Classifying Disinformation by Domain Name

Trevor J. McGlynn

Western Governors University

**Table of Contents**

[Summary 3](#_Toc441471473)

[Review of Other Work 3](#_Toc441471474)

[Changes to the Project Environment 3](#_Toc441471475)

[Methodology 3](#_Toc441471476)

[Project Goals and Objectives 3](#_Toc441471477)

[Project Timeline 4](#_Toc441471478)

[Unanticipated Requirements 4](#_Toc441471479)

[Conclusions 4](#_Toc441471480)

[Project Deliverables 4](#_Toc441471481)

[References 6](#_Toc441471482)

[Appendix A: 7](#_Toc441471483)

[Appendix B: 8](#_Toc441471484)

[Appendix C: 9](#_Toc441471485)

[Appendix D: 10](#_Toc441471486)

# Summary

The purpose of this project was to see with what accuracy a domain name could be classified as disinformation using only publicly-available information. A list of domain names was curated from several sources: legitimate domains were scraped from Alexa web rankings, while domains classified as sources of disinformation were curated from per-compiled reports from librarians at Merrimack College, the New York Times, security and computer science researchers, and other sources, which were repeatedly validated with set intersections throughout the curating process. From these domain names and their applied trust ratings, WHOIS data was queried, analyzed, and summarized, with clustering of connected websites validated through Google Analytics codes scraped from HTML head tags. The data were fed through a random forest classifier (RFC) which resulted in a 99.4% accurate classification against its test data—far exceeding expectations.

Data collection started with data sets currently available on Kaggle.com, all of which were curated for the intention of spotting influence operations on the web. Per the requirements for the project, the most interesting attribute were domain names in these data sets. Next, a recently released data set from the New York Times which included fake local news websites was brought in, followed by web scrapes and a PDF scrape of confirmed sources of disinformation from Merrimack College.

Alexa web rankings fed the data marked “initial trust”. An assumption was made that trustworthy websites, for the most part, filter themselves over time. Due to internet connection errors here in northern Vermont during the querying, the sample size was reduced from n = 60,000 to n = 25,301; this was still an acceptable size for the purposes of the project. Domains which were labeled “fake” were validated and audited against this sample. 32 domains were re-categorized based on the results—this involved partial bias on my part, and for some domains the learned function ended up classifying some of them back to their original state.

Once the domains were collected, scripts were written to query and process whois data tied to the domains. Attributes collected included: the registrar, number of name servers, and update, creation, and expiration information. From these, date information were analyzed (such as the age of a domain in days). Domain names split into keywords and were calculated for their total length, number of keywords, and were scored according to what keywords were used in the domain and weighed against the frequency of those keywords in the fake domain sample. Registrars were scored in a similar fashion.

During this stage, each domain that was labeled “fake” was also scraped to see what, if any, Google Analytics codes they may have been using. This revealed the intricate but obvious connections among domestic influence operations were shell “local news” websites are created to drive narratives (all of these sites were pro-Trump, pro-GOP and none of them are legitimate media sources).

After collecting the data, an RMarkdown notebook was created to explore the finished data set. During this stage the data were again validated, and relationships were exposed which revealed the strengths of the domain name analysis and metrics computed during the collection process. This step provided meaningful insight into the data and prepared for the success seen during the modeling step.

A RFC was chosen as a model for the machine leaning aspect of the project, due to its accuracy in classification problems with multiple features. The final features chosen were the following: the length of the domain name, the length of keywords in the domain, the number of name servers, the age of the domain (in days), the computed domain name score, the computed registrar score, and the domain’s Alexa web ranking. The classifier reported an accuracy rate of 99.4%, which exceeded the expectation of 93% set in the proposal.

# Review of Other Work

In this section, provide an expanded review of the Review of Other Work section in task 2, including three additional third-party artifacts on the topic that supported the development of the project, and explain how the artifacts supported the implementation.

https://www.nytimes.com/2020/10/20/technology/timpone-network-pay-to-play-local-news.html?action=click&module=Top%20Stories&pgtype=Homepage

https://www.nytimes.com/2020/10/18/technology/timpone-local-news-metric-media.html

https://library.southtexascollege.edu/false-misleading-clickbait-y-and-satirical-news-sources/

This module segments text according word frequency using the Viterbi algorithm. Probably

due to Peter Norvig somehow.

