Fake News Detection

Project Report : rohanb — Rohan Bansal
saiprahp — Sai Prahladh Padmanabhan

Introduction:

Motivation:

Content moderation has been a challenging task for major corporations and news agencies as the authenticity of news is of paramount importance. With that consideration in mind, we wanted to apply machine learning tools and algorithms to tackle the issue of fake news, by observing recurring patterns that can be found in articles which can be deemed fake. These patterns are then compared with authentic news to determine the criteria used to classify an article as fake or authentic.

Methodology:

In our ML pipeline we have first collected data deemed to be true from reputed sources such as the New York Times and the Guardian for over a year. Followed by cleaning and formatting the mixed data sets. We then took our dataset and applied three statistical methods tasks, mainly Logistic Regression, Naïve Bayes and a Random Forest implementation to see which would be more suitable for our classification task.

Project Overview:

The implementation of our project can be broken down into the following steps:-

1. Data Collection.
2. Visualizing the data.
3. Model Selection
4. Cross Validation and hyperparameter tuning.
5. Model Evaluation.
6. Broad comparison between selected models.

Data Collection and Visualisation:

For our data collection we have scraped data from two prominent newspapers mainly the New York Times and The Guardian. These newspapers provide API keys and those were used to extract data from the 2016 news cycle. To try various types of data we collected both political news and apolitical news to see how that would affect the evaluations of our models.

For our fake news dataset, we used an unstructured collection of fake news articles on Kaggle. We cleaned and merged this dataset with our real news data set to obtain a combined collection of 1910 articles.
We have maintained the

To visualise the results, we have created scatter plots and try to understand the most commonly occurring words in authentic and fake news. We attribute this correlation to the fact that many articles belonging to either categories would have similar topics and sets of words.

For instance the use of words like Trump, Hillary cannot automatically classify an article as real or fake and both types of news could use them.

Model Selection:

Amongst an eclectic set of algorithms, we considered which would be ideal for a natural language processing task.

Another consideration was to understand how different models would perform given that our results had a significant amount of correlation.

We wanted to understand how a Naïve Bayes model would perform since it assumes independence. Next, we wanted to see what would happen in a model where such a barrier didn't exist and selected a model quite widely used for NLP related tasks: Logistic Regression and would give a probability of a article being fake or real.

Lastly, to overcome the problems our large dimensional data set we opted to implement random forests.

Cross Validation & Hyperparameter Tuning

For the Naïve Bayes implementation, the main step was to create a bag of words model which would encompass the most informative features correctly and then provide words common to both the classes to the Naïve Bayes classifier to avoid the case of missing values. The model was trained on a bag of about 30000 words to get a satisfiable accuracy.

For the Logistic Regression and the Random Forests model, we have used a 5-fold cross validation to determine the optimal hyperparameter values.

For Logistic Regression we have considered different regularization parameters such as L1 norm, L2 Norm and No Regularization. We finally selected L2 Norm and a regularization coefficient value of 5, which is relatively high. We attributed this high coefficient to the prevent overfitting for this large number of parameters in our bag of words model.

Finally for Random Forests, we have considered the maximum depth of the random forest , the number of estimators was tuned to find the optimal number of trees in a given random forest. We also considered Gini impurity as the splitting criterion for our decision trees. It is very easy to overfit trees, we selected parameters such that the depth is not too high and the number of estimators in each tree are reasonably selected.

Model Evaluation.

All of the chosen models performed with an accuracy of at least 80% and above.
The Naïve Bayes Model had an accuracy of 79.83% where it had a bag of words containing words common to real and fake articles along with their counts in each of the class. The Bayes Model assumes conditional independence amongst the features and operates based on this assumption. Hence it ignores any correlation among the words. For example if the word 'Trump' occurs x number of times, there is a very high possibility that the word 'Donald' or the word 'president' also occurs in a large number.

The Naïve Bayes Model performed poorly when there was an imbalance in the bag of words owing to real and fake classes, for example, if the real dataset had articles from the sports category in large number and the fake dataset did not, then the classifier performs poorly.

