickled model and the pickled confusion and classification scores for the Author Type process.

## 

## The second piece of the Composite Model is the model execution:

## Identify a known set of fake and real text that will be used to train the composite model

## Run the composite model with this data and produce the initial feature set

## Add the sentime score results to the feature set

## Add the word specific info to the feature set (Word count, syllable count and matching words in dictionary count)

## 

## This execution produces a feature set in a form that will be used by the Supervised Learning and Unsupervised Learning models to generate the predictions.

## 

## The feature set is a table that contains, per observation, a list of features generated from the Composite Model prediction and preparation steps as described above:

## - Has headers

## - 31 columns

## - 6 for the veracity mode (False, Mostly False, Mostly True, True)

## o Accuracy of model

## o Prediction (for veracity this is one of 0, 1, 2 or 3)

## o Precision

## o Recall

## o F1-score

## o Support

## - 6 for binary model (false, true)

## o Accuracy of model

## o Prediction (for binary is one of 0 or 1)

## o Precision

## o Recall

## o F1-score

## o Support

## - 6 for AuthorType model (unknown, maybe known, known)

## o Accuracy of model

## o Prediction (for veracity this is one of 0, 1 or 2)

## o Precision

## o Recall

## o F1-score

## o Support

## - 6 for Sentiment model (false, true) – all values are 0 in my file

## o Accuracy of model

## o Prediction

## o Precision

## o Recall

## o F1-score

## o Support

## - Wordcount for the body of text for this observation

## - Syllable counts for the body of text for this observation

## - Count of words in the text that are in the dictionary

## - Original Labels

## o Veracity

## o Binary

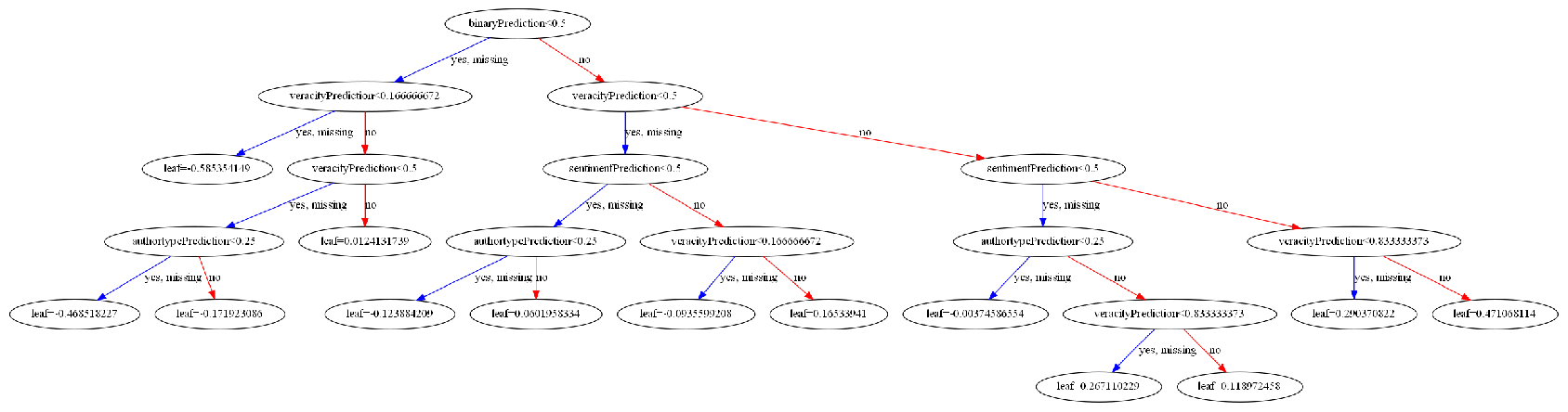
## o Authortype

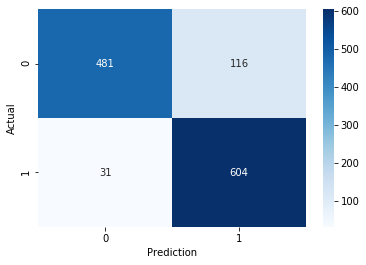
## o Sentiment

## Model 7: Supervised Composite Model For Prediction

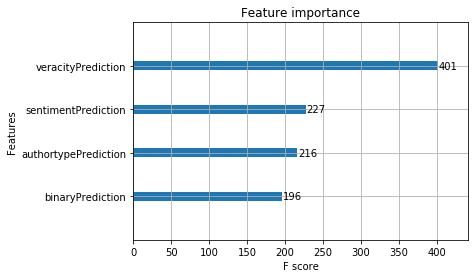
A supervised composite model was built to predict whether a text document is from a real or fake news source. A final testing set of 4928 articles was randomly selected in equal parts from the 4 ordinal labels. This set was fed through each of the 4 pre-trained individual models and the predictions given by each were compiled into an array for inputs for use in the composite model.

