# An Ensemble Model Using N-grams and Statistical Features to Identify Fake News Spreaders on Twitter

## Notebook for PAN at CLEF 2020

Jakab Buda, Flora Bolonyai

Eötvös Loránd University, Budapest
bakajb@gmail.com, f.bolonyai@gmail.com

**Abstract.** In this notebook, we summarize our work process of preparing a software for the PAN 2020 Profiling Fake News Spreaders on Twitter task. Our final software was a stacking ensemble classifier of five different machine learning models; four of them use word n-grams as features, while the fifth one was based on statistical features extracted from the Twitter feeds. Our software uploaded to the TIRA platform achieved an accuracy of 75% in English and 80.5% in Spanish. Our overall accuracy of 77.75% turned out to be a tie for the first place in the competition.

## 1 Introduction

The aim of the PAN 2020 Profiling Fake News Spreaders on Twitter task [12] was to investigate whether the author of a given Twitter feed is likely to spread fake news. The training and test sets of the task consisted of English and Spanish Twitter feeds [13].

We used an ensemble of different machine learning models to provide a prediction for each user. All of our sub-models handle the Twitter feed of a user as a unit and determine a probability for each user how likely they are to be fake news spreaders. For the final predictions, these sub-models are combined using a logistic regression.

In Section 2 we present some related works on profiling fake news spreaders. In Section 3 we describe our approach in detail together with the extracted features and models. In Section 4 we present our results. In Section 5 we discuss some potential future work and in Section 6 we conclude our notebook.

## 2 Related Works

Using word n-gram variables for author profiling has been shown to be effective [3, 5, 9, 14, 15, 18], especially with TF-IDF weighting [20]. Identifying fake news based on such features has been tested earlier [1]. Statistical features, such as the number of punctuation marks [15, 19], medium-specific symbols (for example

hashtags, and at signs in tweets, links in digital texts) [7, 8, 14, 15, 17, 19], emoticons [7, 8, 14, 16, 19] or stylistic features [8] are also commonly used for text classification purposes.

SVMs [3, 5, 9, 14, 15], XGBoost [21], logistic regression [19] and random forest [2] models are commonly used for author profiling and text classification purposes. Although the state-of-the-art results for many text classification tasks are achieved with transformer-based language models [4, 11], these are computationally very expensive solutions and perform better on tasks where text semantics is more important. Ghanem et al. proposed an emotionally infused LSTM model to detect false information in social media and news articles. Their model yielded state-of-the-art results on three datasets, but it is also computationally expensive [6], so experimenting with lighter approaches still has practical benefits.

## 3 Our Approach

### 3.1 The corpus and the environment setup

#### *3.1.1 The corpus*

The corpus for the PAN 2020 Profiling Fake News Spreaders on Twitter task [12] consists of one English and one Spanish corpus, each containing 300 XML files. Each of these files contains 100 tweets from an author. Because of the moderate size of the corpus, we wanted to avoid splitting the corpus into a training and a development set. Therefore, we used cross-validation techniques to prevent overfitting. As opposed to earlier editions of the PAN competition, the dataset this year came pre-cleaned: all urls, hashtags and user mentions in the tweets were changed to standardized tokens.

#### *3.1.2 Environment setup*

We developed our software using the Python language (version 3.7). To build our models we mainly used the following packages: scikit-learn[1], xgboost[2], emoji[3], lexical-diversity[4], pandas[5] and numpy[6]. Our codes are available on GitHub[7].

