Results of RealvFake News Assessment
Andrew Monnot

Objective: Our objective in this project was to build a model that makes an accurate binary prediction (i.e. Real or Fake) based on an input string. The assumption made is that the input string is a news article.

Process: We use as a starting point two datasets from Kaggle, consisting of roughly 20-25k news articles each that are initially defined as "True" or "Fake". The strategy was to build supervised machine learning models as well as a neural network model, feed 80% of the data to the algorithms and network for training, and then assess on the remaining 20% of the data. We made assessments of the models using the area under the ROC curves in the 4 machine learning models, and of accuracy in all 5 models.

For the supervised setting, we used two machine learning algorithms (Logistic Regression and Gradient Boosted Trees) each with two different feature extraction methods (Word2Vec and HashingTF)--resulting in 4 different models. For the deep learning model we used a Keras LSTM neural network with a textprocessing words_to_sequences feature extractor.

After building and testing the models on the 20% testing subsets, we also built a small TestSet.csv file consisting of 10 recently scraped news articles about events that have happened (and marked them True) as well as 10 articles from the satirical Onion site (and marked them Fake). We then tested this set as well in all 5 models.

Early Observations and Exploration: Since all of the raw data consists of strings, all of the initial data exploration consisted of things like finding word counts, densities, common words, etc. The initial columns of the two files True.csv and Fake.csv were:

[title, text, subject, date]

We aren't likely to get useful information from the date unless we were interested in incorporating trends of Real or Fake news into our predictions, but even then we'd need to ensure we have a sufficiently temporally exhaustive dataset to ensure the data accurately reflects peaks or troughs of Real and Fake prevalence in the media. So for this reason, we decide to omit date from further analysis.

Doing an initial wordcount on the subject column immediately shows some concerns:

DF_True subject count

subject	count(subject)
politicsNews	11209
worldnews	10115
"" Trump Jr. resp...	2
"" said Phil Orlando	1
Associates Up 83...	1
Australia	1
"" according to t...	1
"" said Tom Vilsack	1
"" Petrocelli tol...	1
said Joseph Cassidy	1
U.S. beef hits C...	1
if implemented	1
Linda McMahon	1
"" he said during...	1
Pruitt decided n...	1
one of the lobby...	1
nor will help	1
Poland: bit.ly/2...	1
"" Intel Chief Ex...	1
"" Mattis said in...	1

only showing top 20 rows

DF_Fake subject count

subject	count(subject)
News	8501
politics	6525
left-news	4216
Government News	1543
US_News	767
Middle-east	762
fjs);}(document	124
2017"|	26
2016"|	18
2016Featured ima...	8
i'm sure it wasn...	8
s	7
2017 "|	7
2017Ranking memb...	7
and?"" he says. ...	6
make it into art...	5
fry em like baco...	5
in his mind	4
2017So all day	4
2017.@Delta empl...	4

only showing top 20 rows

As we can see, if the subject is “politicsNews” or “worldnews”, the article is almost certainly True, and we could simply guess Fake otherwise. Thus we also opt to ignore subject in the ML and DL models.

It is reasonable to include title in the assessment, however we have opted not to do this since the aim is to have text body be sufficient to infer the “Truthness” of the article. However, for convenience, we have included title in our pipelining process for potential later iterative analysis

Now, onto the crown jewel: the text column, which consists of the text body of the article. We first look at the top 100 words by density (total occurrences/total number of articles) in both DF_True and DF_Fake. Nothing in particular stands out here. However, when we look at “distinct word densities” (articles words appear in/total articles), a few words stand out (here we include the first half or so of them):

DF_True distinct word density: DF_False distinct word density:

