Detection of Fake News Posts on Facebook

I. INTRODUCTION

In today's social media obsessed world, it is quite uncommon to verify the authenticity of facts or news. There have been innumerous case of fake news being shared and promoted for private propaganda. In our analysis we have tried to determine the authenticity of the news posts on Facebook by using machine learning algorithms to classify the news as false or true.

The dataset for this analysis is derived from a labelled dataset for Facebook news post by multiple news agencies available on GitHub. Using the post ID and Facebook API (in R) we have collected the information associated with each post.

Our goal is to classify the posts using the post message, user comments and reactions on it.

II. METHOD

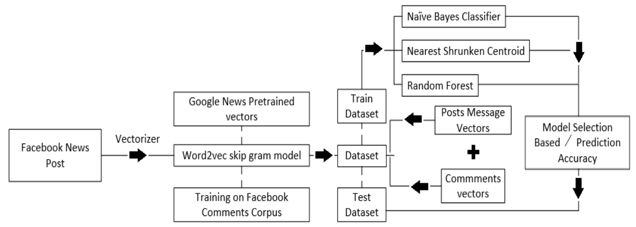

Figure 1. Diagram illustrating the modeling and prediction process

A. Data Collection

The original labelled dataset available on GitHub had more than 2000 unique post with class labels. However, the proportion of each class in the dataset was highly biased. To overcome this limitation, we sample 288 unique posts containing 67 percent posts belonging to class labelled as true and 33 percent to false. Then this data set was randomly partitioned into 30 percent test and 70 percent train dataset.

B. Vectorization

Per Mikolov et al., words and phrases can be represented by the Skip-gram model. In this respect, it becomes possible to apply analogical reasoning by simple vector calculations. The linear form of the vectors that symbolizes words can be add-up each other to bring the sentiment of a text [3].

For each post p we have comments c with different number of words, w:

$$P_i\colon\{(c_1\colon[w_{11}, w_{12} \dots w_{n_1}])_{pi}, (c_2\colon[w_{21}, w_{22} \dots w_{n_2}])_{pi} \dots (c_n\colon[w_{n1}, w_{n2} \dots w_{n_n}])_{i}$$

To determine the 300-dimensional sentiment of a post, this technique was put into use:

$$\vec{p_i} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n_i}\vec{w}_{ij}/\,n_i$$

The first approach involved training the vectors on the comments corpus and for second approach a pre-trained vector on 3 billion words (on Mikolov's word2vec skip gram model) from google news was chosen.

The wordVectors package by bmschimdt was used to tokenize the post message and comments into a 300-dimensional vector space using Mikolov's word2vec skip gram model. Then the post message and comments vectors were averaged for each post and combined in a proportion selected by cross validation.

The comments were also vectorized using the pre-trained vector on google news and comparative analysis study was undertaken to determine the best choice of vectors for comments.

C. Exploration

Due to high no of predictors (~304) we use various unsupervised clustering techniques like k-means to detect clusters and Principal Components Analysis (PCA) as shown in figure 2 and figure 3, to inspect the predictors in a lower dimensional space.

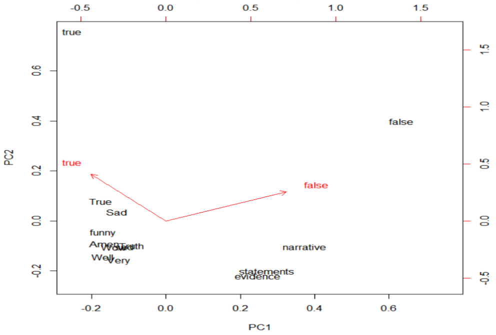

Figure 2. Biplot of principal components of cosine similarity vectors of "true" and "false" trained on comments corpus

However, the first few principal components failed to explain a large proportion of variance and the clustering technique using k-means performed poorly in detecting separable clusters.

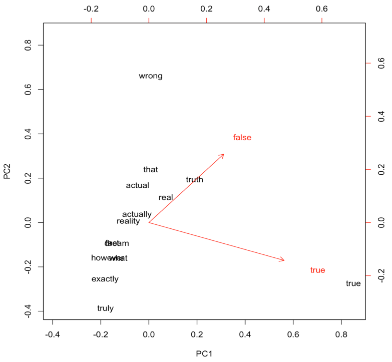

Figure 3. Bi-plot of principal components of cosine similarity vectors of "true" and "false" pre-trained on google news

Figure 4 and figure 5 list the nearest words to "true" and "false" in the vectorized n dimensional space.

