CSEC 620 CYBER ANALYTICS AND MACHINE LEARNING

PROJECT 3: MACHINE LEARNING FOR SECURITY - PHISHING

November 30, 2024

Nikhil S. Patil
Kamron J. Cole
Graham G. Rogozinski
Patrick T. Elser
Department of Cybersecurity
College of Computing and Information Sciences
Rochester Institute of Technology
nsp4746@rit.edu
kjc8084@rit.edu
ggr4460@rit.edu
pte6144@rit.edu

I. Abstract

This section provides on overview about the project. It should be completed at the very last stage of the writing, i.e., after you have completed all the other sections.

II. Introduction

In this section, you need to address the following items:

- The goal of this project
- The rationale of the project
- The importance of the project
- Your main contributions to the project

III. Literature Review

- How, when the problem was raised by whom, what event, etc.
- Any attempts have been proposed to attack the problems
- Pros and Cons of each existing solution
- How does your proposed solution fit within here, i.e., your solution is improving an existing solution, just another solution, or revolutionizing the way of thinking?

Make sure to cite references. Here is how to cite the great textbook written by Knuth [1] in LaTeX. Here is the way to cite a journal article written by the father of computer science Alan Turing [2].

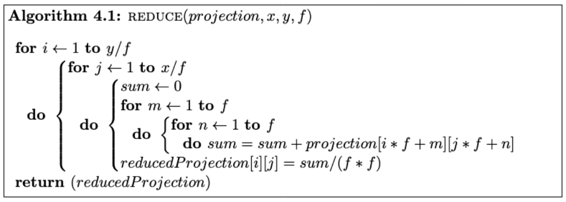
IV. Your project idea/description

In this section, first, we redefine the problem in great detail. Then, describe your solutions. Try to use as many illustrations, such as pictures, graphs, and pseudo-codes, as you see fit. Here are some examples of including pictures, graphs, and pseudo-code in latex.

Figure 1 shows how to include a picture in a Latex document.

Figure 1. This is my great security design.

Figure 2 is a sample pseudo code copied from this site [3].



```
Algorithm 4.1: REDUCE($projection, x, y, f$)

for $i \leftarrow 1$ to $y/f$
  do
    for $j \leftarrow 1$ to $x/f$
      do
        $sum \leftarrow 0$
        for $m \leftarrow 1$ to $f$
          do
            for $n \leftarrow 1$ to $f$
              do $sum = sum + projection[i * f + m][j * f + n]$
        $reducedProjection[i][j] = sum/(f * f)$
return ($reducedProjection$)
```

Figure 2. This is my great pseudocode

Figure 3 is a sample float chart copy from this website [4]

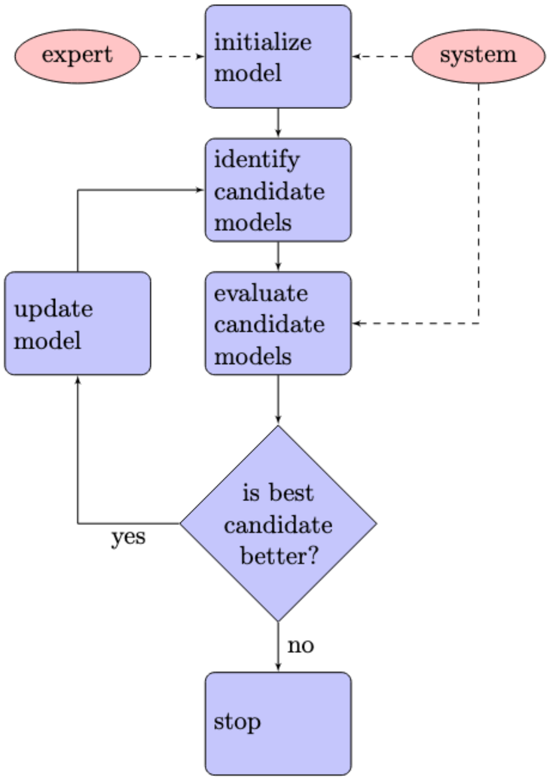

expert
initialize model
system
identify candidate models
update model
evaluate candidate models
is best candidate better?
yes
no
stop

