PROJECT TO IDENTIFY AND CLASSIFY FAKE NEWS

Ritvik Dhupkar

Introduction

The purpose of the project is to apply NLP techniques to identify and classify fake news into Fake and Non-Fake. Leading up to the 2016 Election, fake news thrived on social media, generating millions of clicks and providing ad revenues to hosting websites. One Analysis found that top 20 fake news outperformed top 20 real news in terms of revenue and comments, shares and reactions. The top election news story, "Pope Francis Shocks World, Endorses Donald Trump for President, Releases Statement," was fake (Silverman,2016).

There has been debate in the media on whether the proliferation of fake news during elections influenced election results. A study by two economists found that fake news articles supporting Trump were shared 30 million times on Facebook, while Fake news articles supporting Hillary were shared 8 million times. The study concluded that fake news played a role in reinforcing existing biases of people, rather than switching them over to the other side (Alcott and Glentzkow,2017).

Since then, Fake news sites like google and Facebook have stated that they have implemented algorithms to squeeze the revenues of fake news websites. Facebook has updated its UI to allow users to allow users to flag articles as fake.

Data

Fake news dataset was downloaded from a number of GitHub repositories and compiled. The compiled data-set consisted of 10000 articles classified as fake and non-fake. The data- set was generated on various repositories by scraping websites marked as 'b-s' by the b-s detector. I believe that a classifier tailored to specific domains would better able to classify fake news, as some domains – such as politics would have a higher frequency of fake news than others.

The nature of Fake News is very diverse. An article (falsely) that 'Trump von the popular vote' would be very different from a satirical piece on Trump. A section articles would not necessarily be factually incorrect but would be highly biased towards one point of view. These articles substantiate a particular point of view with data, while withholding counter-arguments to the same. These articles are classified as biased

Possible Features That Could be Included in the Model-

Misspelled Words & punctuations- Professionally written articles by newspapers and journals unlikely to have any spelling or grammar errors.

Strong Sentiments and Punctuations-Real news articles are more likely to use neutral words, and a rational evidence-based argument. Fake news articles are likely to use strong words and exaggerated sentiment. Capitalized words, multiple exclamation marks and punctuation marks are typical of fake news articles ex- (YOU WON'T BELIEVE THIS !!!). These features can probably be used as indicators to classify articles as fake/ non- fake news.

Article Domain names- There are existing lists of domains that have been classified as Fake News. One of these is the PunditFact Fake News Almanac. PunditFact is a project of the Tampa Bay Times and the Poynter Institute, dedicated to checking the accuracy of claims by pundits, columnists, bloggers, political analysts, the hosts and guests of talk shows, and other members of the media. If the domain of a particular article falls under the ones classified as 'Fake News', then the news article can immediately be classified as Fake news.

Article Title- The title of the articles for Fake/Real News can be a differentiating factor. Fake news titles have significantly larger number of punctuations, Use of more capital letters and adjectives. The analysis done for text in terms of sentiment and punctuations can be applied to the title. Fake news titles use significantly fewer stop-words and nouns, while using significantly more proper nouns and verb phrases (Horne, Shah 2016).

Classification Algorithms Used

Naïve Bayes Algorithm- The Naïve Bayes algorithm uses the Bayes theorem and makes the assumption that features are Independent. That is, the features are multivariate normal with the covariances =0. Applied to text classification, this means that the presence of a word in an article does not have any bearing on the presence of another word in the same article. An example of the algorithm followed by is given below –

1. Collect all words & other tokens that occur in a bag of words (Henceforth called bag1). Vocabulary – all distinct words and tokens in bag1 . Docsj – is the subset of all examples

For which the target values is Vj. J can take values(1,0). 1- fake news,0 – non-fake news. This gives - V1- fake news. V0 – non-fake news.

2. Calculate P(Vj) and P(Wk|Vj) as shown below.

   P(Vj) = docsj/bag1

3. Textj – Single document created by concatenating all members of docj. j can take values(0,1)
4. N- total number of words in Textj, including duplicates
5. Nk – no. of times Wk occurs in Textj.
6. P(wk|Vj)= (Nk+1)/(N+|Vocabulary|)
7. Multiply the individual probabilities to calculate the probability of an article being Fake/Non-Fake.

