Sample Paper: <http://cs229.stanford.edu/proj2012/ChongLiu-MaliciousURLDetection.pdf>

Assumptions: -1: Legitimate, 0: Suspicious, 1: Phishing

**Abstract**

Phishing has been a widespread issue for many years, claiming countless victims, some of which have not even realizes that they fell prey. The sole purpose of phishing is to obtain sensitive information from its victims. There have yet to be a consensus on the best way to detect phishing. The main aim of this paper is to present an analysis of using Random Forest to detect web-based phishing. We also determine some URL features that are more important than others in determining whether a site is phishing.

**Keywords:** URL, cyberattack, cybercriminal,

**Introduction**

Phishing is a cyberattack rooted in scare tactics, with the sole purpose of eliciting personally identifiable information (PII) from its victims. An attacker disseminates a fraudulent version of a legitimate website, usually via email, telephone or text messages [1], in the hopes that the victim would believe the claims made in the email. A successful phishing attack can result in an attacker obtaining credit card details and login information.

With the ever-increasing number of Internet users, comes even more data that needs to be protected. This data includes login information and credit card information, both of which are priceless to cyber criminals. These cybercriminals would try anything to gain this information. One such way that has been overutilized is via phishing websites.

Phishing can occur in three forms: web-based phishing where a website is duplicated to resemble a trusted website and tricks users into submitting sensitive information [2]. Email-based phishing, where an attacker sends email to countless users claiming some account issue, in hope some of them fall for it. Email phishing usually involves web-based phishing as well [2]. Malware-based phishing where malicious code in injected into a legitimate website and when the user visits that site, the malicious software is installed on the user’s system [2].

**Our Research Goal**

Our project focuses on web-based phishing detection and aims to identify the most relevant subset of features that can accurately identify phishing URLs, using Random Forests algorithm.

**Related Work**

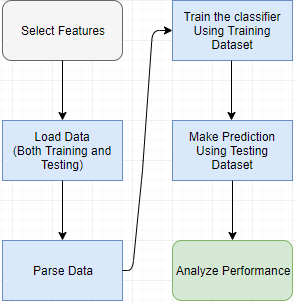
Why we choose machine learning to detect phishing: easy to implement, have high accuracy and recall rates, over traditional methods, black and whitelists.

**Methodology/System Design**

The aim of phishing websites is to trick users into submitting their private information like login credentials or credit card information. Some of these websites are easily recognizable as illegitimate but others require deeper analysis. One method of conducting a deeper analysis of websites is to look at their URLs.

In order to detect phishing websites with high precision and recall, we carefully select specific features that represent each URL, for training and testing inputs. We then divide this input into four different arrays: training input, training output, testing input and testing output. We then use the training input array to train our Random Forest classifier. Once the classifier is trained to accept the specific features and make the appropriate decision, we used the testing input to test the classifier. This process is summarized in Figure 1. These predictions are then evaluated based on accuracy, precision, recall and F-score.

We’ve recorded these four scores for five different subsets of features and compared them. Our goal was to find the subset of features that give the highest precision and recall rates.



**SubSect: Data**

We collected our training and test data from the UCI phishing dataset [6] that is publicly available. This dataset contained 2456 unique URL instances and a total of 11,055 URLs of which 6,157 are phishing and 4,898 are legitimate sites. Each row represented a URL and each URL was previously parsed and represented according to 30 features which could determine whether or not the URL is used for phishing or just identify a specific feature as suspicious for a particular URL. The features considered includes whether or not an IP Address is used instead of a URL, the length of the URL, the presence of link tags to the same domain as the webpage and whether or not the webpage uses IFrames. At the end of each row, there is a result which identifies the true nature of the URL, 1 if it is a phishing site and -1 if it a legitimate site.