Logistic Regression Classifier:

The Logistic Regression Classifier took a TF-IDF vectorized input for performing the classification. TF- IDF represents Term Frequency and Inverse Document frequency. The term frequency is the number of times a word occurs in a given article and IDF refers to the frequency of that word in all the articles of a given label. The vectorized input was used to get prediction probability of each class and the class with higher probability was assigned as the label.
Upon performing a 5 fold cross validation on our dataset, we got a cross validation accuracy of 90.77%. with a regularization term of value 5.
The test accuracy for this model was 88.88%

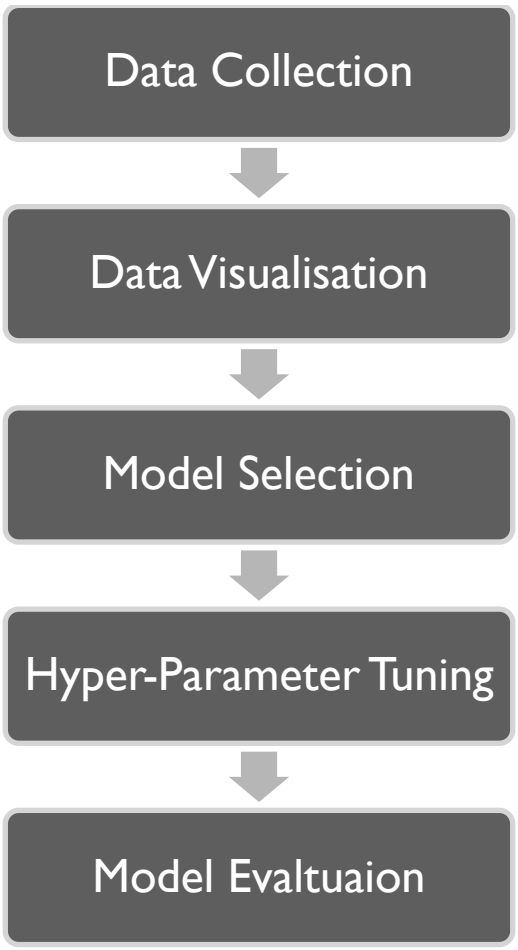
Random Forest Classifier:
This ensemble method is similar to bagging but it operates on a subset of the given features. The criteria chosen for this classifier was gini impurity as it gave a better classification accuracy as compared to information gain. The depth of trees was set to 5 in order to prevent overfitting. The cross validation accuracy was 89.02% and the test accuracy was 88.68%.

Inference:

- Amongst the three models, highest accuracy was achieved by logistic regression model and the second highest accuracy was achieved by the Random Forests model.
- Scope for improvement in the Naïve Bayes model would be to enable the model to handle missing data for higher accuracy.
- The models were tested on data acquired from the headline of the article and the body of the article and evidently so the models performed better with the latter, because the data held more semantic significance.

Appendix

Project implementation pipeline.



Visualization :