   For example in the sentence –"I hate the poor acting"

   P(Fake)=P(1)*P(I|1)*P(hated|1)*P(the|1)*P(poor|1)*P(acting|1)

   P(non-fake) = P(0)*P(I|0)*P(hated|0)*P(the|0)*P(poor|0)*P(acting|0)

The class having the maximum probability will be taken as the predicted class. Thresholds can be assigned, to alter this rule. For example, the algorithm can be asked to classify a news article as fake if the P(Fake) >0.1. The multinomial Naïve Bayes algorithm from the SKlearn library will be used. This uses Laplace smoothening, so that if a feature is not present in the training data, its probability is not assigned as zero; otherwise, the model performs poorly, suggesting certain events are highly unlikely or impossible. Although Naïve Bayes assumes independent features; for example, if an article contains 'Donald Trump' it will consider these two words as independent and multiply the probabilities that an article belongs to a certain class. This feature ought to prohibit the use of Naïve Bayes for text classification, however, it has been observed that Naïve Bayes performs well for text classification because the dependencies tend to cancel each out (Zhang, 2004).

Logistic Regression

In logistic Regression, each word x_i is multiplied by a weight w_i . The probability p(x) of an article being fake

- Let $p(X) = \frac{e^{\beta_0+\beta_1 x}}{1+ e^{\beta_0+\beta_1 x}}$, where p(x) = probability of x=1.
- $\log(p(x)/(1-p(x)) = \beta_0 + \beta_1 x$ Log odds can be regressed as a linear regression

Linear Discriminant Analysis and Quadratic Discriminant Analysis

LDA is similar to Naïve bayes except that it assumes that predictors and target have a multivariate normal distribution. An Assumption is made that the covariance matrix for each class is the same.

$$\Pr(\mathrm{Y} = \mathrm{k}|\mathrm{X} = \mathrm{x}) = \pi_k * \frac{\mathrm{f_k}(x)}{\Sigma\, \pi_l f_l\,(x)}$$

$$f(x) = \frac{1}{(2*\pi)^{0.5*p}\ |\Sigma|^{0.5}} * \exp(-0.5 * (x-\mu)^T * \Sigma^{-1} * (x-\mu))$$

A linear decision Boundary can be derived from this and is given below

$$\delta k(x) = x^T * \Sigma^{-1} * \mu_k - 0.5 * \mu_k^T * \Sigma^{-1} * \mu_k + log\pi_k \qquad \text{(ISLR p. 143)}$$

In QDA, a common covariance matrix is not assumed for each of the classes. We get a quadratic decision boundary-

- $\delta k(x) = x\verb|^|2 \cdot \mu_k / \sigma^2 - \mu_k^2 / 2\sigma\verb|^|2 + log(\pi_k)$

Random Forests

A random Forest starts with a Machine Learning technique called a Decision Tree. A decision tree is a set of Nodes with a tree graph structure. At each decision node, the input is tested (information about news article) and the algorithm goes to the next node. This continues till the end of the tree is reached, or the tree reaches a certain depth. A random forest combines trees with an ensemble. The Bagging algorithm creates B trees by taking B samples by means of bootstrapping. These trees are then averaged, to obtain an average tree. In Random Forests, in addition to this, a subset of m predictors is randomly selected at from a set of p predictors at each split. This is known to improve performance.

Results and Evaluation

From the above list of possible features, only the title and text was considered. Among the algorithms implemented, Multinomial Naïve Bayes and Logistic Regression gave the best accuracy with an accuracy for Title of 79% and 81% respectively, and an accuracy for Text of 90 and 92 % respectively. The summary of results is given below-

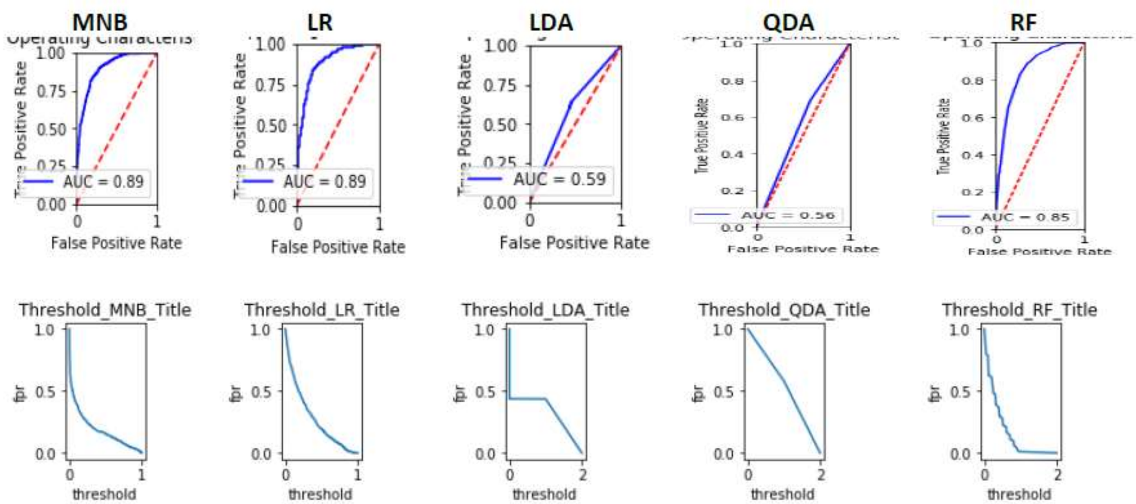
Analysis performed taking only Title into consideration