Fix this paragraph:

Our dataset is made up of discrete data. This means each feature can only be represented by -1, 1 and sometimes 0. Random Forest also proves to be highly efficient when working with large datasets. We used 10,000 URLs for training and 1,055 for testing. The test data contained 461 phishing sites and 594 legitimate sites. Our belief is that using more features to train the algorithm should result in better predictions, and hence higher accuracy, recall and F1 scores, for the test data

**SubSect: Feature Selection**

In any classification scheme, there are features which seem more prominent than others in achieving a correct classification. These features, combined with other salient features or even less salient features, can perform outstandingly. The difficulty arises when we must determine what are the most relevant features from a set and what combination of features give us near perfect classification accuracies. From the 30 features, we identified five subsets. These were grouped as shown below.

~~Set A: Features we think are most important.~~

~~Having\_IP\_Address, URL\_Length, having\_At\_Symbol, Domain\_registration\_length, links\_in\_tags, submitting\_to\_email, Iframe, age\_of\_domain, having\_Sub\_Domain, Redirect.~~

~~0, 1,3,8,14,16,18,22,23 (A-68.90%)~~

Set A: Web-presence related features. We chose these four features to determine if it is practical to determine the nature of a URL simply by looking at its presence on the internet, and not by any structural features of the URL itself. These features included: Domain age, Website traffic, Page Rank, Google Index.

23,25,26,27 (A-50.42%) 4 features

Set B: Features with only two (2) possible outcomes {-1,1}. This makes the data more binary and eliminates the possibility of uncertain URLs. 21 features

having\_IP\_Address, Shortining\_Service, having\_At\_Symbol, double\_slash\_redirecting, Prefix\_Suffix, Domain\_registeration\_length, Favicon, port, HTTPS\_token, Request\_URL, Submitting\_to\_email, Abnormal\_URL, Redirect, on\_mouseover, RightClick, popUpWidnow, Iframe, age\_of\_domain, DNSRecord, Page\_Rank, Google\_Index, Statistical\_report.

0,2,3,4,5,8,9,10,11,12,16,17,19,20,21,22,23,24,26,27,29 (A-76.77%)

Set C: Features with three (3) possible outcomes {-1,0,1}. These features allows for uncertainty in their output and includes: 8 features

1,6,7,13,14,15,25,28 (A-94.12%) Figure out WHY this is so good!

URL\_Length, having\_Sub\_Domain, SSLfinal\_State, URL\_of\_Anchor, Links\_in\_tags , SFH, web\_traffic, Links\_pointing\_to\_page.

Set D: All 30 features (A- 95.73%)

Set E: The most important features (those rated 0.01 and up) according to classifier.feature\_importances\_ function. These include:

0,1,5,6,7,8,12,13,14,15,23, 24, 25, 26, 27, 28 (A-96.20%) 16 features

having\_IP\_Address, URL\_Length, Prefix\_Suffix, having\_Sub\_Domain, SSLfinal\_State, Domain\_registeration\_length, HTTPS\_token, Request\_URL ,URL\_of\_Anchor ,Links\_in\_tags ,SFH ,age\_of\_domain ,DNSRecord,web\_traffic,Page\_Rank ,Google\_Index,Links\_pointing\_to\_page

1. SFH { -1,1,0 }
2. Submitting\_to\_email { -1,1 }
3. Abnormal\_URL { -1,1 }
4. Redirect { 0,1 }
5. on\_mouseover { 1,-1 }
6. RightClick { 1,-1 }
7. popUpWidnow { 1,-1 }
8. Iframe { 1,-1 }
9. age\_of\_domain { -1,1 }
10. DNSRecord { -1,1 }
11. web\_traffic { -1,0,1 }
12. Page\_Rank { -1,1 }
13. Google\_Index { 1,-1 }
14. Links\_pointing\_to\_page { 1,0,-1 }
15. Statistical\_repo {-1,1}
16. having\_IP\_Address { -1,1 }
17. URL\_Length { 1,0,-1 }
18. Shortining\_Service { 1,-1 }
19. having\_At\_Symbol { 1,-1 }
20. double\_slash\_redirecting { -1,1 }
21. Prefix\_Suffix { -1,1 }
22. having\_Sub\_Domain { -1,0,1 }
23. SSLfinal\_State { -1,1,0 }
24. Domain\_registeration\_length { -1,1 }
25. Favicon { 1,-1 }
26. port { 1,-1 }
27. HTTPS\_token { -1,1 }
28. Request\_URL { 1,-1 }
29. URL\_of\_Anchor { -1,0,1 }
30. Links\_in\_tags { 1,-1,0 }