Taking the predictions/classifications of each of the 4 models as inputs, the composite model proved better than any of the individual models. Each input was normalized using SKLearn’s MinMaxScaler. The composite model was then 5-fold cross-validated on the data. For use as the composite model algorithm, XGBoost proved superior to RandomForest, neural networks, C.45-based decision trees, and all kernels of SVMs. The final testing accuracy achieved was 88% with an almost identical training accuracy. The tree and confusion matrix of this model can be seen below.





In a similar trend to that observed in the individual models, the supervised machine learning on the composite model moderately but significantly overpredicted the True (1) label. The “weight” feature importance of each individual model’s prediction as an input for the supervised learning is shown below.



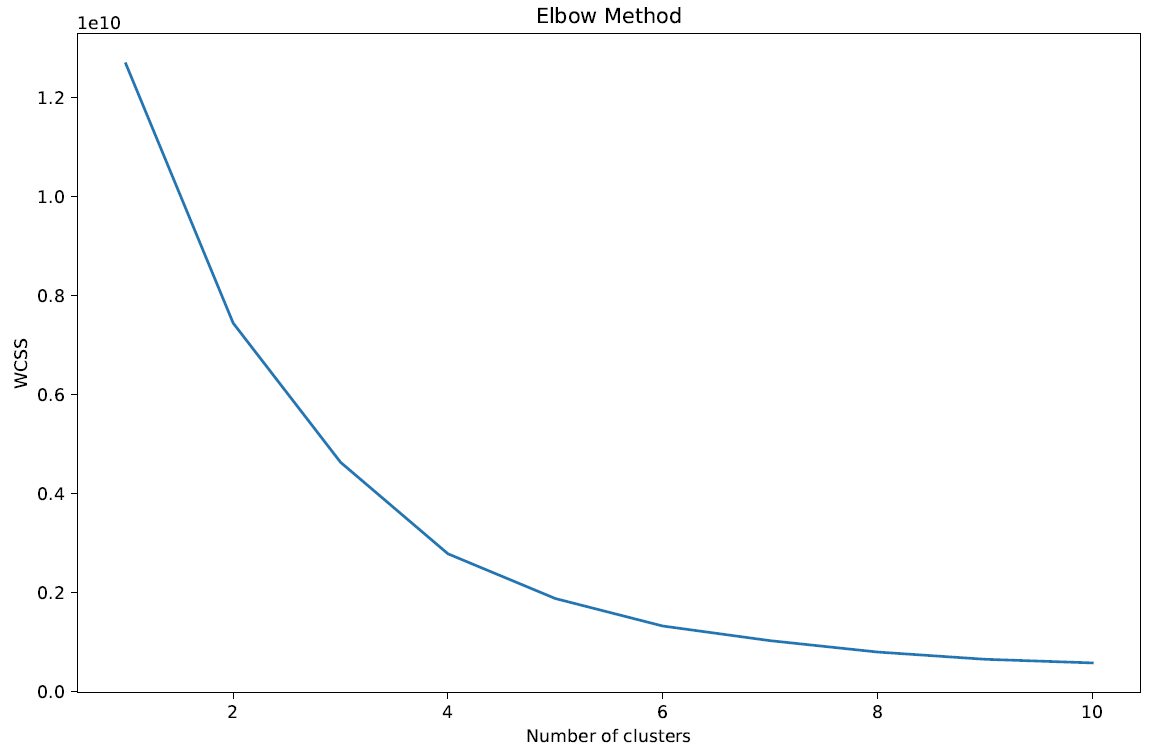
## Model 8: Unsupervised Composite Model For Scoring

Using the Feature Set that was output from Model 6, a segmentation model was built to determine whether

Real News?” prediction process.