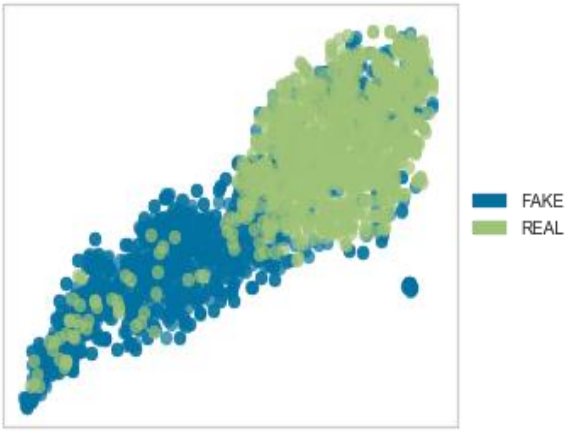
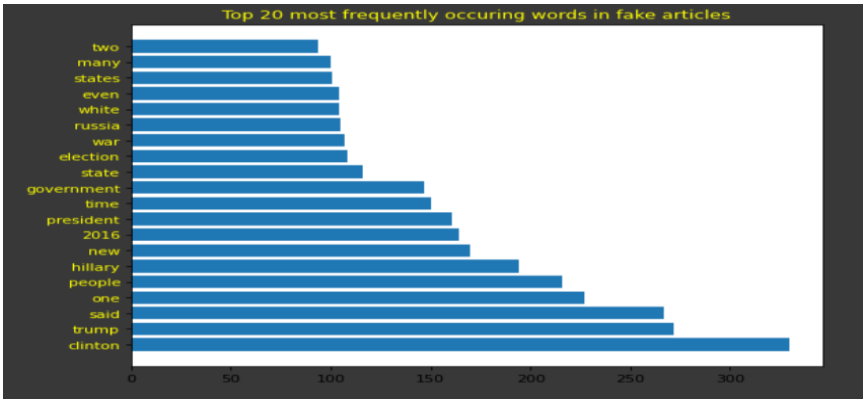
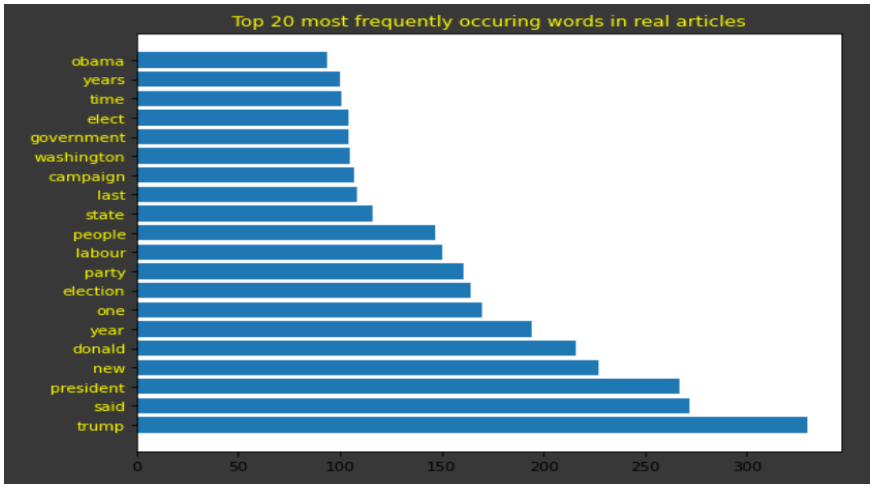


Fig.Vectorized format visualization of bag of words





Algorithms employed and the math behind them.

1.Naïve Bayes Classifier:

The Naive Bayes is a very famous choice when it comes to bag of words models because of their simple complexity and well performing metrics. The algorithm is built on the core assumption that the features under consideration can be treated as independent and any correlation is overlooked.
For the bag of words, we have the words which are common to both real and fake news articles along with their counts in each category. The probability of each feature in a given category is as follows.

$$P(X_i) = \frac{n(X_i)}{\sum_{i=1}^{N} n(X_i)}$$

Where n(Xi) is the count of a given feature in the given category.
Thus we have the probabilities of all the features in a category. In this implementation, we calculate the log probability of any article belonging to either of the category. Depending on the magnitude of the probability, we assign the label. The probability is calculated as follows.

$$P(Y|X) = log(P(Y) * \prod_{i}^{N} P(X_i)^{n_i})$$

P(Y|X) = Conditional log probability of a category given features X.
P(Y) = Prior of the category.
P(Xi) = Probability of the features in a given category (Probabilities of features in either fake or real articles)
ni = Frequency of a given feature in an article under consideration.
Depending on the value of P(Y|X) the category label is assigned to the article.

2.Logistic Regression :

In the implementation, we have performed binary logistic regression in order to classify the articles.
Logistic regression measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic/sigmoid function.
The loss function chosen here is the cross entropy loss function.

The cross entropy loss function is given as :

$$\log p(y|x) \;=\; \log\left[\hat{y}^{y}\,(1-\hat{y})^{1-y}\right]$$

Here y is the prediction which is given by sigmoid(w'*X) where w' represents the transpose of the weights. The classifier thus predicts the probability of each class for a given article and assigns the label of the greater probability class to the article. The usage of sigmoid function helps us to squish the value of the prediction between 0 and 1.
The logistic regression expression does not have a closed form solution for finding the MLE, so we use gradient descent. In the expression below, L denotes the loss function

$$\nabla_{\theta} L(f(x;\theta),y)) \;=\; \begin{bmatrix} \frac{\partial}{\partial w_1} L(f(x;\theta),y) \\ \frac{\partial}{\partial w_2} L(f(x;\theta),y) \\ \vdots \\ \frac{\partial}{\partial w_n} L(f(x;\theta),y) \end{bmatrix}$$

The final equation for the gradient descent step is as follows:

$$\frac{\partial L_{CE}(w,b)}{\partial w_j} \;=\; [\sigma(w\cdot x+b)-y]x_j$$

3. <u>Random Forests</u>:

Random Forests is an ensemble method where multiple decision trees are used to perform a classification task. In this algorithm, the classification is performed on a subset of features. This is the main difference between Bagging and Random Forests. The feature on which the split occurs in the decision trees can be found by calculating information gain or Gini Impurity values.