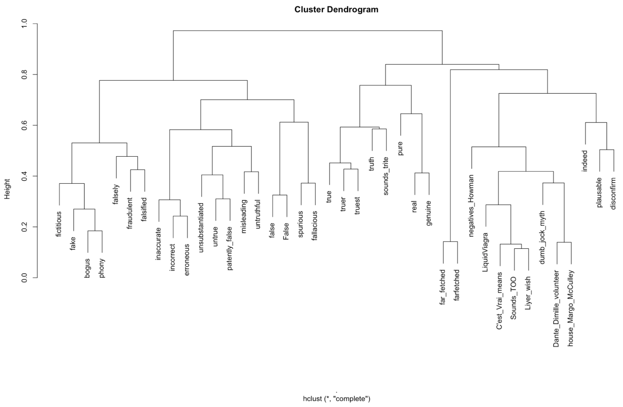

Figure 4. Cluster Dendrogram of top words nearest to vector of "true" and "false" trained on comments corpus

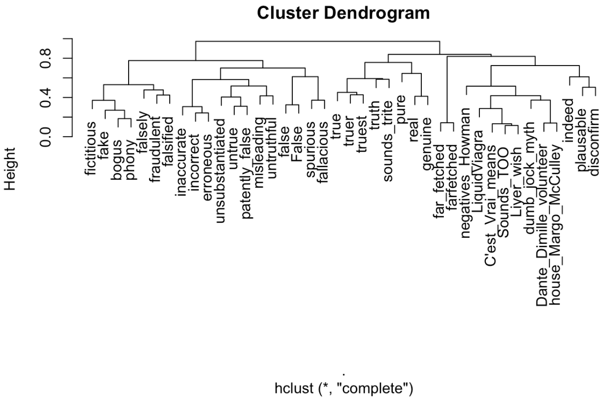

Figure 5. Cluster Dendrogram of top words nearest to vector of "true" and "false" pre-trained on google news

D. Modelling Techniques

The Ratio of no of predictors to observations is quite high ~ 1.5 and hence we decided to explore machine learning techniques like Nearest Shrunken Centroid and Random Forest Machine Learning Techniques. Since most of the 300 dimensional vectors were uncorrelated, we also explored the performance using a Naive Bayes Classifier.

1) Nearest Shrunken Centroid

In this technique, a standardized version of centroid is computed for each class. The centroid of each class is shrunken towards the overall centroid for all classes by the threshold amount which can be computed by cross validation.

It compares each new data point to these shrunken class centroids and the class whose centroid is nearest to that data point is the predicted class for that data point.

The implementation of Nearest shrunken centroids (NSC) classifiers for predicting the reliability of Facebook news is robust approach for classification problems with multi-dimensional predictors. Each dimension holds a unique sense to the Facebook news and by shrinking the extend, we can acquire most accurate predictors per the research conducted by Tibshirani et al. By this, the algorithm subsets the dimensions that represents the data with the most related sentiments[6].

For every news post on Facebook, our dataset has 304 dimensions. Each can be expressed by x_{ij} ($i: predictors, j: observations$) to predict 2 classes, $k = 2$. For the ith component of the centroid for class k, NSC standardizes in-class centroids for all classes.

The training sample in p observation for our data is $\{(x_{1j}, y_1), (x_{2j}, y_2) \dots, (x_{nj}, y_n)\}_{j=1,2\dots p}$.

Class centroids are $\mu_k = \Sigma_{j=1}^{n_k} x_{ij}/n_k$.

Let

$$d_{ik} = \frac{\mu_{ik} - \mu_i}{\sqrt{\frac{1}{n_k} + \frac{1}{n} (s_i + s_0)^2}} \quad [1]$$

where d_{ik} represents the estimated standard error divided by $\sim{s_i}^2$, a variation inside the class for ith dimension. Also, s_0 (median of σ for all dimensions) is to prevent the d_{ik} values to increase for the dimensions with low expressions levels [6].

After the shrinkage of d_{ik} values towards zero by using soft thresholding, many of the dimensions are likely to be eliminated as the Δ value increased.

Soft thresholding:

$$d_{ik}' = sign(d_{ik})(|d_{ik}| - \Delta) + , \qquad [2]$$

To capture strong threshold, we applied cross-validation preventing to eliminate all the dimensions becoming zero and to choose the possible smallest size with smallest error rates.

2) *Random Forest*

It is an ensemble machine learning technique in which many trees are grown using a subset of predictors. This helps reduce over fitting of the training set as it helps to reduce the collinearity between predictors by taking a subset of predictors for growing new tree and by only taking a sample of training data points for growing new trees (bagging).