Figure 3. This is my great float chart

A. Mathematics

If your project idea needs mathematics to formulate your methods, LaTeX is great at typesetting mathematics. Let $X_1, X_2, \ldots, X_n$ be a sequence of independent and identically distributed random variables with $\mathrm{E}[X_i] = \mu$ and $\mathrm{Var}[X_i] = \sigma^2 < \infty$, and let

$$S_n = \frac{X_1 + X_2 + \cdots + X_n}{n} = \frac{1}{n} \sum_{i}^{n} X_i$$

denote their mean. Then as n approaches infinity, the random variables $\sqrt{n}(S_n - \mu)$ converge in distribution to a normal $\mathcal{N}(0, \sigma^2)$.

V. Project Implementation

In this section, describe details of your project design and implementation, document challenges and obstacles, and how you overcome them in design and implementation.

VI. Testing and Experiments

In this section, describe your testing and experiment design and setup, and conduct the testing and experiments, and generate data.

Need to explain why your experiment design will do what is supposed to do, and describe the expected result, and how the result may validate your ideas and/or support your project.

VII. Data Analysis

A. Performance Metrics

To interpret the results of the 3 models tested, it is important to first describe the various metrics used when evaluating a decision making system: precision, recall, F1 score, and accuracy. While accuracy is the easiest to understand, the ratio of the successes to all attempts, it does not provide as much insight into reliability. If a particular user encounters more spam than another, perhaps due to their email address being involved in a breach, then the same system may show a wildly different accuracy than that of a user receives very little spam. Many categorization systems will either handle false positives or false negatives better than the other, which is not reflected in the accuracy.

Instead, precision and recall are used to describe these facets of our detection system. Precision refers to, out of times where the system predicted a URL to be spam, the URL was indeed spam. Poor precision occurs when results identified as spam aren't usually spam, such as when benign URLs are misidientified more often the spam URLs are identified. If testing data contains significantly more safe URLs then spam URLs, precision is bound to be low, whereas precision is usually higher in datasets containing more spam than safe URLs. Recall instead refers to, out of the times where a piece of spam is evaluated, how frequently it is identified as such. Because recall only refers to data that is spam, the amount of spam relative to safe URLs in the dataset is not relevant, and recall is independent from that aspect of the sample. The F1 score is a single metric that reflects both the precision and recall, which allows for both to be gleaned from a single statistic.

In addition to these metrics, which are commonly used to evaluate the performace of AI categorization models, it is also important to discuss performance metrics commonly associated with binary systems like this, such as false positive and false negative rate. Unlike precision and recall, both of these metrics are independent from the distrobution of the testing sample, allowing them to be more universally applicable. The false negative rate refers to, out of the URLs that are truly spam, how many were misidentified as safe; whereas the false positive rate refers to, out of the URLs that are truly safe, how many where misidentifies as spam.

There are two main reasons to split out false positive and false negative rates: to allow for predicting and simulating accuracy in systems given how much spam they are expected to see and to prioritize one rate over the other. For instance, if falsely identifying spam as safe just provides the receiver mild annoyance but falsely identifying a safe message as spam causing a critical message to be lost, than a system with a lower false positive rate would be desirable. However, if the system is just flagging messages as spam instead of hiding them, lower false negative rates might be more desireable.

B. Results

Of the three methods, using traditional AI performance metrics seen in figure 4, the random forest algorithm consistently outperformed the neural network, which in turn outperformed the K-Nearest Neightbor classifier. Across all methods, precision was at or above the recall, suggesting the models perform better with data that is safe rather than spam. Regardless, all of the metrics are close, and none of the methods have substantially poor performance in an area.

