COVID Infodemic: A misinformation classifier for coronavirus related news using Machine Learning

Kriti Kansal
kriti@vt.edu

Rajarshi Mukherjee
rajarshim13@vt.edu

Spoorthi Airody Suresh
spoorthi@vt.edu

Abstract

With the ever burgeoning levels of misinformation, it is important to differentiate between real and fake information sources.Our work involves sifting through text data in the form of tweets related to the coronavirus pandemic, processing it to best remove any unwanted noise, and then make use of several popular machine learning models to effectively differentiate between credible and fake sources of information. Thus, we propose a comparative study of these different models and contrast their results against each other.

1 Problem statement

Ever since the social media boom, it became very simple to disseminate news, whilst fact-checking of the same failed to scale at the same rate. With the advent of the COVID-19 pandemic, this problem became ever more apparent. We need a classifier that sorts through the information on social media, flags the fake content, and potentially redirects the user to a credible and vetted source. In this paper, we try to take inspiration from a few existing publications where machine learning tools were used for automating this problem, and if possible, improve upon them.

2 Data

Using the CoAID [1] misinformation dataset, we first pre-processed the data and have discussed the same in sub-section 2.1. Thereafter, we proceeded to select the additional features that we thought were relevant to our models, and discuss that in sub-section 2.2.

2.1 Data Preprocessing

The chosen dataset, CoAID, has a diversified misinformation dataset that includes both fake and true news articles from websites and social media. It also includes tweet data and user engagement data collected from Twitter's Advanced Search API. Researchers are authorized to access and take data from Twitter's live feed or search and pull older tweets, but they are not allowed to share the raw data with a third party due to Twitter's policies. Hence, the datasets pertaining to Twitter only have user IDs and Tweet IDs. To re-create a raw dataset consisting of other attributes such as the content of the tweet, we hydrate the tweet IDs. For hydrating we use a desktop application called, Hydrator[2]. In order to fetch information related to the tweet ID using Hydrator, one must configure a Twitter Developer account. Figure 1 shows the process of hydrating tweet IDs using the Hydrator.

From the CoAID dataset, we extract 4 sets of fake and real news tweets alike, and pull information pertaining to the tweet IDs using Hydrator. Table 1 shows the number of tweets available in the datasets against the actual number of tweets extracted during the process. We observed that around 14.5% of the tweets were lost during hydration – this is mostly because these tweets may have been deleted and hence could not be fetched using their tweet IDs. After extraction of these tweet IDs, we see that there are over 30 features available. We added a feature "credible" with values "-1" for fake

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

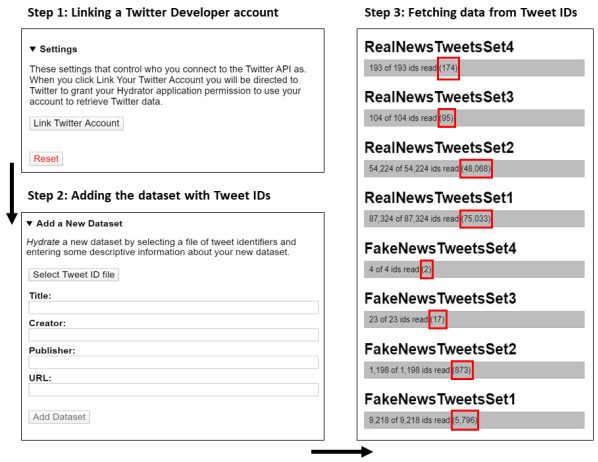

Figure 1: Process of hydrating tweet IDs using Hydrator.

Table 1: Dataset Properties.

Dataset	Fake				Real				Total
	Set1	Set2	Set3	Set4	Set1	Set2	Set3	Set4	
Before Hydrating	9218	1198	23	4	87324	54224	104	193	152288
After Hydrating	5798	873	17	2	75029	48076	95	174	130064

news and “1” for real news. Then, the 8 labeled datasets are combined into a single dataset to process further.

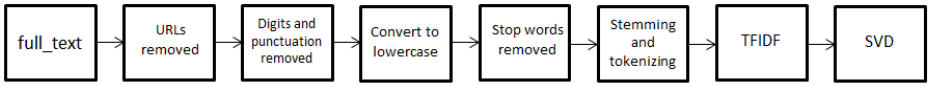

Figure 2: Steps undergone to process the data.

For any dataset, we aim to build a model that can predict with the highest degree of confidence as possible. The algorithm employed, the size and quality of the dataset, and the features used to train the model all have a significant impact on the model’s performance. It is quite common to find features in their raw form in a dataset which does not provide the best information for training a model. The Twitter dataset, for example, has a feature called full_text, which is one of the most critical attributes that influences model performance. The raw full_text, on the other hand, cannot be fed into the model as is and must be modified to build a more predictive system.