P(Fake|Title) - Probability of Fake news based on analyzing Title	
Algorithm	Accuracy
MNB (Multinomial Naïve Bayes)	79.95

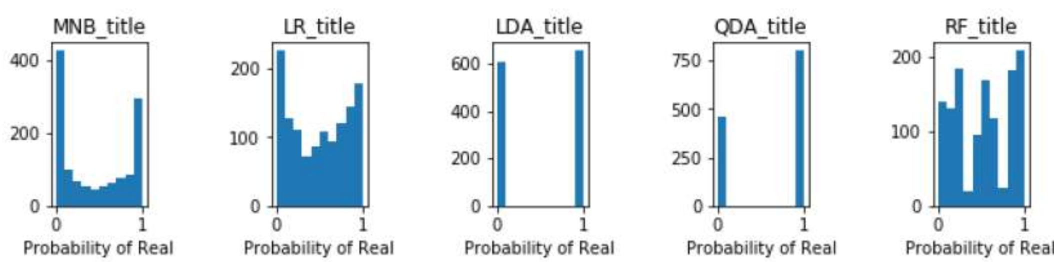
LR (Logistic Regression)	81.84
LDA (Linear Discriminant Analysis)	58.32
QDA (Quadratic Discriminant Analysis)	55.48
RF (Random Forests)	76.71

- **Plots**

1. ROC & Threshold



2. Posterior Probabilities



Analysis performed taking only Text into consideration

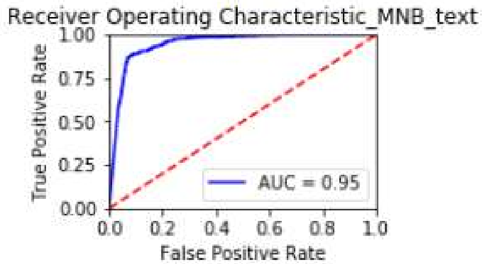
P(Fake|Text) - Probability of Fake news based on analyzing Text	
Algorithm	Accuracy