For growing regression trees the residual sum of squares is taken as the appropriate error measure rate given by

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2,$$

and for classification trees we have the classification error rate(E), Gini index(G) and the entropy (D) given by

$$E = 1 - \max_k(\hat{p}_{mk}).$$

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

A sample of observations are taken to grow each new tree(bagging). A prediction model is built on each tree and the resulting predictions are averaged. This helps to reduce the overall variance. The error rate for bagging is given by

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^{*b}(x)$$

On average, each bagged tree uses about two-thirds of the observations. The remaining one-thirds (referred as out of bag observations) can be used to get the predictions for the given observation.

3) *Naïve Bayes*

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x} \mid C_k)}{p(\mathbf{x})} \qquad \hat{y} = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}}\, p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k).$$

This method is constructed using the Bayes conditional probability theorem. It assumes that value of a feature is independent of the value of other features given the class variable. In one of the research was stated that Naïve Bayes estimation of word based skip-gram models are interpreting words independently [5]. The classifier tries to predict probability of class k given the average of word vectors for that news post. For mathematical representation:

$$\Pr(k = 1|\vec{p}) = \frac{\Pr(k = 1) \prod_i \Pr(\vec{p}_i|k = 1)}{\Pr(k = 1) \prod_i \Pr(\vec{p}_i|k = 1) + \Pr(k = 0) \prod_i \Pr(\vec{p}_i|k = 0)}$$

For the $k = 0$ the expression is the vice-versa.

III. RESULTS

We have used two different approaches for the feature creation. First, we used word vectors trained on Google News and calculated the arithmetic mean for the words within comments for each post.

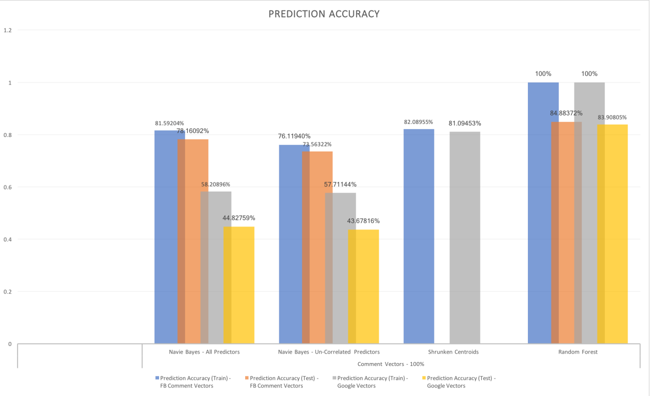

Figure 6. Prediction Accuracy Comparison between Google News pre-trained vectors and Facebook Comments corpus

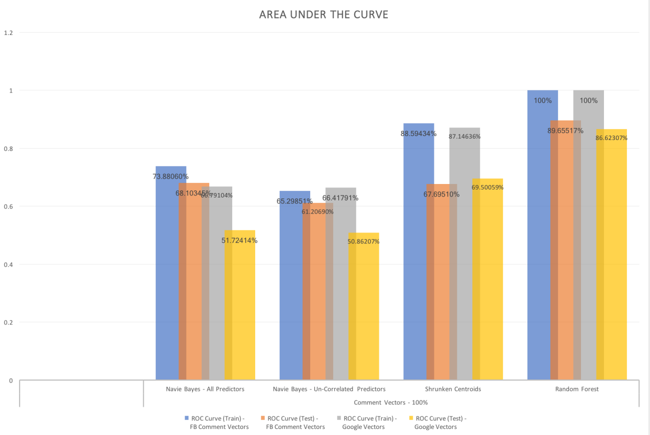

Figure 7. Area Under ROC Curve Comparison between Google News pre-trained vectors and Facebook Comments corpus

For the second approach, we use the word vectors trained on the Facebook comments corpus to retrieve 300 dimensional vectors.

The vector trained on comments corpus outperforms the vector trained on google news vector as shown in figure 6 and 7, however this might be because of overfitting. The comments corpus more closely resembles the corpus we need for our analysis however the results may be biased because of similar training.

The best proportion for comments and post vectors determined by cross validation is 10% and 90% resp. as shown in the figures 8 and figure 9.

There is a slight decrease in prediction accuracy and area under ROC curves as the proportion of comments increase from 10% to 100%.

Random forest machine learning technique outperforms other method for prediction accuracy and has higher area under the curve roc curve. This behavior can be explained because of low training observations (201) and since trees in random forest are fully grown and not pruned which might coerce it to find a perfect split till the very last node.

Nearest shrunken Centroid method also gives good prediction results > 90 % and is a reliable technique used in high dimensional setting. However, its prediction accuracy and area under the ROC curve are less than that obtained by random forest technique.