Three sources of frequency information is provided.

The third is from a webcrawl dataset of anchor text provided

by Vinay Goel of the Internet Archive.

<https://towardsdatascience.com/understanding-random-forest-58381e0602d2>.

A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.

uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. The reason for this wonderful effect is that the trees protect each other from their individual errors

# Changes to the Project Environment

The project environment has worsened since the project began, at least under the context that disinformation on the web has probably gotten worse. This project, however, does entertain a new way of identify disinformation using only data that is publicly available, and one that apparently works quite well. Most research investigated during the proposal required information given by social media companies; this project addresses domain-based disinformation that is entirely OSINT.

That said, this project does offer a promise of another solution for the problem, but not a terribly meaningful one. It best uses would be in finding coordinated domestic fake news operations who on the web, but not on social media.

# Methodology

This project used a waterfall methodology. In the first phase, requirements were established and data were gathered. Following data collection, the data were tidied and summarized to make analysis easier. Following data munging, the data were explored and analyzed to audit, explore, and understand its underlying features.

The next stage was system design, including detailing entity relationships and reducing the dimensionality of the data. During this stage, research was also done on how to best model the problem for a ML implementation.

Following design, the learned function was developed and verified against the test data and the optimization of its hyper-parameters.

The last stage is interpretation and communication of results, with emphasis on documentation. Data sets are accompanied with detailed data sheets explaining the purpose of the curation, sources used, and detailed information about the data contained within. An RMarkdown notebook was published showing some of the findings within the data set.

# Project Goals and Objectives

## Goal 1: Collect & analyze the data

The first goal was to curate data sets finding various sources of fake news. Data sources collected include governmental research, data sets publicly posted on Kaggle.com, fact-checking organizations, and a few custom web scrapes. After extraction, the data were transformed and cleaned, then loaded into .csv files—this completed objective 1.a. After this process was complete, an exploratory data analysis took place, and an RMakdown notebook was published to explore and validate the data set. This step completed objective 1.b. Objective 1.c., which included database design and modeling, was completed and normalized to 2NF.

### Goal 2: Implement a supervised binary classification algorithm

The second goal and its sole objective was the implementation of the machine learning algorithm. The deliverable was the completed algorithm. Research in the project proposal set a benchmark of >90% accuracy with its classifications. Two ML implementations classify disinformation, though with much different applications, achieved a 93% accuracy in classification. In the end the RFC algorithm for this project produced a 99.4% accurate classification rate, far exceeding expectations. A much broader data set, and some additional validation steps, would produce a more meaningful result, but if one thing is clear, this implementation could be extremely successful in classifying domain-based disinformation campaigns using only OSINT. Implementing the ML algorithm completed objective 2.a.

### Goal 3: Document & make publicly available

The final goal was aligned with the communication step of the data science process: results were published and discussed, documentation was written and made available, and a plan for implementation is discussed. Objective 3.a. governs the publishing of all content created during the project and their publication with appropriate data sheet. All source code is available on Github with an accompanying README.md file. Github also hosts the cleaned data sets. Objective 3.b. focuses on post-implementation planning, which discusses further implementation or improvement of the algorithm, its uses and intentions, and lessons learned during execution of the project—this is included in the README.md file file on Github.

# Project Timeline

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Milestone or deliverable | Duration  (hours or days) | Projected start date | Anticipated end date | Actual end date |
| Definition of requirements and data collection | 2 days | 10/19/2020 | 10/20/2020 | 10/21/2020 |
| Data tidying and analysis | 2 days | 10/22/2020 | 10/23/2020 | 10/28/2020 |
| Database design & modeling | 6 hours | 10/26/2020 | 10/26/2020 | 10/29/2020 |
| Implementation of the ML algorithm | 2 days | 10/27/2020 | 10/29/2020 | 10/30/2020 |
| Writing of documentation & uploading content | 1 day | 10/30/2020 | 10/30/2020 | 10/30/2020 |
| Post-implementation write-up | 1 day | 11/2/2020 | 11/2/2020 | 10/31/2020 |

The project ended ahead of schedule, though specific work units missed their anticipated end dates. The first milestone, on data collection, took 1 day longer than estimated. Then, the data analysis and data tidying milestone (and its RMarkdown notebook), took five days to complete (instead of two). This was drastically underestimated; it is said that data analysts spend up to 80% of their time cleaning data, and it that certainly applies here. Implementation of the ML algorithm only took one day (instead of two), and the documentation and diagramming steps moved a little faster than expected.