Figure 2 highlights the steps that were followed to process our dataset and to best rid it of unwanted information and noise. The TF-IDF (Term-Frequency,Inverse Document Frequency) operation was carried out to accurately ascertain the relative values each word holds across the entire dataset.This helped us observe how some words are repeated more often and also to better map word associativity. We carried out basic SVD (Singular Value Decomposition) and UMAP (Uniform Manifold Approximation and Projection) dimensionality reduction operations to better understand the distribution of the data and this additionally helped us to avoid the "curse of dimensionality". Figure 3 shows the clustering of data points when 6500 random samples are chosen and reduced from the fake and real news sets (each) using UMAP. As we can observe, this seems to suggest a non-linearly separable set of data-points (as expected), but shows points that are well clustered locally. We believe that after thorough feature selection, we can separate these values better.

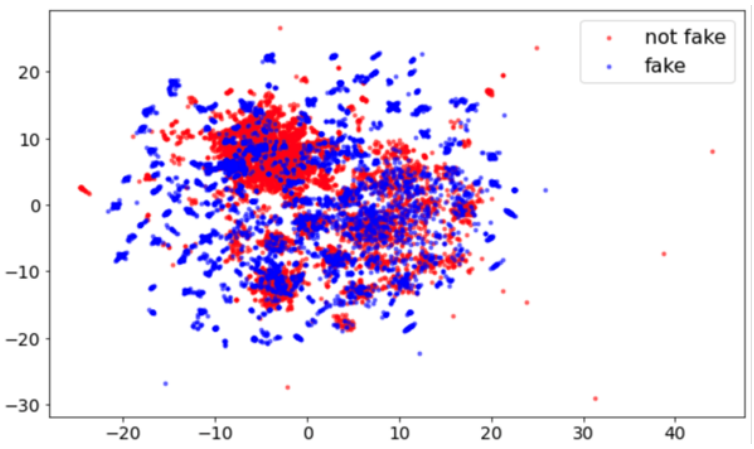

Figure 3: UMAP scatter plot.

2.2 Feature Extraction

Another important strategy we used is feature selection. In our dataset of choice, over 32 features were available. However, all these features may not be useful in building an accurate model. Careful feature selection helped to improve the accuracy with which the model is able to predict but also reduced the computational cost. We employed various approaches to select the most feasible features. We first eliminated features that did not have varying values; features like `truncated`, `contributors`, and `protected` had the same values for all tweet IDs. We also removed features that majorly consisted of NaN values. Features with duplicate values were dropped – features like `id` and `id_str` had values but in varied formats.
With the help of correlation matrices, we determined which features are linearly dependent on each other and eliminated one of the them. As an instance, we eliminated the feature `verified` as the features `followers_count` and `verified` were found to be closely related.
After the basic elimination process, the feature importances for 8 features were determined with the help of the feature importance method for random forest classifier in scikit-learn. The final features that were extracted for training the models are as below:

- `full_text`: The body of the tweet.
- `follower_count`: The current number of followers for this account.
- `retweet_count`: The number of times the tweet has been retweeted.
- `favorite_count`: The number of times the tweet has been liked
- `statuses_count`: The total number of tweets and retweets by a user.

We had to discard a few examples from the original data-set due to data corruptions/absences in some instances of the selected feature fields.

3 Experiments

As shown in in Figure 2, we first carried out an extensive pre-processing sequence of operations to remove unwanted noise from the text data. We considered these different features along with the text of the tweet(which had been dimensionally reduced to two dimensions) as columns of one large table, and started with feeding this tabular data to the different machine learning models.

3.1 Ensembling

Furthermore, we decided to implement stacking, or ensembling, on the same data-set to compare the results. As shown in Figure 4, ensembling is essentially a 2 stage process. First, individual base classifiers are trained on the data-set. Thereafter, a meta-classifier is trained on the responses obtained from the different base classifiers. Ultimately, the meta-classifier is responsible for the decisions that

are made by the model.
We retained four prediction families for level-0 classifiers, covering a wide range of methods: KNN, SVM, decision tree and logistic regression. We cycled through different models to select the optimum meta classifier algorithm, the results of which are included in the next section.