[1] https://scikit-learn.org/
[2] https://xgboost.readthedocs.io/
[3] https://pypi.org/project/emoji/
[4] https://pypi.org/project/lexical-diversity/
[5] https://pandas.pydata.org/
[6] https://numpy.org/
[7] https://github.com/pan-webis-de/bolonyai20

### 3.2 Our models

*3.2.1 N-gram models*

We experimented with a number of machine learning models based on word n-grams extracted from the text. Precisely, we investigated the performance of regularized logistic regressions (LR), random forests (RF), XGBoost classifiers (XGB) and linear support vector machines (SVM). For all four models, we ran an extensive grid search combined with five-fold cross-validation to find the optimal text preparation method, vectorization technique and modeling parameters. We tested the same parameters for the English and Spanish data. We investigated two types of text cleaning methods for all models. The first method (M1) removed all non alphanumeric characters (except #) from the text, while the second method (M2) removed most non alphanumeric characters (except #) but kept emoticons and emojis. Both methods transformed the text to lower case. Regarding the vectorization of the corpus, we experimented with a number of parameters. We tested different word n-gram ranges (unigrams, bigrams, unigrams and bigrams) and also looked at different scenarios regarding the minimum overall document frequency of the word n-grams (3, 4, 5, 6, 7, 8, 9, 10) included as features. Table 1 describes the tested model hyperparameter values during the training phase of our models.

**Table 1:** Grid-searched hyperparameters for the used machine learning models

| **Model** | **Model hyperparameters** | |
|---|---|---|
| | **Name** (*Python parameter name*) | **Values** |
| LR | Regularization coefficient (*C*) | {0.1,1,10,100,1000} |
| RF | Number of boosting rounds (*B*) | {100,300,400} |
| | Minimum number of cases on each leaf (*min_samples_leaf*) | {5,6,7,8,9,10} |
| SVM | Regularization coefficient (*C*) | {1,10,100,1000} |
| XGB | Learning rate (*eta*): | {0.01,0.1,0.3} |
| | Number of estimators (*n_estimators*) | {200,300} |
| | Maximum depth of a tree (*max_depth*) | {3,4,5,6} |
| | Subsample ratio (*subsample*) | {0.6,0.7,0.8} |
| | Subsample ratio of columns (*colsample_bytree*) | {0.5,0.6,0.7} |

For the early bird testing phase conducted through TIRA [10], we simply chose the model and parameter combination in each language that had the highest accuracy during the cross-validation and fitted these models on the entire training set. However, the accuracy of our model was approximately 5% lower on the test set

compared to the cross-validation results (79% vs. 83% for the Spanish dataset and 69% vs. 76% for the English dataset), so we used a different approach during the final testing phase.

The ensemble method we used for the final version of our software (described in detail in Section 3.2.3) required the best text cleaning and vectorization parameters and hyperparameters for each model. These hyperparameters are summarized in Table 2.

**Table 2:** The best performing text cleaning methods, vectorization parameters and model hyperparameters for the n-gram based machine learning models

| Language | Model | Text cleaning | Vectorization | | Model hyperparameters[8] |
|---|---|---|---|---|---|
| | | | N-grams | Min. global occurrence | |
| EN | LR | M1 | uni- and bigrams | 6 | C=1000 |
| | RF | M2 | uni- and bigrams | 9 | B=300 |
| | | | | | min_samples_leaf=9 |
| | SVM | M1 | uni- and bigrams | 5 | C=100 |
| | XGB | M1 | uni- and bigrams | 8 | eta= 0.01 |
| | | | | | max_depth=6 |
| | | | | | colsample_bytree=0.6 |
| | | | | | subsample=0.8 |
| | | | | | n_estimators=300 |
| ES | LR | M1 | bigrams | 9 | C=100 |
| | RF | M1 | uni- and bigrams | 3 | B=100 |
| | | | | | min_samples_leaf=8 |
| | SVM | M1 | bigrams | 8 | C=10 |
| | XGB | M1 | uni- and bigrams | 8 | eta= 0.3 |
| | | | | | max_depth=6 |
| | | | | | colsample_bytree=0.7 |
| | | | | | subsample=0.6 |
| | | | | | n_estimators=200 |

[8] Parameter names in the relevant Python package/function. Detailed description in Table 1.