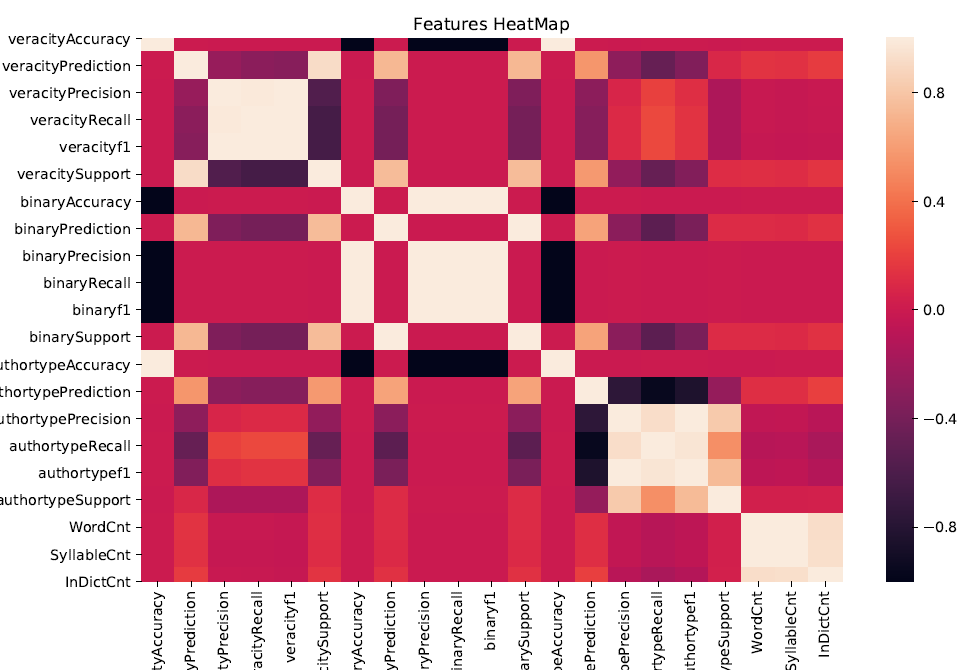
To build this segmentation model, a 5000 document text set was used.

First step was to check the optimal number of segments using an elbow plot.

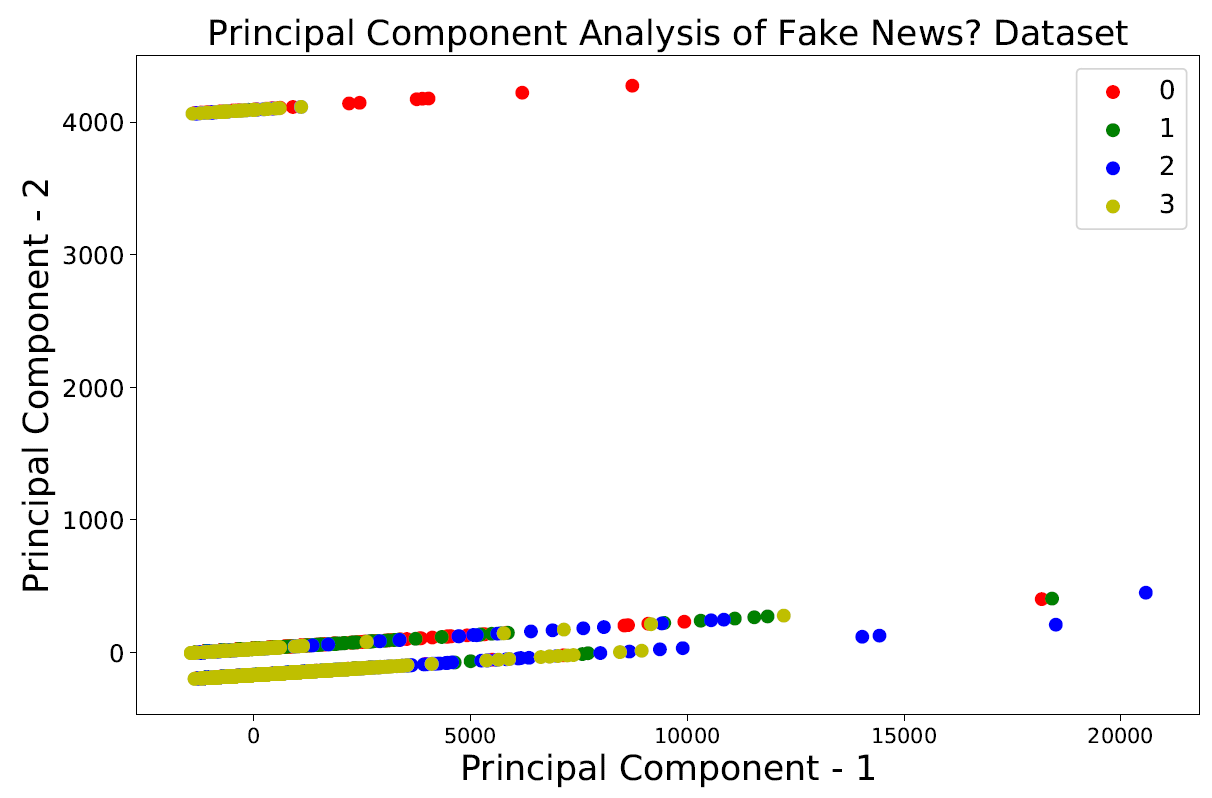


Visually we can tell that 4 is the optimal number of segments that should be used.