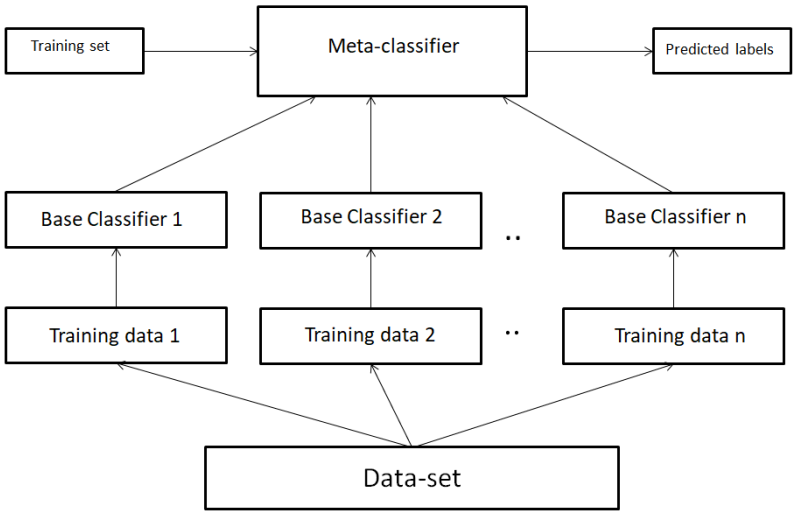

Figure 4: Stacking ML models

4 Results

From our experiments, we noticed that in the case of noisy data or when target classes are overlapping, KNN and SVM performed sub-optimally. At times, logistic regression was seen to perform poorly and this might be due to the target label not having a linear correlation with the features. In such cases, logistic regression (or linear regression for regression problems) could predict targets with good accuracy (even on the training data).
Additionally, upon looking at Figure 5 as compared to the confusion matrices of the individual models, we saw that the stacked classifier was getting an almost equal percentage of false positives and false negatives, whilst the other classifiers were terribly skewed towards one or the other. We feel that this showcases a more balanced approach considering how(in this application) we would want to avoid both scenarios with an equal emphasis.

Table 2: Performance of different models

Model	Accuracy	Precision	F1 Score	Recall
SVM	80.59%	94.41%	89.21%	84.45%
KNN	94.10%	95.39%	96.94%	98.56%
Logistic Regression	94.84%	94.86%	97.35%	99.88%
Decision Tree Classifier	94.16%	98.17%	98.03%	97.88%
Stacking	**96.41%**	**98.24%**	**98.11%**	**97.98%**

To select the algorithm for the meta-classifier, we ran several trials whilst making use of different classifiers, as shown in Table 3, and after comparing the results, we decided to go ahead with an SVM based meta classifier for our work

Combining all machine learning models and stacking them, helps us improve the accuracy and performance. We conclude that stacking improves accuracy while keeping variance and bias low. We can see from Table 2 that the stacking model outperforms the performance of the individual models across almost all parameters.We feel that thereby, an ensembled approach makes for a more balanced and predictable system in general.

Table 3: Meta-classifier performance

Model	Accuracy	Precision	F1 Score	Recall
SVM	96.41%	98.24%	98.11%	97.98%
KNN	95.99%	97.29%	97.90%	98.52%
Logistic Regression	96.19%	98.08%	97.99%	97.89%
Decision Tree Classifier	94.34%	97.02%	97.02%	97.01%

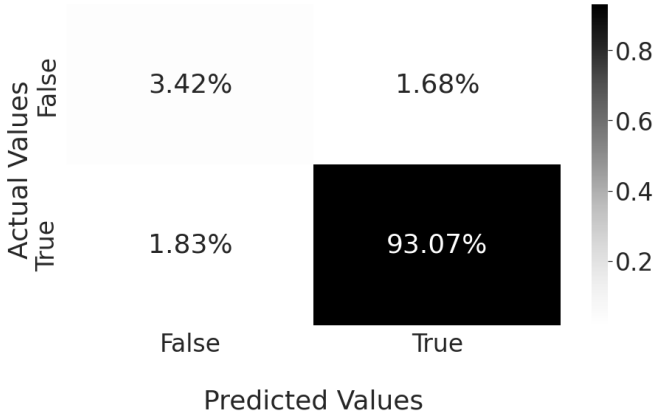

Figure 5: Confusion Matrix for Stacking

As future work, we believe that more features could potentially be included by exploring alternative data-sets that might emerge. We feel that the carefully curated inclusion of more correlated fields could further help generalize the model, and enhance its classification abilities further.

5 Contribution

We believed that it was necessary for each one of us to be involved in the entire implementation process as it went forward, however if we were asked to broadly categorise our individual contributions,we would say that Spoorthi was responsible for assimilating and organizing the data from Twitter into a usable format. Rajarshi worked on data-preprocessing for the models. Kriti was in charge of setting up our base classifiers. We'd claim equal responsibility for the tweaks required to get the stacked classifier working, and also for analysis.

References

[1] L. Cui and D. Lee, "CoAID: COVID-19 healthcare misinformation dataset," *arXiv preprint* arXiv:2006.00885, 2020.

[2] E. Summers and N. Ruest, "Hydrator: Turn tweet ids into Twitter Json; CSV from your desktop!," *GitHub*. [Online]. Available: https://github.com/docnow/hydrator. [Accessed: 17-Mar-2022].

[3] M. N. Alenezi and Z. M. Alqenaei, "Machine learning in detecting COVID-19 misinformation on Twitter," *Future Internet*, vol. 13, no. 10, p. 244, 2021.

[4] S. Dadgar and M. Ghatee, "Checkovid: A COVID-19 misinformation detection system on Twitter using network and content mining perspectives,"*arXiv preprint* arXiv:2107.09768, 2021.

[5] S. Aphiwongsophon and P. Chongstitvatana, "Identifying misinformation on Twitter with a support vector machine,"*Engineering and Applied Science Research*, vol. 47, no. 3, pp. 306–312, 2020.

[6]M. S. Al-Rakhami and A. M. Al-Amri, "Lies kill, facts save: detecting COVID-19 misinformation in twitter,"*IEEE Access*, vol. 8, pp. 155 961–155 970, 2020.