(0.9986459354718215, '-') (0.94365657339977, 'the')
(0.9938833636830555, 'the') (0.9303266470763596, 'to')
(0.9915954615492366, '(Reuters)') (0.9217665346450321, 'of')
(0.9894943269365457, 'on') (0.9178484732336782, 'and')
(0.9672689919223048, 'a') (0.9148673395511264, 'a')
(0.9666153055983564, 'to') (0.8983859290490184, 'in')
(0.953214735957417, 'of') (0.8824581576593842, 's')
(0.9494793855348554, 'in') (0.8651675823005834, 'that')
(0.9366391184573003, 'and') (0.8543503257953239, 'is')
(0.8985852360274548, 'said') (0.838167028661471, 'for')
(0.8391931643087267, 'that') (0.8319066479281121, 'on')
(0.8191623476677405, 'for') (0.7781610663941059, 'with')
(0.7909137600971191, 'The') (0.7249265363485371, 'it')
(0.7597235840687304, 'with') (0.7044418891870022, 'was')
(0.7458094037446888, 'is') (0.6994165495507005, 'be')
(0.73404304991362, 'by') (0.6890251692858056, 'as')
(0.7166269785684269, 'was') (0.6823814999361185, 'by')
(0.699724517906336, 'as') (0.6803372939823688, 'have')
(0.6943082597936219, 'has') (0.6728418721519527, 'this')
(0.6744175187934818, 'from') (0.6705847280780205, 'The')
(0.6516785730961386, 'an') (0.6580213789872663, 'has')
(0.6485969089975253, 'not') (0.656403049273881, 'who')
(0.6427137320819909, 'have') (0.6532089774711469, 'not')
(0.6291263949199234, 'be') (0.650696307652996, 'at')
(0.6257645795396181, 'it') (0.6495890294280482, 'from')
(0.623056450483261, 'at') (0.6409863293726843, 'are')
(0.6075080543493486, 'he') (0.6358758144883097, 'his')
(0.5913993556520521, 'U.S.') (0.632766917933648 5, 'he')
(0.5584815800532288, 'would') (0.62241812529279, 't')
(0.5387776065742167, 'who') (0.6172224351603424, 'an')
(0.5381239202502685, 'President') (0.5875388612069332, 'about')
(0.5300929168417612, 'are') (0.572164728929773, 'they')
(0.5288789279544287, 'said.') (0.5213576934542822, 'but')
(0.5288322360741468, 'had') (0.5118180656701162, 'their')
(0.5281318578699165, 'his') (0.5041097057195179, 'Trump')
(0.524629966848765, 'which') (0.4904816660278523, 'been')
(0.5154316664332073, 'will') (0.4889059239385034 6, 'out')
(0.4961012279964514, 'after') (0.47642775009582217, 'I')
(0.4850352523696129, 'been') (0.46441804011754184, 'we')
(0.48115982630620535, 'but') (0.46382181338103146, 'or')

This shows that “(Reuters)” and “-” appear in 99% of the True articles, and we don’t even see them in the right hand column. In fact, both of these words appear respectively in .03% and .2% of the Fake articles. Thus if an article has one of these two words in it, it is almost certainly True, so we should remove these during the preprocessing phase.

The problem here is that all the articles from True.csv were actually chosen from Reuters, so this is contributing to that high prevalence.

For similar reasons, we also opt to remove the words “Trump's”, “"I”, “"The”, and “”We” from the text body in preprocessing.

Results: After 5 maximum iterations on the 80%-size training set, the areas under the 4 ROC curves for the machine learning models applied to the 20%-size testing set were:

LR, Word2Vec: 0.9892923963033571
GBT, Word2Vec: 0.9578940242388466
LR, HashingTF: 0.9822392314954729
GBT, HashingTF: 0.9657035070853843

And the accuracy on them was:

Accuracy:

LR, Word2Vec: 0.9510019819423035
GBT, Word2Vec: 0.8816340013212949
LR, HashingTF: 0.9241356529398811
GBT, HashingTF: 0.9003523452983925

With another 80-20% train-test split, the Keras LSTM neural network attained the following results on the test set:

loss: 0.0436 - accuracy: 0.9883

This of course looks very good, but then we ran all 5 models on the TestSet.csv file–consisting of the 10 recently scraped True articles and 10 Onion articles marked as Fake. The accuracies here were not as good:

Accuracy:

LR, Word2Vec: 0.5
GBT, Word2Vec: 0.5
LR, HashingTF: 0.6
GBT, HashingTF: 0.45

And for the neural network:

loss: 2.3347 - accuracy: 0.5500

In a nutshell, these models all had a propensity to predict most of the articles in TestSet.csv as True–resulting in high true positive rates but also high false positive rates.

Conclusions and Reiteration Ideas: While the models all performed well on their testing subsets, there were shortcomings on the custom-built TestSet.csv. Since the models did a good job with true positives in all cases, one thing to look at then is if there were any noticeable differences between Fake articles in Fake.csv versus Fake articles in TestSet.csv. Skimming through the articles in Fake.csv, many seem to be op-ed style, whereas the ones in True.csv are your standard news articles. Onion articles (aka the Fake articles of TestSet.csv) are often written like standard news articles (and not in op-ed form), but simply have hyperbolically false assertions for the purposes of satire. Because of this, the models may have difficulty in distinguishing Fake.csv articles from Onion articles–even though both are classified as Fake in this context. That is, the models may assess: if standard news style, then True, else if op-ed style, then Fake.

Thus in a future iteration, we could attempt to run the models using op-ed style articles with fake or hyperbolic assertions in them as our Fake data and see how the models perform. Of course, another initial cause for concern is that the data from True.csv apparently came from Reuters exclusively, so updating True.csv with a more eclectic set of data would be ideal.

A possible non-ML strategy for discerning the datasets could be to look at the word set of an input article and compare them to the exclusive DF_True wordsets and DF_Fake wordsets and see which overlap is greater.