# Unanticipated Requirements

One unanticipated requirement was a reliable internet connection. During the data collection phase, scripts were written which required repeatedly querying, and locally processing, tens thousands of domains from a DSL connection in northern Vermont. Even left to run over night, this would have been made faster, and with wider reach, with a better internet connection. Little could be done about this (especially due to COVID restrictions during the project environment), though it did make technology, especially network, limitations a consideration for any project in the future.

# Conclusions

The learned function form the ML algorithm was correct in its classifications 99.6% of the time—in other words, it was wrong 4 times per 1,000. This number is the average correct classification rate for both the test and validation sets (combined n = 10,120). False positives occurred more often than false negatives with both sets. Due to the nature of the problem, false negatives would ideally be minimized—for example, misidentifying legitimate health information as disinformation, and in both instances this was the case. The algorithm’s accuracy suggest potential a fairly accurate model for classifying domain-based disinformation, *especially* if its outputs are being validated by human supervision.

Overall, the project was successful because its classification algorithm performed >90% (99.6%) on a sample n > 10,000 (n = 25,299).

# Project Deliverables

In the Project Deliverables section, explain and detail the project key deliverables. The actual project development will be documented by the key deliverables. The project includes some sort of formal report. The deliverables should provide a detailed logical explanation of what the project provided to substantiate the work and completion of such. Describe the artifacts being used to show evidence of the project’s completion and use the appendices to include the actual artifacts. Actual project artifacts may include code samples or screen shots; flowcharts, UML, or other process diagrams; charts, tables, and graphs; network diagrams (before and after); training materials; and/or the technical IT product itself.

### Data sets

Two data sets were produced: **datasets/fake\_real\_domains\_combined.csv**, which is the final data set that was fed into the learned algorithm, and **datasets/fake\_domain\_word\_freq.csv**, which was used to evaluate the domain name scoring.

### Exploratory data analysis

### **Binary Classification Algorithm**

The source code for the classification algorithm is found in the Jupyter notebook file classifier.ipynb.

**domains\_eda.html**

# 

# References

List all the outside sources that the narrative refers to in-text. For in-text and reference list citations, please refer to the web link in or visit the WGU Writing Center.

Smyth, A. M., Parker, A. L., & Pease, D. L. (2002). A study of enjoyment of peas. Journal of Abnormal Eating, 8(3), 120-125. Retrieved from

http://www.articlehomepage.com/full/url/

Bernstein, M. (2002). 10 tips on writing the living Web. A List Apart: For People Who Make Websites, 149. Retrieved from http://www.alistapart.com/articles/writeliving

Bell, T., & Phillips, T. (2008, May 6). A solar flare. Science @ NASA Podcast. Podcast retrieved from http://science.nasa.gov/podcast.htm

OLPC Peru/Arahuay. (n.d.). Retrieved April 29, 2011 from the OLPC Wiki: http://wiki.laptop. org/go/OLPC\_Peru/Arahuay

Plath, S. (2000). The unabridged journals. K. V. Kukil (Ed.). New York, NY: Anchor.

# Appendix A

# Title of Appendix

Put any supporting material in these appendices. Add additional or delete superfluous appendices as needed.

# Appendix B

# Title of Appendix

Put any supporting material in these appendices. Add additional or delete superfluous appendices as needed.

# Appendix C

# Title of Appendix

Put any supporting material in these appendices. Add additional or delete superfluous appendices as needed.

# Appendix D

# Title of Appendix

Put any supporting material in these appendices. Add additional or delete superfluous appendices as needed.