*3.2.2 User-wise statistical model*

Apart from the n-gram based models, we constructed a model based on statistical variables describing all hundred tweets of each author, thus giving one more prediction per author. The variables used in this model are as follows:

- the mean length of the 100 tweets of the authors both in words and in characters;
- the minimum length of the 100 tweets of the authors both in words and in characters;
- the maximum length of the 100 tweets of the authors both in words and in characters;
- the standard deviations of the length of the 100 tweets of the authors both in words and in characters;
- the range of the length of the 100 tweets of the authors both in words and in characters;
- the number of retweets in the dataset by each author;
- the number of URL links in the dataset by each author;
- the number of hashtags in the dataset by each author;
- the number of mentions in the dataset by each author;
- the number of emojis in the dataset by each author;
- the number of ellipses used at the end of the tweets in the 100 tweets of the authors;
- a stylistic feature, the type-token ratio to measure the lexical diversity of the authors (in the dataset each author has 100 tweets thus the number of tokens per author does not differ as much that it would cause a great diversity in the TTRs).

This gives a total of 17 statistical variables. Since we used an XGBoost classifier, we did not normalize the variables and the linear correlation between the variables posed no problem.

To find the best hyperparameter set, we used a five-fold cross-validated grid search and finally refitted the best model on the whole data. The cross-validated accuracies achieved this way are 70% and 74% for the English and Spanish data respectively. Table 3 contains the best hyperparameters found.

**Table 3:** The best model hyperparameters for the XGBoost model using statistical features

| Parameter name | Parameter values | |
|---|---|---|
| | **EN** | **ES** |
| Column sample by node | 1 | 0.8 |
| Column sample by tree | 0.9 | 0.8 |
| gamma | 2 | 4 |
| Learning rate | 0.2 | 0.3 |
| Max depth | 2 | 3 |
| Min child weight | 4 | 5 |
| Number of estimators | 200 | 100 |
| alpha | 0.1 | 0.3 |
| Subsample | 0.8 | 0.8 |

*3.2.3 Stacking ensemble*

After identifying the best hyperparameters for the five mentioned models with cross-validation, we had to find a reliable ensemble method. To avoid overfitting this ensemble model to the training set, we did not train it using the predictions of the five final trained models. Instead, we wanted to create a dataset that represents the predictions that are produced by our models. To do this, we refitted the five sub-models with the cross-validated hyperparameters five times on different chunks of the original training data (each consisting of tweets from 240 users). The predictions given by these five models to the 60 remaining users were appended to the training data of the ensemble model, thus this training set consisted of predictions given to all 300 users in the training data, but these predictions were given by five different models in case of each model type. The sample created this way can be interpreted as an approximation of a sample from the distribution of the predictions of the final five models on the test set. We created a test set with the same method but with a different split of the training data.

We then used these constructed training and test sets to find the best ensemble from the following three methods: majority voting, linear regression of predicted probabilities (this includes the simple mean), and a logistic regression model. The best and most reliable results were given by the logistic model; therefore, we used this model as our final ensemble method. Table 4 summarizes the logistic regression coefficients for the probabilistic predictions of each model for both languages.

**Table 4**: Logistic regression coefficients for the predicted probabilities by each sub-model

| Model | Coefficient values | |
|---|---|---|
| | **EN** | **ES** |
| LR | 0.8 | 1.31 |
| SVM | 0.48 | 1.16 |
| RF | 0 | 0 |
| XGB | 1.07 | 0.54 |
| Statistical XGB | 0.2 | 0.12 |

The validity of this method is backed by the fact that our results on the training sets (an accuracy of 75% and 81% for the English and Spanish set respectively) were only slightly better than the final test results.

## 4 Results

As mentioned in Section 3, we tested two versions of our software. For the early bird testing, we used the single best n-gram models based on our cross-validated grid search (a random forest classifier for the English set and a support vector machines classifier for the Spanish set). Using these models, we experienced a significant decrease in the accuracy of the models compared to their cross-validated performance, so this was one of the reasons why we decided to incorporate a number of different models for our final software. As Table 5 shows, relying on a number of different models and a statistically based ensemble method proved to be a good solution. First, the cross-validated accuracies of our final models were almost the same as their accuracies on the test set, and second, our final software was able to reach a higher accuracy in both languages than our early bird solution.