**SubSect: Experiment Design**

**Experiment Design**

Brief overview of Random Forest:

Random Forest is a supervised classification algorithm that makes use of several classification trees [7]. A classification is made by passing each input vector down each tree, randomly. Each tree gives a classification, or vote, and the forest chooses the classification with the most instances, or votes [8]. We decided to use this algorithm because it is unexcelled in accuracy among its counterparts [8], it runs efficiently on large datasets and it can handle missing values [7].

Libraries used

How we parse the data into the different matrices.

**SubSect: Evaluation Methods**

To fully evaluate the effectiveness of a classification model, you must include its precision and recall scores. In order to evaluate the performance of our classifications, we’ve calculated the Accuracy, Precision, Recall and F1 Scores for each set tested.

Precision measures the number of instances that have been correctly classified and is a measure of the classifier’s exactness. It is the number of positive predictions divided by the total number of positive instances predicted. For us, precision answers the question, “Of all the URLs labeled as phishing, how many are actually phishing?” The formula to calculate precision is given by Equation 1 below.

Precision: P = TP/TP + FP - all correct classified / all classified

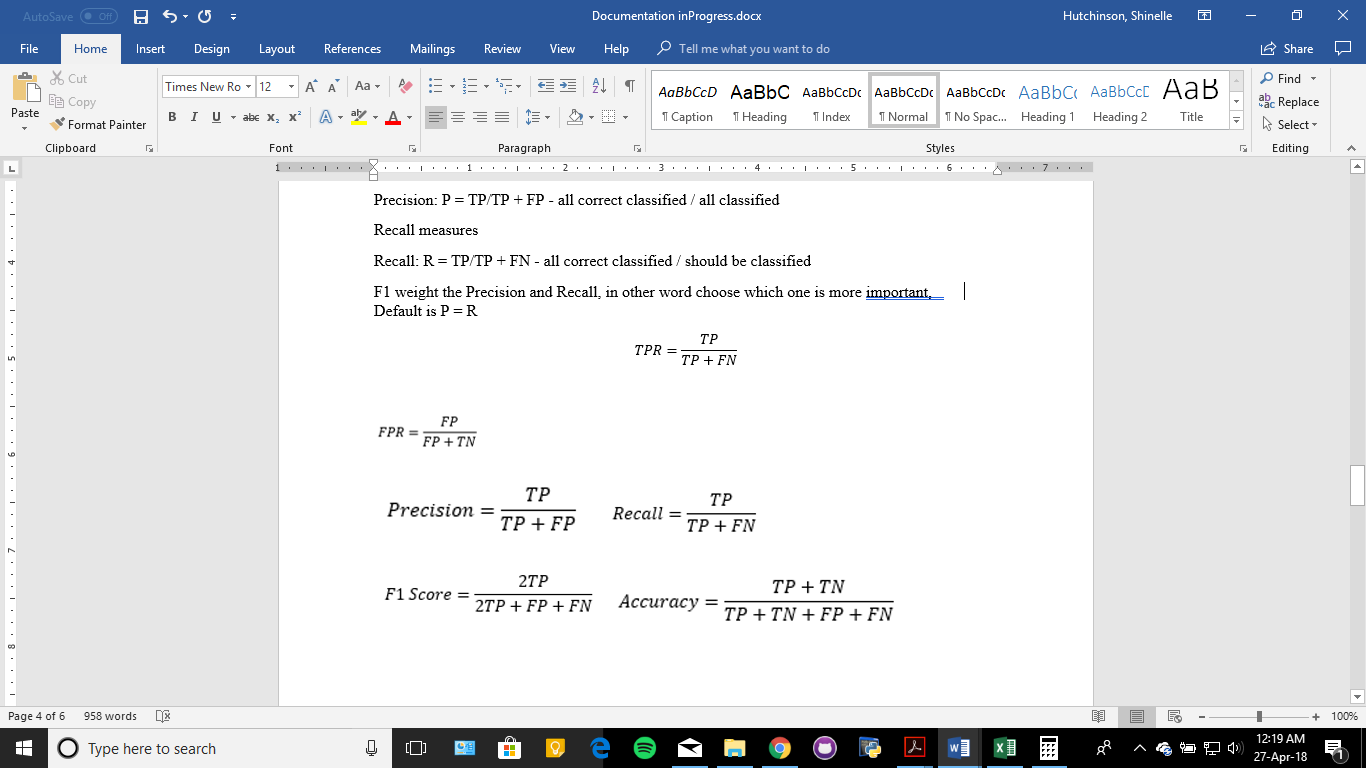
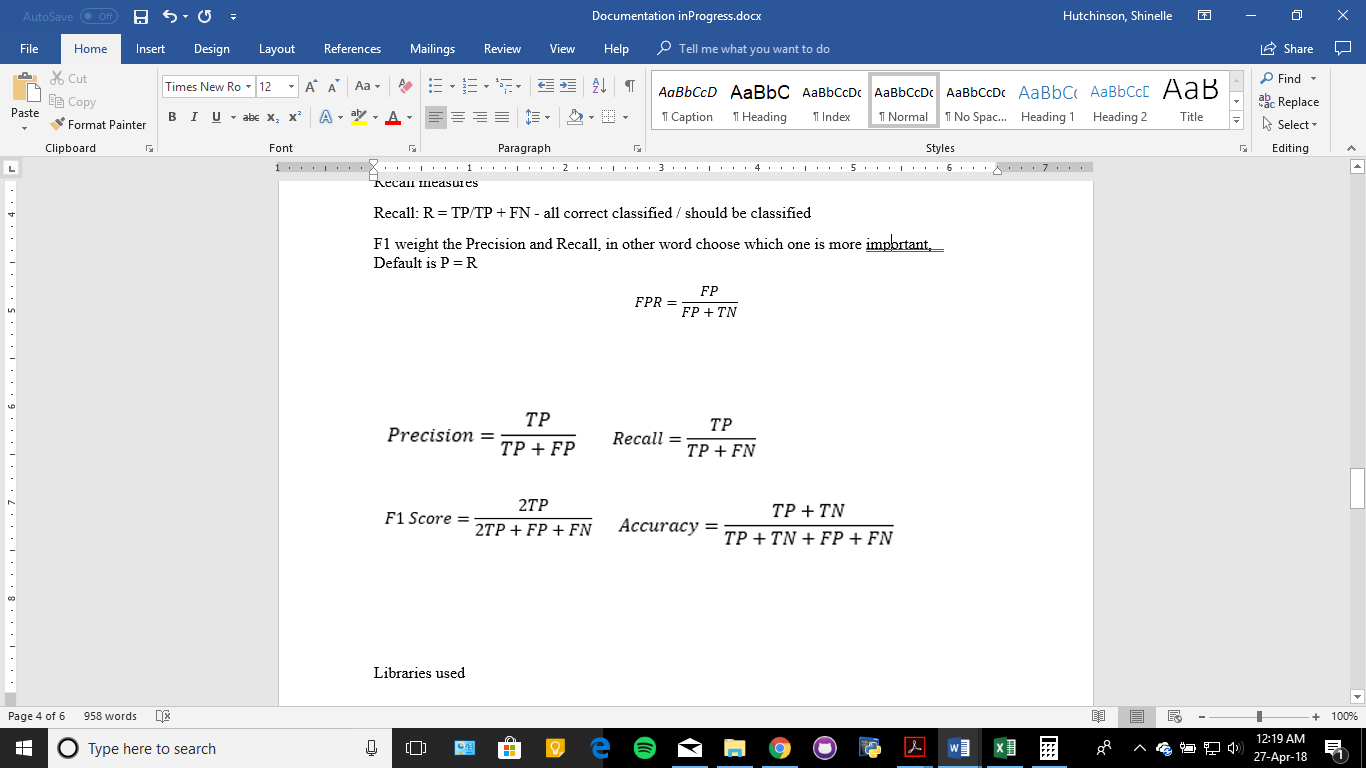
Recall measures the number of positive instances that the classifier correctly identified from the set of all positive instances. In other words, recall measure the number of instances that were missed [1]. Recall is a measure of the classifier’s completeness. For us, recall answers the question, “Of all the URLs that are truly phishing, how many did we identify as phishing?” The formula to calculate recall is given by Equation 2 below.

