Using Decision Trees to Identify Phishing Sites

Lherisson Medina

College of Engineering, ECES-T480-002

Drexel University

3141 Chestnut St, Philadelphia, PA 19104

06/15/2017

Abstract

Every day more and more individuals and organizations become susceptible to phishing attempts through email links and websites they may visit. In a 2016 study, an estimated 85% of companies were hit with phishing attacks [5]. It is therefore necessary to apply the concepts of Machine Learning and Pattern Recognition to solve the problem of phishing. Determining whether a website is part of a phishing attempt presents us with a textbook classification problem. In this report, I propose a machine learning based approach to classification through the means of Decision Trees. It was found that this approach obtained an 89.3% accuracy when classifying websites as either legitimate, suspicious, or as part of a phishing attempt. We further evaluate our results by comparing the classification approach with other approaches such as tree ensembles and Support Vector Machines among other classifiers. Finally, the approaches outlined in this paper were compared against similar work done by other members in the field. Their results showed a higher accuracy using more advanced machine learning methods.

Introduction

The affects of a successful phishing attempt are long term and can ruin an individual's economic and social life for many years. A successful phishing occurs when a unknowing user willingly provides information to a false online entity. The information a phishing website could potentially extract can include usernames, passwords, and credit card information [6]. In May, millions of users of Google emailing service Gmail were hit with a sophisticated phishing attack which took most of 24 hours to identify and fix. Because of its consequences, it is important to recognize when a site may be attempting to phish data.

Over the years, phishing attacks have become more sophisticated. Services that allow websites to self certify and deploy Secure Sockets Layer (SSL) Certificates make it more difficult for internet browsers to discern who to trust; and advancements to the unicode standard make homograph attacks — described below — more possible.

Key Terms

Phishing - A phishing attempt with respect to this paper is defined as an email or website masquerading as coming from a legitimate source for the purpose of extracting personal information from an unknowing user.

Unicode - Since 1987 Unicode has been a computing industry standard for encoding and normalizing characters in text. It consists of character code tables for 135 scripts and over 100,000 characters. [3]

Homograph Attack - Domain names have historically been represented using ASCII characters in the Domain Name System (DNS). Due to the need for more localized URLs in other countries and languages, Punycode was developed to easily represent Unicode characters using the limited set found in the ASCII standard; known as Internationalized domain names. The use of Unicode ultimately allows character homographs, or lookalike characters, to be used to spoof users into thinking they are accessing a desired website. [3]

SSL Certificate - In public key infrastructure, SSL Certification Is a way to identify the legitimacy of a website by way of a public key. A Certificate Authority (CA) issues certificates which contain information about the owner of a site and the CA. SSL Certificates are a primary way for browsers to legitimize access to websites and allow for secure transport of internet communications.

Experimental Results

Methodology

A supervised learning, multinomial classification, Decision Tree based approach was conducted for this project. The dataset was downloaded from the University of California, Irvine Machine Learning Repository sourced from Neda Abdelhamid at Auckland Institute of Studies. The data is comprised of attributes for 1353 websites classified as being either phishy, suspicious, or legitimate. These attributes make up the feature-set used in training and evaluating the performance of our classifier. Out of the 1353 sites, 702 are phishing sites gathered by the source from the Phishtank Database — a community run repository of known and verified phishing websites — while 548 sites are legitimate website. The rest represent websites that are deemed suspicious for having many characteristics of phishing but not sufficient to classify it as so.

Out of 1353 sites, 25% of them were randomly removed from training and instead used to test our Decision Tree classifier. Experiments where the total data was randomly split 50% for testing

and 75% for testing were also conducted as a means to understand how the number of training data affected the accuracy of our classifier. The results these experiments are shown in **Figure 5**.

Other classification mechanisms where used to compare the performance of the Decision Tree. These include K-Nearest Neighbors, Random Forest Tree Ensembles, Support Vector Machines (SVM), and Multilayer Perceptrons (MLP). Figure 3 describes the performance of each classification method against our Decision Tree in terms of its accuracy. Python and the Scikit-learn machine learning library were used to create and evaluate our classifiers.