Information Gain: The information gain of a feature to be split on is calculated as:

Gain = H(Y) – H(Y|X)

Where H(Y) is the entropy of the target variable and H(Y|X) is the conditional entropy given a feature based on which the split is supposed to take place.

Entropy is given as:

$$H(Y|X) = \sum_{i=1}^{C} fi * log(1/fi)$$

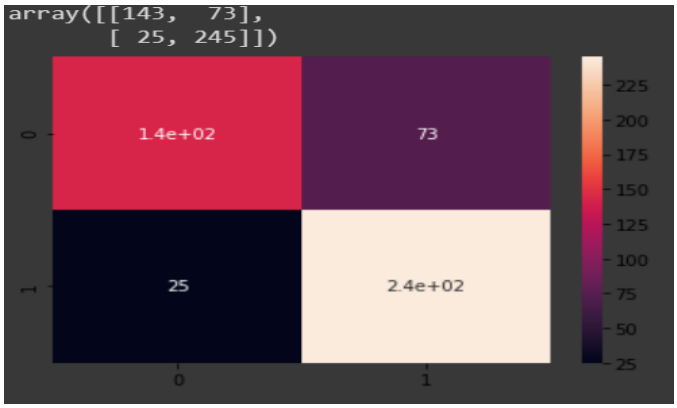
fi is the frequency of a feature at a node and C is the number of unique features.

Similarly, gini impurity can be calculated from the given expression:

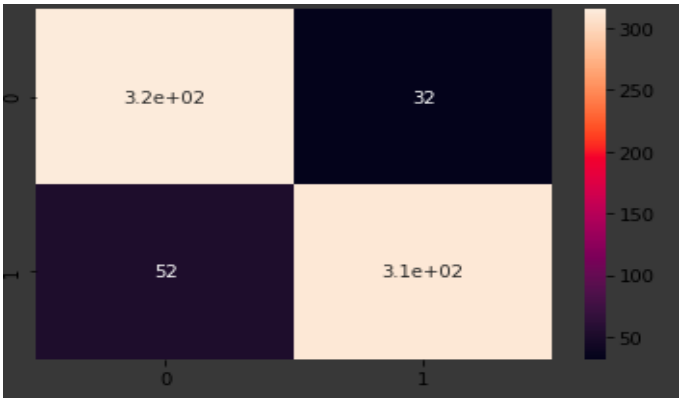
$$\sum_{i=1}^{C} fi * (1 - fi)$$

Results: Confusion matrices for test data

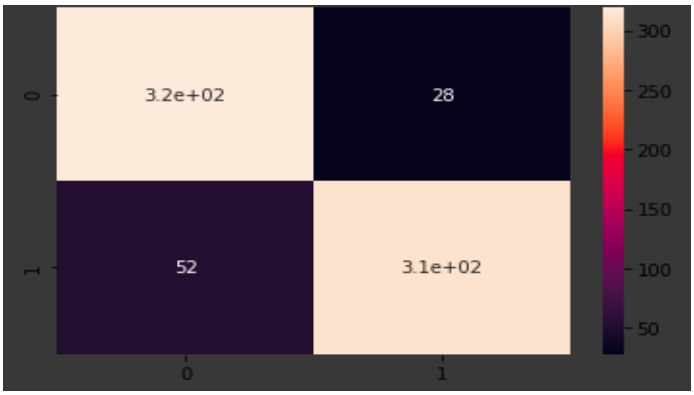
Naïve Bayes:



Logistic Regression:



Random Forests:



Final results for Test Data:

Model	Test Accuracy(%)
Naïve Bayes	79.83
Random Forests	88.88
Logistic Regression	88.68