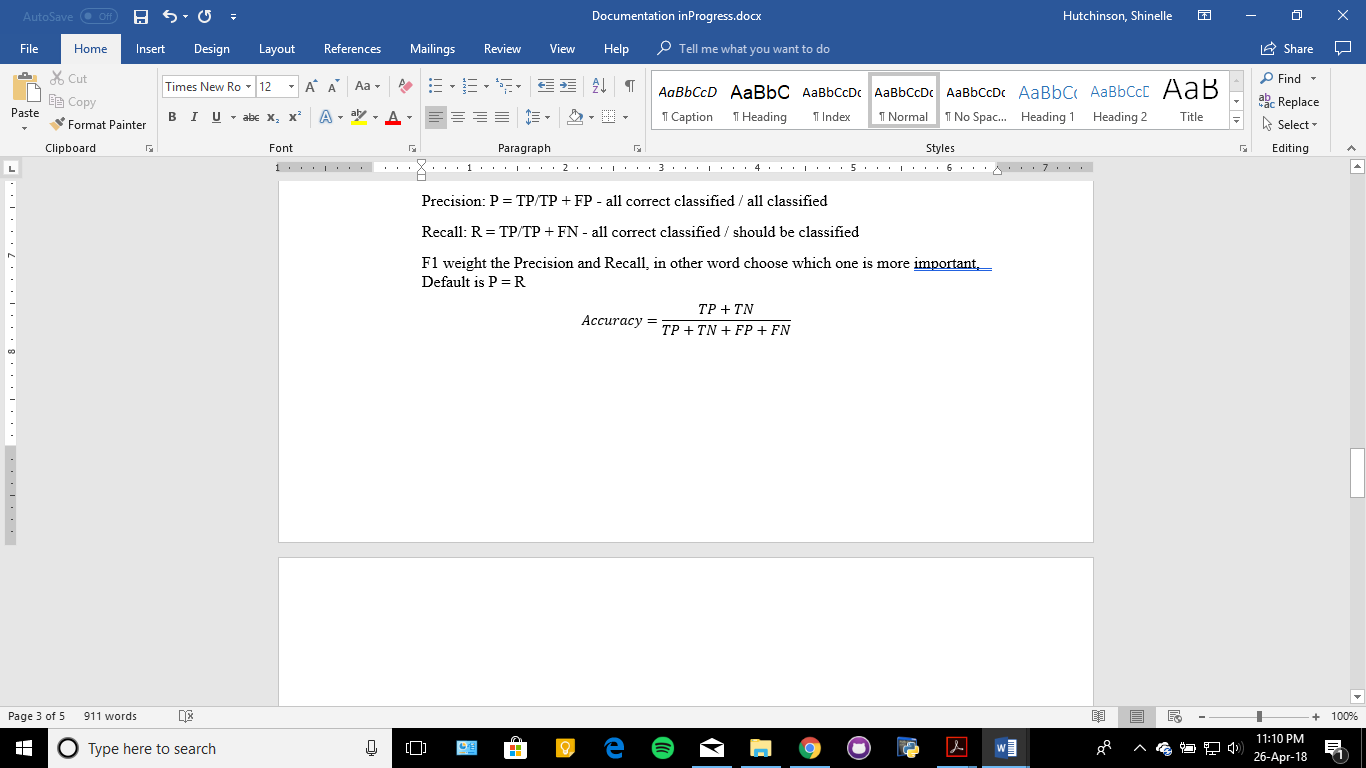