Feature Selection

Nine features were available to study which represented attributes for an arbitrary website. These features are: SFH, popUpWindow, SSLfinal_State, Request_URL, URL_of_Anchor, web_traffic, URL_Length, age_of_domain, and having_IP_Address. Each feature is characterized by having one of three values: Phishy, Suspicious, or Legitimate represented with integers -1, 0, and 1 respectively. The methodology for obtaining these numbers are described in Neda Abdelhamid's paper but a brief description of the most important attributes is given below.

Feature Selection involved picking the most important features based on how likely they are to help make a successful prediction. Feature Selection also helps pick out redundant features that can make training times slower without impacting the overall result. The features selected for training

The approach for feature selection involved finding each feature's Gini Importance. The Gini Importance for a given feature — also known as the Mean Decrease in Impurity — is calculated by the sum over the number of splits that include the feature, proportional to the number of samples that feature splits [4]. The derivation of the Mean Decrease in Impurity is:

$$G = \frac{1}{N_T} \sum_{T} \sum_{t \in T : v(s_t) = Xm} p(t) \Delta i(s_t, t) \qquad (1)$$

where:

$$p(t) \Delta i(s_t, t) \qquad (2)$$

Is the weight impurity decrease.

Figure 2 shows the Gini Importance for all nine features; collectively these values add up to 1.0. We can see that the Server Form Handler was the feature that was the most informative in

evaluating how risky a website is. This result makes intuitive sense, as processing of potentially sensitive data should most commonly be handled by the corporation responsible for the website.

In order to increase the performance of our Decision Tree, features deemed unnecessary had to be extracted from the experiment. Performance for the number of features can be evaluated by **Figure 3**. We see that for a Decision Tree, choosing the top 6 features resulted in higher performance compared to all the features, or too few. During the evaluation process it was also seen that, in terms of a Decision Tree, overfitting was experienced when more features were included.

Figure 4 further supported the argument that 6 features optimized the performance of all classifiers. Therefore we

Feature Description

Below is a summary of the methodologies used by Neda Abdelhamid et. al for evaluating the 4 features we deemed most informative through our feature selection process. A full list and descriptions for each feature can be found in their research paper sited on the references page.

Server Form Handler (SFH) - If when a user submits a form and the information is handled by a server from the same domain as the website the attribute is legitimate; if the form is handled by a server at a different domain the attribute is suspicious; and if the form handler field is left blank the attribute is phishy.

URL_of_Anchor - The feature is legitimate if less than 31% of links within the website point to a different domain; suspicious if less or equal to 67%; and phishy if more than 67% of links points to a site outside the webpage domain.

SSLfinal_State - If the SSL Certificate is 2 years or older and was issued by a Trusted CA, the website is likely to be legitimate. If the Certificate was issued by an untrusted CA the attribute is suspicious else if the website does not have a Certificate the attribute is phishy.

popUpWindow - The attribute is phishy if the user must submit a form through a pop up window, suspicious if any pop up window alerts the user, and legitimate if no popup appears.

Evaluation of Performance

The performance of our classifiers were evaluated by calculating their accuracy when predicting whether a website is legitimate, suspicious, or a phishing attempt. An accuracy score was calculated by comparing the prediction against the ground truth; the ground truth being the expected results for the subset of the entire dataset used for testing.

Figure 6 illustrates the accuracy of the Decision Tree vs those of K-Nearest Neighbors with 10 Nearest Neighbors, Random Forest Tree Ensembles, Support Vector Machines (SVM), and Multilayer Perceptrons (MLP). We observe that a Decision Tree, Random Forest, and Multilayer Perceptron were among the most accurate classifiers tested with an average of about 89% accuracy. The Multilayer Perceptron approach had the most accurate results of the three. Overall the Decision Tree approach faired favorably amongst all classifiers.

The Hamming Loss — the number of wrong labels over the total number of labels — for each classifier was also computed by the formula:

$$\text{HammingLoss}(x_i, y_i) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{xor(x_i, y_i)}{|L|} \tag{3}$$

Figure 7 shows the respective Hamming Loss for each classifier. Again the Decision Tree had a lower hamming loss than most, though was bested by its ensemble counterpart, the Random Forest Trees.