We built two models using the Naïve Bayes Classifier. The first model was built using all the predictors (304) and the second model using de-correlated predictors (302) determined by the pairwise correlation matrix. Model 1 outperforms model 2 in prediction accuracy and area under the ROC curve however, the relative increase in prediction accuracy is a mere due to presence of highly correlated predictors. It should be noted that Model 2 meets the assumptions for application of a Naïve Bayes Classifier.

Both the models built using Naïve Bayes classifier perform poorly compared to other two techniques which are particularly better suited for high dimensional setting.

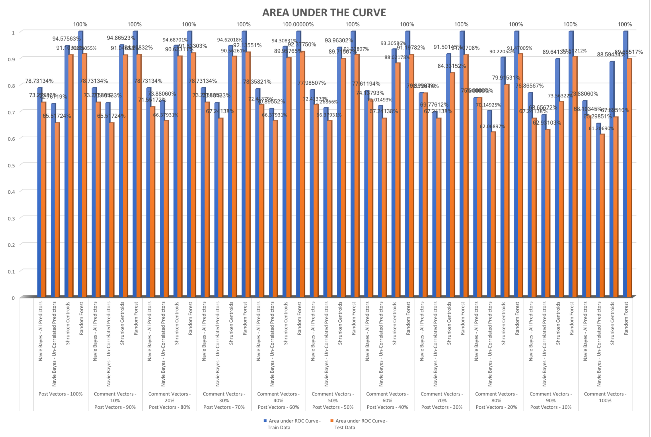

Figure 8. Area under ROC curve for different proportions of comments and post vectors for Train and Test data.
(Using Cross Validating for determining the best proportion of Post & Comment Vectors)

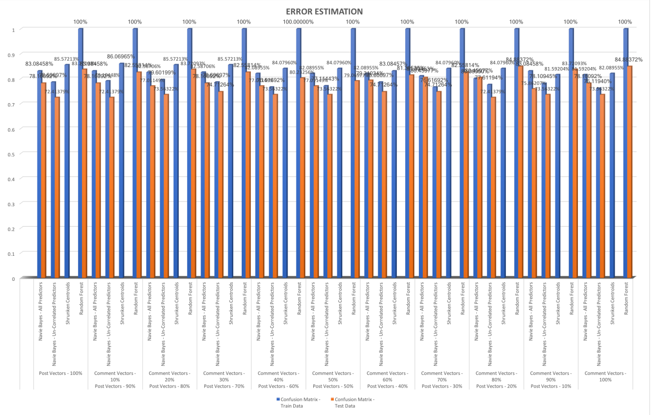

Figure 9. Prediction Accuracy for different proportions of comments and post vectors for Train and Test data. (Using Cross Validating for determining the best proportion of Post & Comment Vectors)

IV. DISCUSSIONS

The predictor – 'Share count' of a post is an important predictor in classifying the post as found in random forest and some preliminary exploration with correlation matrix. Intuitively it makes sense that people won't be willing to share a news post that is fake.

Naïve Bayes did perform poorly with Google news vectors as well as the Facebook comments corpus trained vector as the method is based on Bayesian theorem which tries to consider each word independent of other predictors. However, for further analysis, Naïve Bayes method could perform better with tf-idf vectors which is explained in one research [2].

Also, the Random Forest method might over fit on the training data as the dataset size comprises of 201 observations. The effect of overfitting might be more visible when the comparing the original dataset (with

201 observations) with a dataset that has large no of observations.

In the course of data preparation, the computational challenge decrease our speed when we were dealing with 3 billion of words from Google News document. Therefore, our analysis for cross-validating the News post using Google News pre-trained vectors would be time consuming. Although it is a good area to discover it for further consideration. As this analysis just include the word vectors of comments made by users which include high number of irrelevant comments compared to the post tags. In this respect, it might be effective to include news posts. Moreover, the computational challenge also brings a problem about using weighted vectors for each comment since the comments of each news post are very broad. For future work, the importance of the words to detect "true" and "fake" news can be calculated by td-idf tokenizers and based upon the founded importance the weighted average can be applied to strengthen the sentiments of each posts.

V. References

[1] http://statweb.stanford.edu/~tibs/PAM/Rdist/howwork.html)

[2] Joachims, T. (2002). Learning to Classify Text Using Support Vector Machines. doi:10.1007/978-1-4615-0907-3

[3] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems*. 2013H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.

[4] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013)

[5] Sarkar, S. D., Goswami, S., Agarwal, A., & Aktar, J. (2014). A Novel Feature Selection Technique for Text Classification Using Naïve Bayes. International Scholarly Research Notices,2014, 1-10. doi:10.1155/2014/717092

[6] Tibshirani, R., Hastie, T., Narasimhan, B., & Chu, G. (2002). Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences,99*(10), 6567-6572. doi:10.1073/pnas.082099299