

Project Overview: Describe the dataset, the problem being addressed, and the specific cybersecurity context.

Phishing is the most effective way to gain initial access to an organization. Multiple studies have shown that in most ransomware and data loss incidents, phishing was the first attack vector. Furthermore, phishing accounts for billions of dollars lost every year as a result of data breaches. This problem has only grown over the years, and attacks have started incorporating AI to draft more sophisticated emails to get into organizations. Although a rule or filter-based email filtering platform solves some issues, a lot of spam and phishing attacks sneak through. It is critical that modern email solutions incorporate some form of AI to enhance the detection of these threats. These solutions also must be able to adapt to new attacks, as every new phishing campaign does something a little different - whether it be a new persuasion technique to entice a click or download, or exploiting some part of an email client/environment. As I do not have access to a dataset of phishing emails, I have found one online that specifically targets URL phishing, which is one of the most common tactics to achieve an account compromise and thus initial access. In this attack, an actor tries to get a user to click a link in an email, which leads to them either providing their credentials to a fake login page, having their cookies stolen, or downloading malware. The goal of this project is to find a model that best identifies phishing URLs. The dataset contains a set of URLs, from which it determines a variety of attributes, such as the length of the URL, the number of dots, hyphens, and more. It also includes attributes such as punycode, which is a common technique used in phishing attacks where a URL is slightly changed by changing a character(s) to another similar one (such as a Latin 'o' to a Cyrillic 'o'). This slight change is indiscernible to end users, resulting in many attacks utilizing this technique. As phishing is the leading cause of compromise, developing strong models to detect it is necessary to secure modern environments.

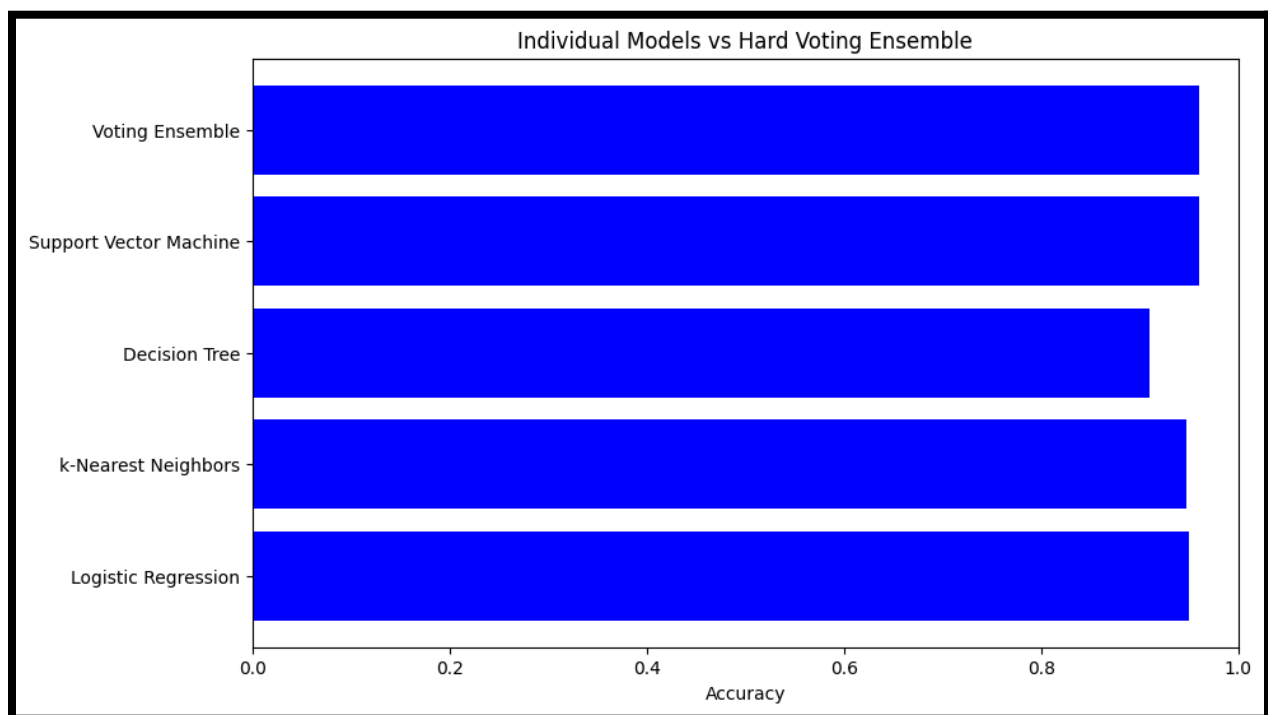
Methodology: Explain the chosen models and rationale for each.

I used four supervised learning models: Support vector machine (SVM), logistic regression, kNN, and decision tree. I also used two ensemble methods: Soft ensemble learning with voting and bagging. SVM is best for capturing nonlinear relationships, which are exceptionally common across phishing datasets. Logistic regression, although linear, is still a strong model for detecting phishing URLs. This is due to the fact that specific features of a suspicious URL may be easy to detect, such as Punycode characters or excessive periods. This model is also easy to follow, or identify why a message may be found malicious. kNN is a very effective model in phishing detection as it groups data based on similarity to known attempts. As

phishing campaigns are very similar to each other, this is an effective technique. The downside of this method is that it may not be able to detect new and differing techniques. A decision tree model can be incredibly effective in phishing detection. It is also very easy to see why a message may be classified as malicious or not, based on its feature thresholds. Voting and bagging techniques integrate the strengths of each model, usually leading to increased detection rates.

Results and Analysis: Present evaluation results with tables or graphs and discuss model performance. Include insights from additional metrics for a comprehensive comparison.

As shown below, all the models performed well. The bagging method, not shown in the graph below, was the most accurate with an accuracy score of 97%. Bagging had the highest recall as well, which, in a phishing detection scenario, is arguably the most important metric, as minimizing false negatives is crucial to securing a system or network. Decision Tree struggled with accuracy in comparison to other models; however, this may be a result of the preprocessing steps, which benefit the other models more.



Conclusion: Summarize key findings, including the effectiveness of ensemble methods and any potential recommendations for future work.

The bagging ensemble method was effective in combining models to enhance accuracy. It led to an approximately 1% gain over the next best model. SVM was the best model on its own; however, if one were to identify proper hyperparameters and preprocess specifically for a particular model, such as a decision tree, the results would be different. Unfortunately, attackers may easily bypass [current] AI-enhanced detection. I have personally experienced this as attackers will send a document, for example, that comes from a trusted source, such as sharepoint.com, with a seemingly legitimate partner of ours; however, within that document, lies a URL that is malicious and cannot be scanned by many email solutions; therefore, it is let through. A strong anti-phish solution must contain banners, be able to document that are provided via URL by abusing legit services, among other prevention mechanisms. Email security is pretty complex. AI can be used across many parts of a message to determine its legitimacy, such as scanning a URL, an attachment, or determining a fake invoice or persuasion techniques through a bag-of-words model, and so on. For a fully encompassed solution or future research, make sure to understand that this dataset is not the ultimate solution. An effective solution would also be able to incorporate new email campaigns marked as phishing (or not) to further improve the dataset. Hyperparameters are important; more research into what's best for each specific model is required for optimal results.

Dataset: <https://www.kaggle.com/datasets/shashwatwork/web-page-phishing-detection-dataset>