The false positive and false negative rates from figure 5 show more descriptive data for these algorithms, though. For all three algorithms, the false positive rate is lower, again showing that the models do better on safe data. The random forest is much less susceptible to this trend, but still has a false negative rate over 1% worse than its false positive rate. Whether or not this behavior is desireable depends on the particular implementation being used.

One naive way to improve the output of the algorithms would be to take input from all three. However, this would both incur a performance penalty and not result in significant improvement. If the three models were independent, such a technique could help, but they are not independent. URLs that are tougher for one model are likely also harder for another, as these harder URLs likely have some features of both safe and spam URLs. Rather, investigating these edge cases and improving one of the models has more potential than simplyed putting them together.

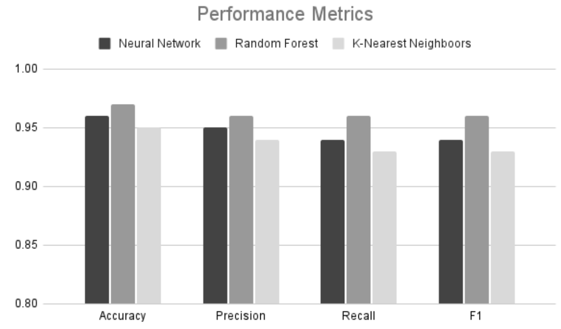


Figure 4. Performance results of the three models

C. Usage

Using the URL's found in received emails is the primary use case for these models. More specifically, emails identified as spam may either be flagged to alert to user to the suspicious nature of the email or be outright discarded or hidden from the user. These two use cases have opposite needs with regard to false positives and false negatives. A legitimate email that is falsely identified as spam and thrown out could potentially cause serious issues for the sender or attempted recipient, leading to a need to lower false positives to near zero. Alternatively, if all the system does is add a banner at the top of the message warning the user, false positives are not harmful, and the potential damage from falling for

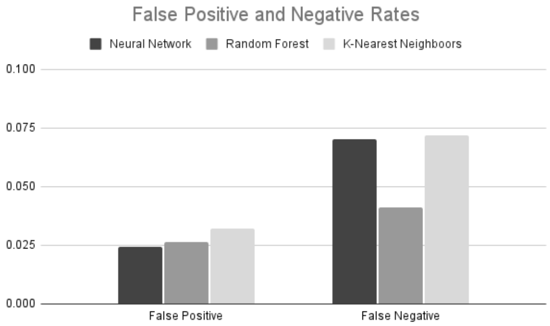


Figure 5. Performance results of the three models

a phishing attempt causes the balance to fall in the other direction.

As all of the discussed models have low error rates and very low false positive rates, more aggressive methods of hiding spam may be employed. Larger email providers can combine these systems with traditional methods, such as whitelists and blacklists for known good and bad domains, to help alleviate the remaining bad cases. These traditional methods generally perform worse on newer domains, leading to the concept of domain aging. Defering to AI to determine the trustworthyness of relatively young domains may help prevent false positives from recent registrations.

VIII. Conclusions

In this section, you should provide a concise summary of your work, and state the importance of your idea and your contribution towards solving the problem. Point out any improvement that could be done if you had more time. List some ideas for future work.

IX. Acknowledgment

I thank my advisor for the guidance and I appreciate the financial and emotion support from my family and sponsors, ...

References

[1] D. E. Knuth, *Art of Computer Programming, Volume 4, Fascicle 4, The: Generating All Trees–History of Combinatorial Generation*. Addison-Wesley Professional, 2006.

[2] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[3] S. Smallen, "Pseudo code." [Online]. Available: http://users.sdsc.edu/ ssmallen/latex/pseudocode.html

[4] K. M. Fauske, "Example: Simple flow chart." [Online]. Available: http://www.texample.net/tikz/examples/simple-flow-chart/

X. Appendices

This includes all source code, scripts, and any material that helps other people replicate the results.