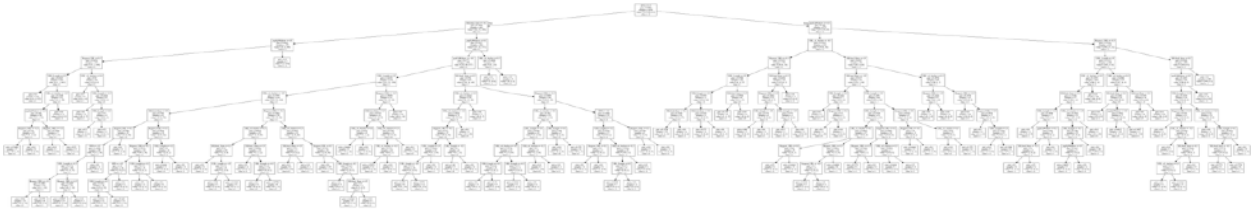

Figure 1: Decision Tree

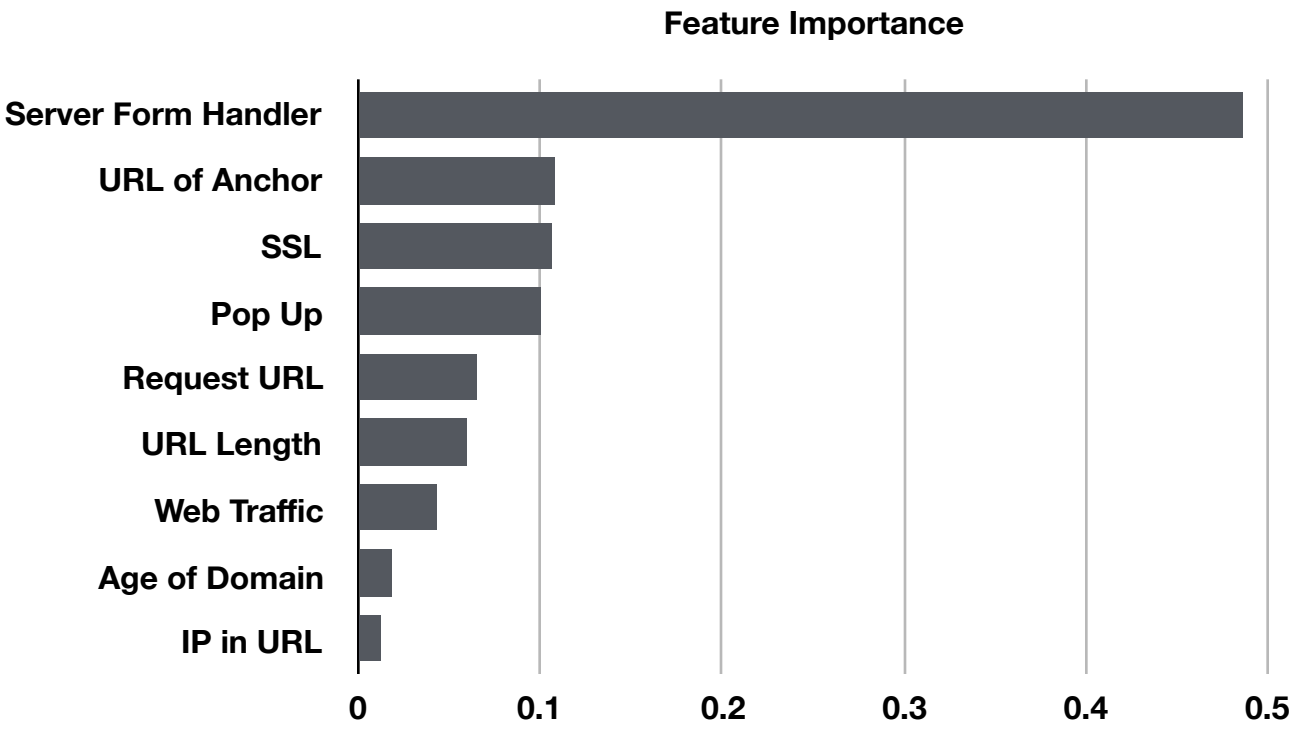

Figure 2: Mean decrease in Impurity for each feature

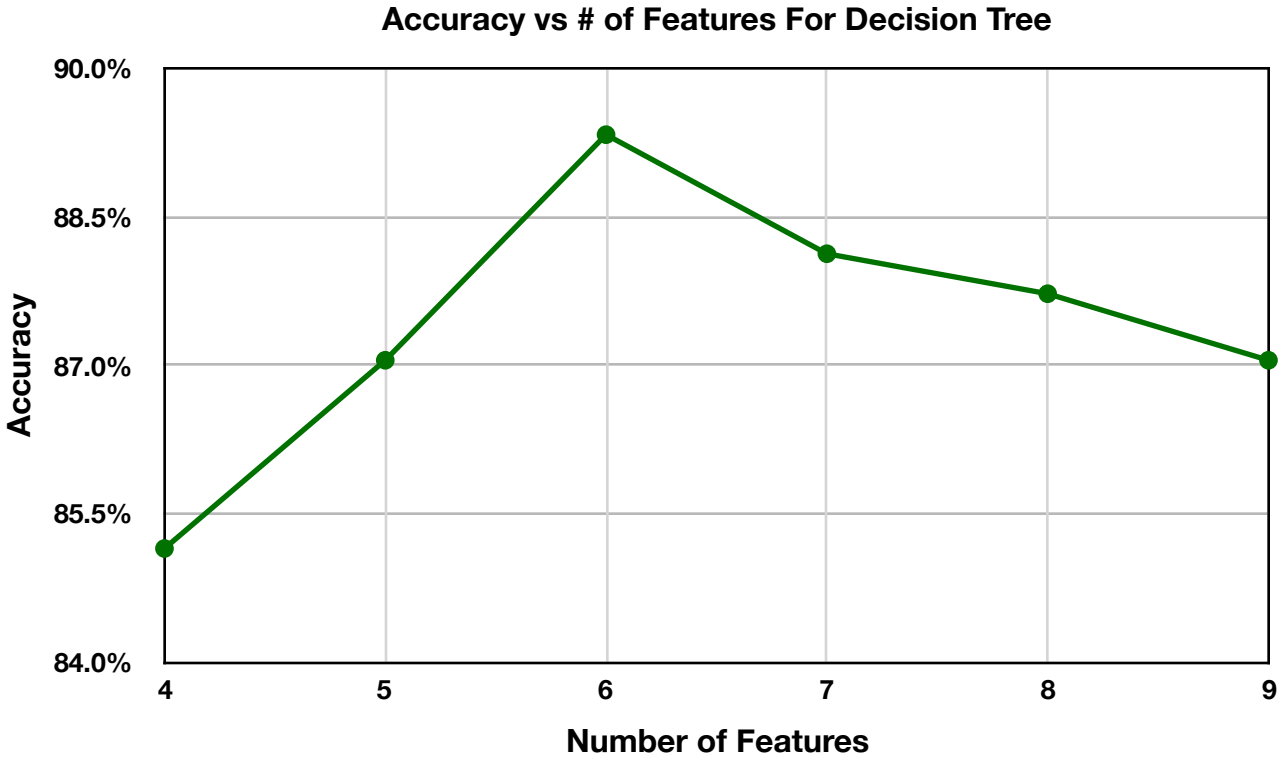

Figure 3: Result of Feature Extraction for Decision Tree

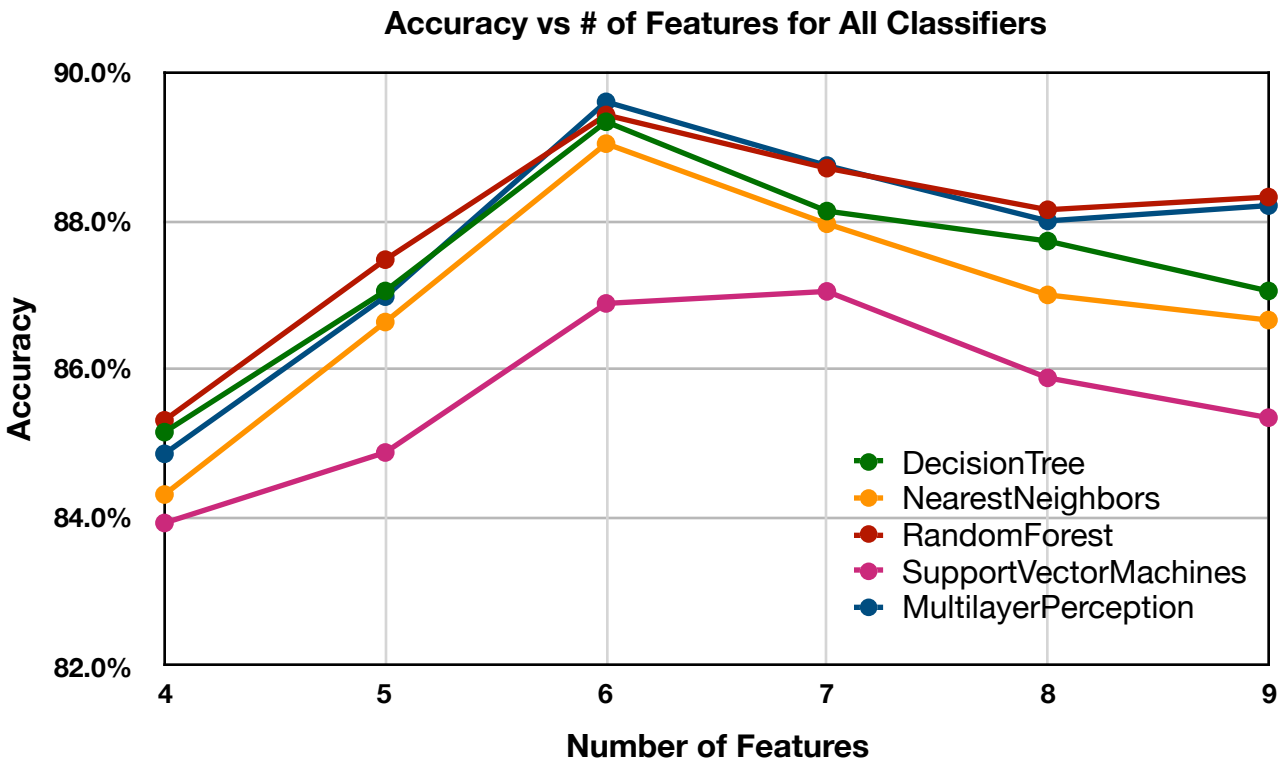

Figure 4: Result of Feature Extraction for All Tested Classifiers

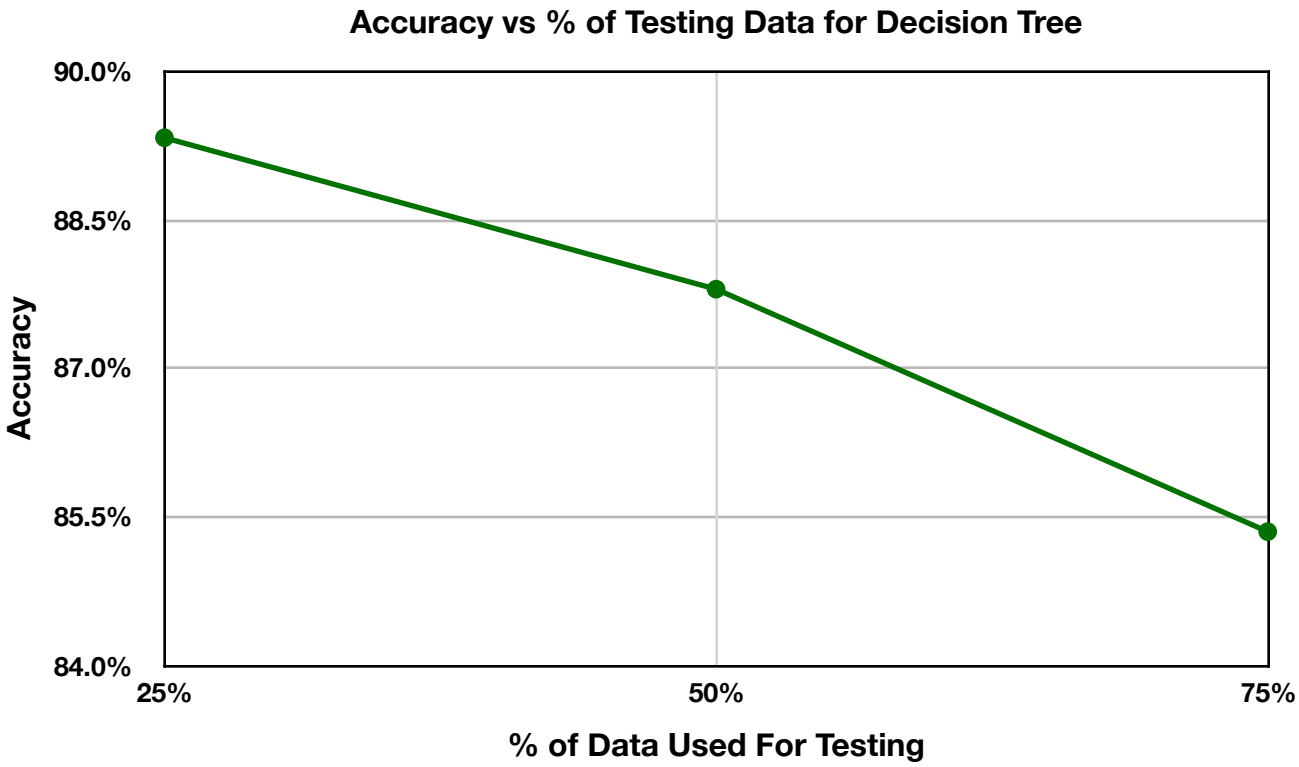

Figure 5: Result of Training/Testing Split for Decision Tree

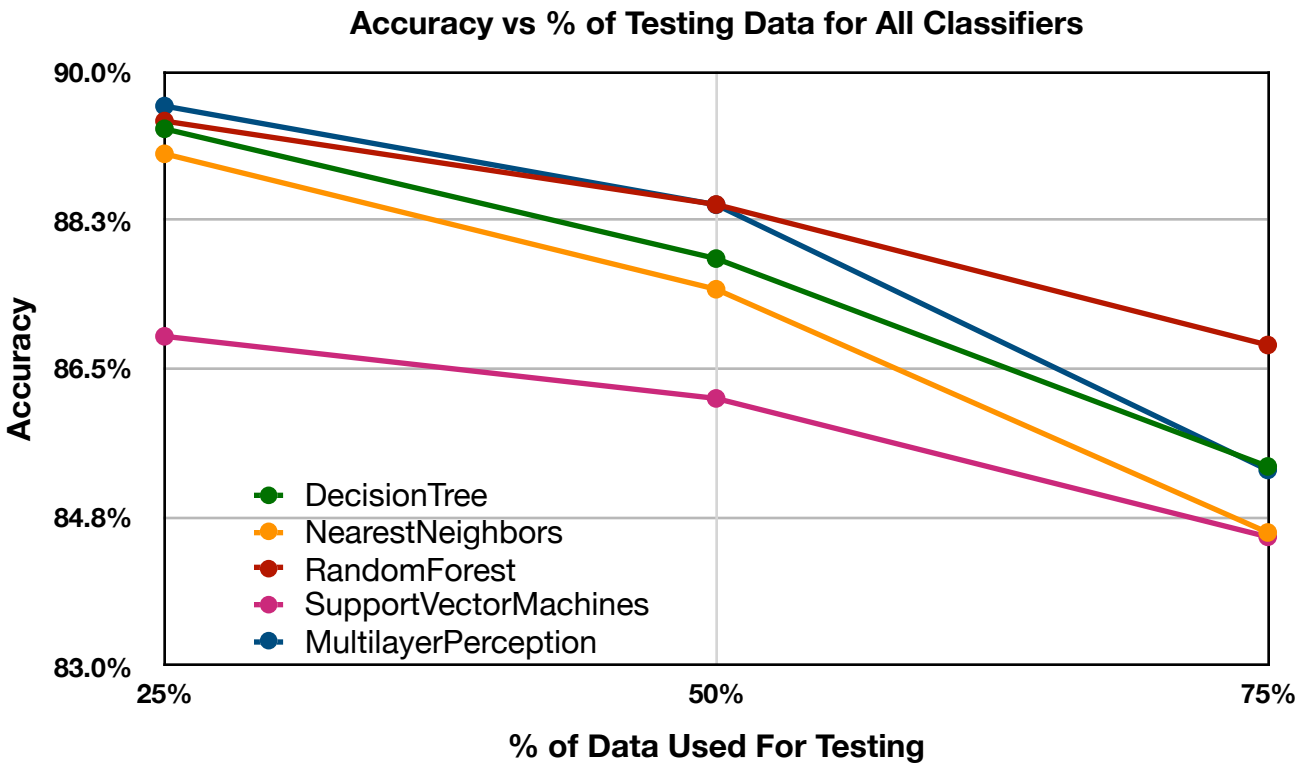

Figure 6: Result of Training/Testing Split for All Tested Classifiers

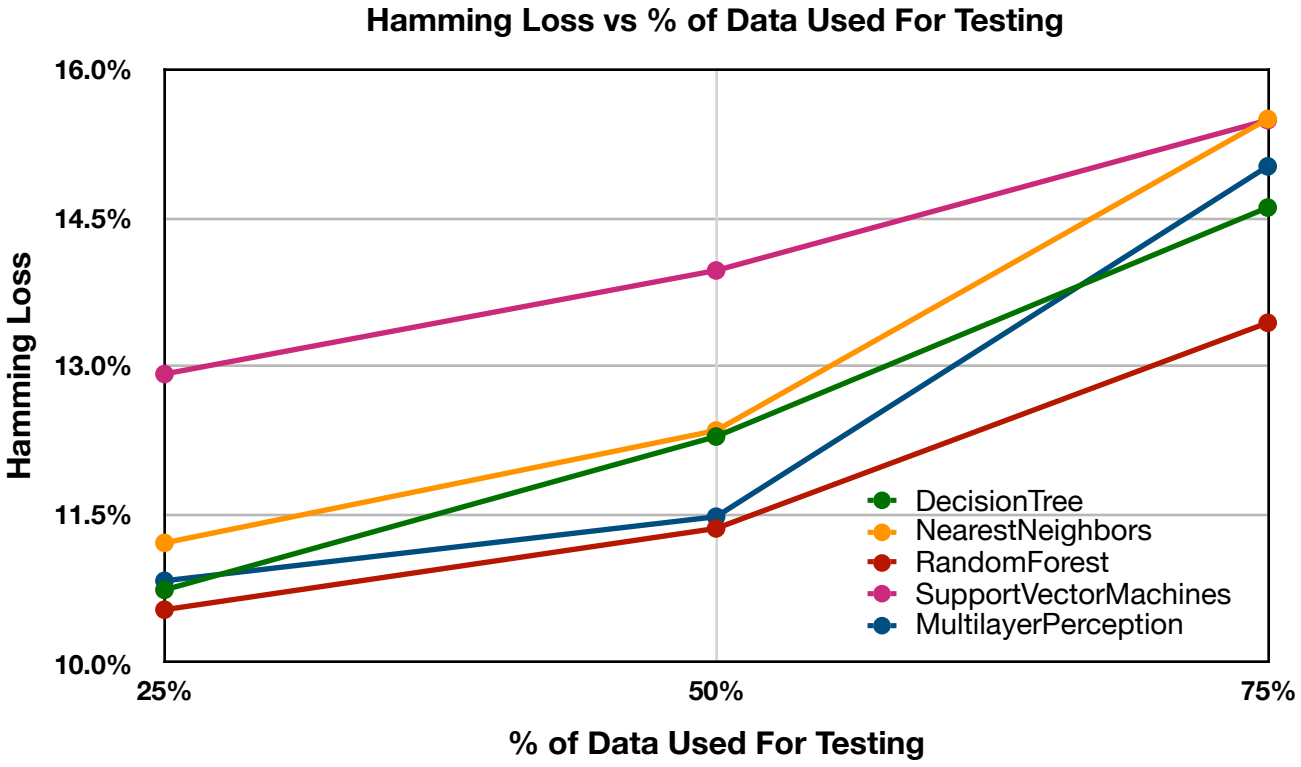

Figure 7: Hamming Loss vs Training/Testing Split for All Tested Classifiers

Discussion & Conclusions

Understanding whether a webpage is attempting to phish is a textbook classification problem. Therefore we can use the many tools available to us, such as python, and