**Table 5:** Accuracies achieved by the two versions of our software during the cross-validation process and on the test set

| Language | Early bird software | | Final software | |
|---|---|---|---|---|
| | **CV (training set)** | **Test set** | **CV (training set)** | **Test set** |
| EN | 83% | 79% | 81% | 80.5% |
| ES | 75% | 69% | 75% | 75% |

## 5 Future Work

One of the unanswered questions that emerged during this project is concerning the reasons behind the fact that our models are better at identifying fake news spreaders that tweet in Spanish. This is true about all of our individual models regardless of the features they used, and about the final ensemble model as well. We assume that it would be beneficial to conduct some qualitative research about the tweets in the dataset to better understand why fake news spreaders that tweet in Spanish are more distinguishable from regular users than those that tweet in English.

Another promising direction for achieving higher accuracy in profiling fake news spreaders is to develop a software that is able to determine whether a single tweet should be considered as fake news. It is reasonable to assume that even those that are labelled as fake news spreaders only post some tweets that can be considered as fake news, while some of their posts are just regular tweets. Therefore, from the perspective of our approach, the current dataset is likely to contain a lot of noise. If we were able to identify fake news on the level of tweets, we could build a model relying on this information that would allow us to give predictions for each tweet. This approach was unfortunately not executable with the PAN20 Fake News Spreaders dataset [13], as it did not provide information about single tweets, and additionally, all URL links, hashtags and user mentions, which could have provided valuable clues about the credibility of the tweet, were replaced by standardized tokens in the text. Moreover, even if we had access to these tweets in their original form, manual labeling would be a tedious process even for the "small" dataset of 300 users. However, it would be interesting to investigate how a software that is able to decide whether a single tweet is fake news would perform in this task.

## 6 Conclusion

In this notebook, we summarized our work process of preparing a software for the PAN 2020 Profiling Fake News Spreaders on Twitter task [12]. Originally, we looked at a number of machine learning models using n-grams as features. To find the best parameters for the models, we conducted an extensive grid search combined with cross-validation. After finding the models achieving the highest accuracy during the cross-validation, we fitted these on the entire training set. However, we realized during the early bird testing phase that this approach results in a significantly lower accuracy on the test set compared to its cross-validation results. Therefore, for our final software, we decided to create a combined model which was a stacking ensemble of five sub-models. Four of these sub-models (a logistic regression, a support vector machine classifier, a random forest classifier and an XGBoost classifier) used word n-grams as features, while the fifth model (another XGBoost model) used statistical features extracted from the Twitter feed. For each sub-model, we used grid search and cross-validation to find the best performing parameters and fitted the models on the entire training data with these parameters. To get a final prediction for each user, we trained a logistic regression that used the probabilistic

predictions of the sub-models as features. Using the ensemble model, we were able to achieve the same accuracy on the test set as during the cross-validation process. Overall, our final software was able to identify fake news spreaders with a 75% accuracy among users that tweet in English, and with an 80.5% accuracy among users that tweet in Spanish. Our overall accuracy of 77.75% was tied as the highest performance in the competition.