MNB	90.2
LR	92.8

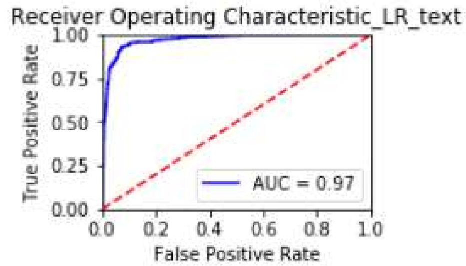
- **Plots**

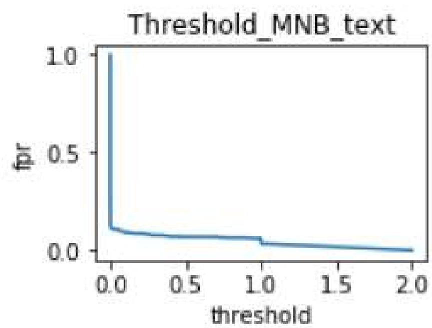
3. ROC & Threshold

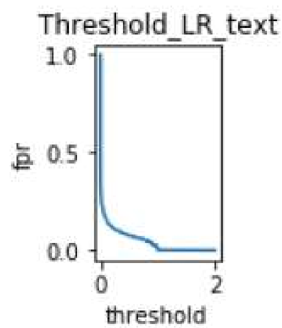
MNB



LR

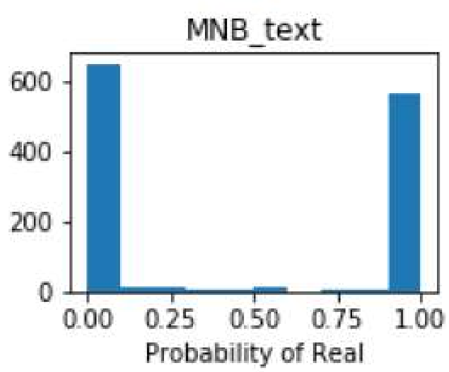




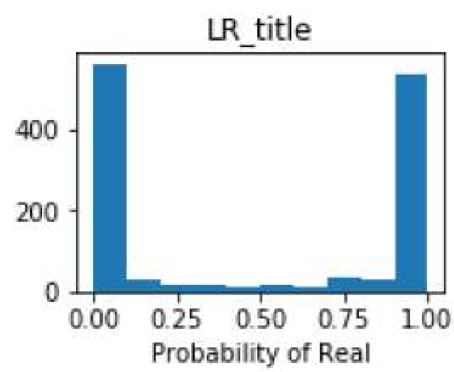


4. Posterior Probabilities

MNB



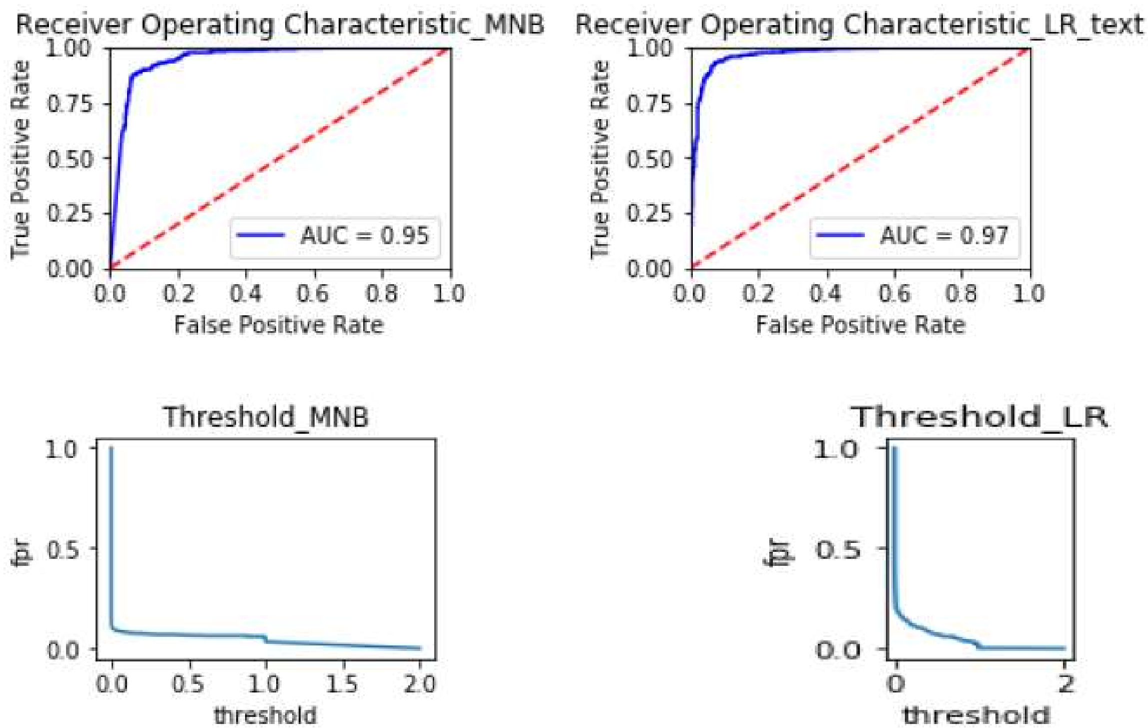
LR



Analysis performed taking both Text and Title into consideration

P(Fake|Text+Title)- Probability of Fake news based on analyzing Text and Title	
Algorithm	Accuracy
MNB	90.2
LR	92.6

- **Plots**
  5. ROC & Threshold



References

Allcott, H., & Gentzkow, M. (2017, January). *Social Media and Fake News in the 2016 Election.* Retrieved from https://web.stanford.edu/~gentzkow/research/fakenews.pdf

Silverman, C. (2016, November 16). *This Analysis Shows How Viral Fake Election News Stories Outperformed Real News On Facebook.*

Horne, Shah (2016), *This Just In: Fake News Packs a Lot in Title, Uses Simpler, Repetitive Content in Text Body, More Similar to Satire than Real News* Rensselaer Polytechnic Institute

Zhang, Harry. (2004). *The Optimality of Naive Bayes*.

http://www.cs.unb.ca/~hzhang/publications/FLAIRS04ZhangH.pdf