MatLab, to create classifiers to try and correctly warn a user of an apparent phishing attack. The chosen approach for classification was that of a supervised learning, multinomial classification, Decision Tree. Feature selection suggested that only 6 of the top 9 features be used, and experiments concluded that a random 75% of the total dataset was sufficient to properly train the classifiers and the rest to train. With these to considerations a Decision Tree classifier was created and its resulting visual representation is shown in **Figure 1**.

This Decision Tree was compared against other classifiers with similar features and training set and its accuracy compared favorably. Multilayer Perceptrons and Random Forest Tree Ensembles had higher accuracies and lower Hamming Losses.

Compared to similar work done by Neda Abdelhamid et. al and Norman Sadeh et. al: all of the classifiers tested had a lower accuracy score. Sadeh used a PILFER classification approach which correctly identified over 96% of phishing emails [2] while Abdelhamid used a Multi-label Classifier based Associative Classification approach with over 94% accuracy [1].

The success of the project is evaluated based on the average accuracy of the Decision Tree and the ability to find a better solution by testing different classification methods. Though the Decision Tree approach was not the best, it held its own against better algorithms such as Multilayer Perceptions. Both Sadeh and Abdelhamid had a higher number of features at their disposal and a greater number of data to train against. Because the process by which the feature values were extracted, through a proprietary PHP script, was unavailable to me, I was unable to gather more data to experiment and was limited to the 1353 websites downloaded from the repository. Nevertheless I conclude this project to be a modest success.