Secondly, a correlation heatmap was drawn using features in the featureset to tell which attributes are meaningful and which are not. Note that for the purposes of this exercise, all features were used.



Thirdly, a Principal Component Analysis was conducted to identify the most important features.



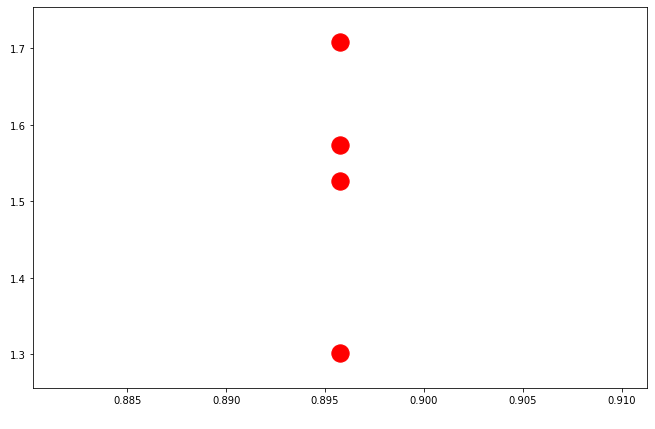
Principal Component 1 corresponds to feature Syllable Count.

Principal Component 2 corresponds to feature Author Type Support.

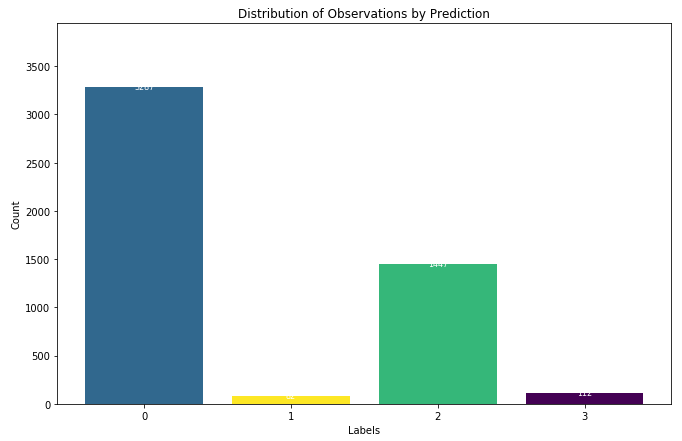
Explained variation for PC1 is 0.83533224

Explained variation for PC2 is 0.15739164

Then k-means clustering was run to determine where the centers are per cluster.



Lastly, a set of predictions were run using the clustering routine, and distribution is given below.



The last step requires a manual check to determine how each of these groups would be labeled.

# Results

Component Model Accuracy vs. Speed Tradeoff

Albeit that the Random Forest model appears to deliver consistently better results than either the linear SVC or Multinomial NB models, it should be taken into consideration that Random Forest model creation is much slower than either of the other two.

Generally, a Random Forest model took approximately 10 minutes to build where a Multinomial NB or linear SVC model took only a few seconds. This is a significant difference.

Composite Model Supervised

In a Supervised Learning approach, a model was trained using the predictions of the 4 individual component models to perform binary classification, predicting whether an article was real or fake. This model outperformed each individual model for a testing accuracy of 88%.

Composite Model Unsupervised

Using an Unsupervised Learning approach, it was learned that it is possible to build a Composite model that can automatically predict which group (segment) a document belongs to based on feature creation using individual component models that are trained as part of an earlier process.

What is not possible, without external input, is to tell whether a group (segment) means true or false or some other permutation.

# Conclusions

Creating many smaller models of varying approaches, this study was able to obtain a high degree of accuracy identifying the reliability of an article using its text. Some interesting trends were discovered throughout, such as the significance of highly-charged sentiment in identifying fake news. Another revelation was the importance of checking whether an article has a specific author listed and what form that author takes.

The final achievement of this study was a diverse and robust composite model, the *“Is It Real News?” Engine*. Comprising this construct were a robust fake news predictor and a thorough unsupervised model showing how a given article fits in with news across the spectrum of veracity. With a few more improvements, this engine will be ready for distribution via browser applet or web service to allow the public better awareness of the information they allow to shape their opinions and decisions.

Despite these promising results, there is still much more work to be done in this field. There are endless permutations of news data ready to thwart any predictor, and the entire world’s news corpus can never be summarized completely. This study merely lays the groundwork to effectively combat the grievous social ill that is false information.