## 7 References


1. Ahmed, H., Traore, I., Saad, S.: Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques. In: Traore I., Woungang I., Awad A. (eds) Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. ISDDC 2017. Lecture Notes in Computer Science, vol 10618. Springer, Cham (2017)
2. Aravantinou, C., Simaki, V., Mporas, I., Megalooikonomou, V.: Gender Classification of Web Authors Using Feature Selection and Language Models. In: Speech and Computer Lecture Notes in Computer Science, pp. 226–33. (2015)
3. Boulis, C., Ostendorf, M.: A quantitative analysis of lexical differences between genders in telephone conversations. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics - ACL '05. Morristown, NJ, USA: Association for Computational Linguistics, pp. 435-442 (2005)
4. Devlin, J., Chang, M., Lee, K., Toutanova K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL-HLT. (2019)
5. Garera, N., Yarowsky, D.: Modeling Latent Biographic Attributes in Conversational Genres. In: Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, pp 710-718 (2009)
6. Ghanem, B., Rosso, P., Rangel, F.: An Emotional Analysis of False Information in Social Media and News Articles. In: ACM Transactions on Internet Technology (TOIT) vol. 20 no. 2, pp. 1-18 (2020)
7. Gonzalez-Gallardo, C. E., Torres-Moreno, J. M., Rendon, A. M., Sierra, G.: Efficient social network multilingual classification using character, POS n-grams and Dynamic Normalization. In: IC3K 2016 - Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management. SciTePress, pp. 307-314. (2016)
8. Marquardt, J., Farnadi, G., Vasudevan, G., Moens, M., Davalos, S., Teredesai, A., De Cock, M.: Age and gender identification in social media. In: CLEF 2014 working notes, pp. 1129-1136. (2014)
9. Peersman, C., Daelemans, W., Van Vaerenbergh, L.: Predicting Age and Gender in Online Social Networks. In: Proceedings of the 3rd International Workshop on Search and Mining User-Generated Contents. New York, NY, USA: Association for Computing Machinery, pp. 37-44. (2011)
10. Potthast, M., Gollub, T., Wiegmann, M., Stein, B.: TIRA Integrated Research Architecture. In: Ferro, N., Peters, C. (eds.) Information Retrieval Evaluation in a Changing World - Lessons Learned from 20 Years of CLEF. Springer. (2019)

11. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D, Sutskever, I.: Language Models are Unsupervised Multitask Learners. OpenAI, San Francisco, CA, (2019)

12. Rangel F., Giachanou A., Ghanem B., Rosso P. Overview of the 8th Author Profiling Task at PAN 2020: Profiling Fake News Spreaders on Twitter. In: L. Cappellato, C. Eickhoff, N. Ferro, and A. Névéol (eds.) CLEF 2020 Labs and Workshops, Notebook Papers. CEUR Workshop Proceedings.CEUR-WS.org (2020)

13. Rangel F., Rosso P., Ghanem B., Giachanou A. Profiling Fake News Spreaders on Twitter [Data set]. Zenodo. http://doi.org/10.5281/zenodo.3692319 (2020)

14. Rao, D., Yarowsky, D., Shreevats, A., Gupta, M.: Classifying latent user attributes in Twitter. In: SMUC '10: Proceedings of the 2nd international workshop on Search and mining user-generated contents. Pp. 37-44. (2010)

15. Santosh, K., Bansal, R., Shekhar, M., Varma, V.: Author Profiling: Predicting Age and Gender from Blogs Notebook for PAN at CLEF 2013. in: Working Notes for CLEF 2013 Conference. (2013)

16. Sboev, A., Litvinova, T., Voronina, I., Gudovskikh, D., Rybka, R.: Deep Learning Network Models to Categorize Texts According to Author's Gender and to Identify Text Sentiment. In: Proceedings - 2016 International Conference on Computational Science and Computational Intelligence, CSCI 2016. Institute of Electrical and Electronics Engineers Inc., pp. 1101-1106. (2017)

17. Schler, J., Koppel, M., Argamon, S., Pennebaker, J.: Effects of Age and Gender on Blogging. In: AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs. American Association for Artificial Intelligence (AAAI), pp. 199- 205. (2006)

18. Stout, L., Musters, R., Pool, C.: Author Profiling based on Text and Images Notebook for PAN at CLEF 2018. In: Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum. (2018)

19. Volkova, S., Bachrach, Y.: On Predicting Sociodemographic Traits and Emotions from Communications in Social Networks and Their Implications to Online Self Disclosure. In: Cyberpsychology, Behavior, and Social Networking (Mary Ann Liebert Inc.) 2015/12, pp. 726-736. (2015)

20. Yildiz, T.: A comparative study of author gender identification. In: Turkish Journal of Electrical Engineering and Computer Science 27, pp. 1052-1064. (2019)

21. Zhang, X., Yu, Q.: Hotel reviews sentiment analysis based on word vector clustering. In: 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA), Beijing, pp. 260-264. (2017)