Related Work

Norman Sadeh et. al used a PILFER method of classification to classify emails as coming from a phishing source or not. Similar features were used along with whether an email had javascript, html, and age. [2]

Neda Abdelhamid et. al conducted research based on a Multi-label Classifier based Associative Classification approach. This approach is rule based similar to how a Decision Tree works. Multi-label classification is the process of assigning multiple classes to an input instead of just one in normal multi-class classification methods. [1]

After the May 2017 phishing attack on Gmail users, on May 31st 2017 Google implemented their own proprietary phishing detection algorithm that looks at behaviors most common to phishing emails and delaying delivery to the user. It is unknown how affective this implementation is or whether they used machine learning.

Further Work

Further study into what characteristics best predict a phishing webpage could greatly benefit the features selected for successful classification. Creating a script that parses webpage behavior and components could help gather more data to use for training and testing. Pruning the Decision Tree is a good way to minimize overfitting, but was unavailable given the toolkit in Scikit-learn. Overall creating a tool to correctly detect phishing attempts is an important social good benefiting many.

References

[1] Abdelhamid, N., Ayesh, A., Thabtah, F. (2014). Phishing detection based Associative Classification data mining. *Expert Systems with Applications*, 41(13), 5948–5959.

[2] Fette, I., Sadeh, N., & Tomasic A. (2007). Learning to Detect Phishing Emails. *WWW '07 Proceedings of the 16th international conference on World Wide Web*, 16(1), 649-656.

[3] Fu, A. Y., Deng, X., Wenyin, L., Little G. (2006). The Methodology and an Application to Fight against Unicode Attacks. *SOUPS '06 Proceedings of the 2nd Symposium on Usable Privacy and Security*, 2(1), 91-101.

[4] Louppe, G., Wehenkel, L., Sutera A., & Geurts P. (2013). Understanding variable importances in forests of randomized trees. *NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems*, *26*(1), 431-439.

[5] Mackowiak, S. (2016) "New Report on the State of Phishing Attacks from Wombat Security Shows Significant Increases Year over Year." Wombat Securities, 27 Jan. 2016. Web.

[6] Stratton, R. J. (2009). Internet Security Threat Landscape. *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research Cyber Security and Information Intelligence Challenges and Strategies - CSIIRW '09,* 5(1), Article Num. 8.

[7] https://archive.ics.uci.edu/ml/datasets/Website+Phishing