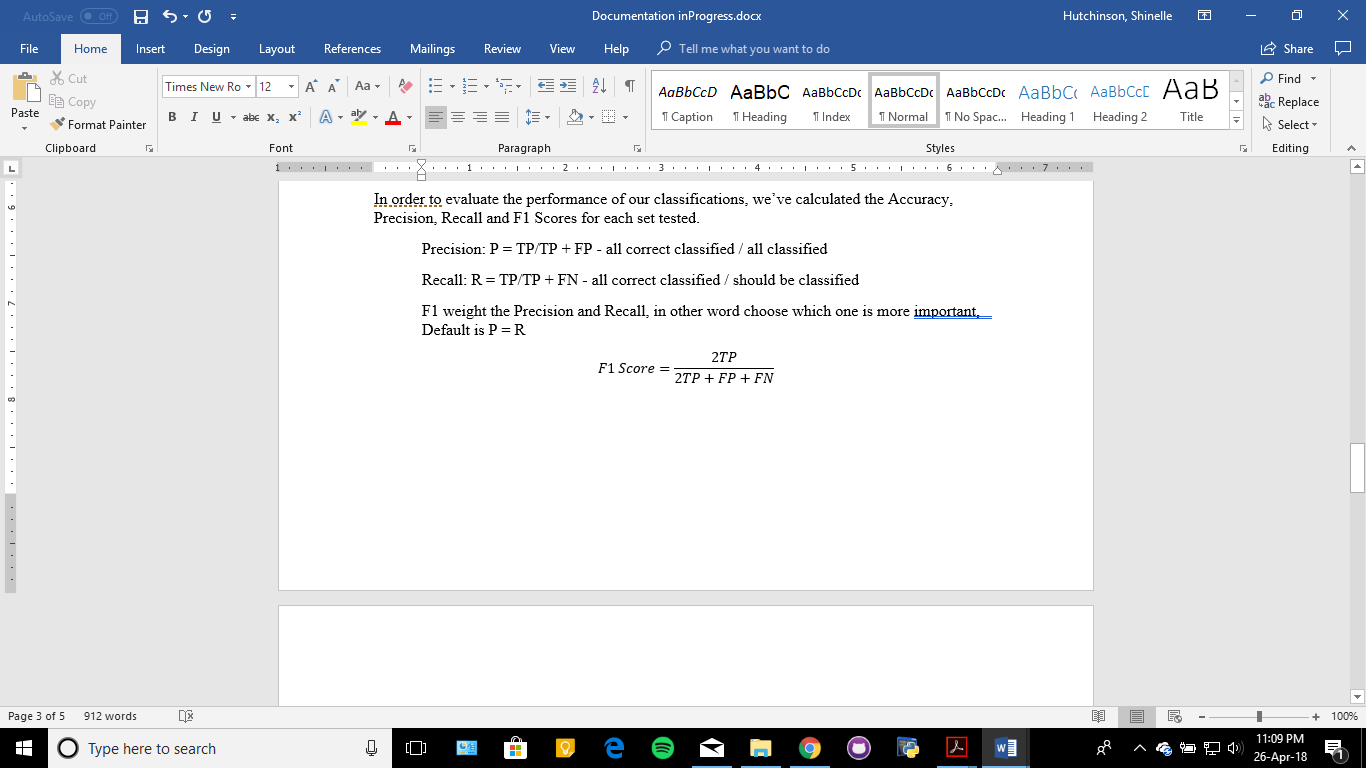
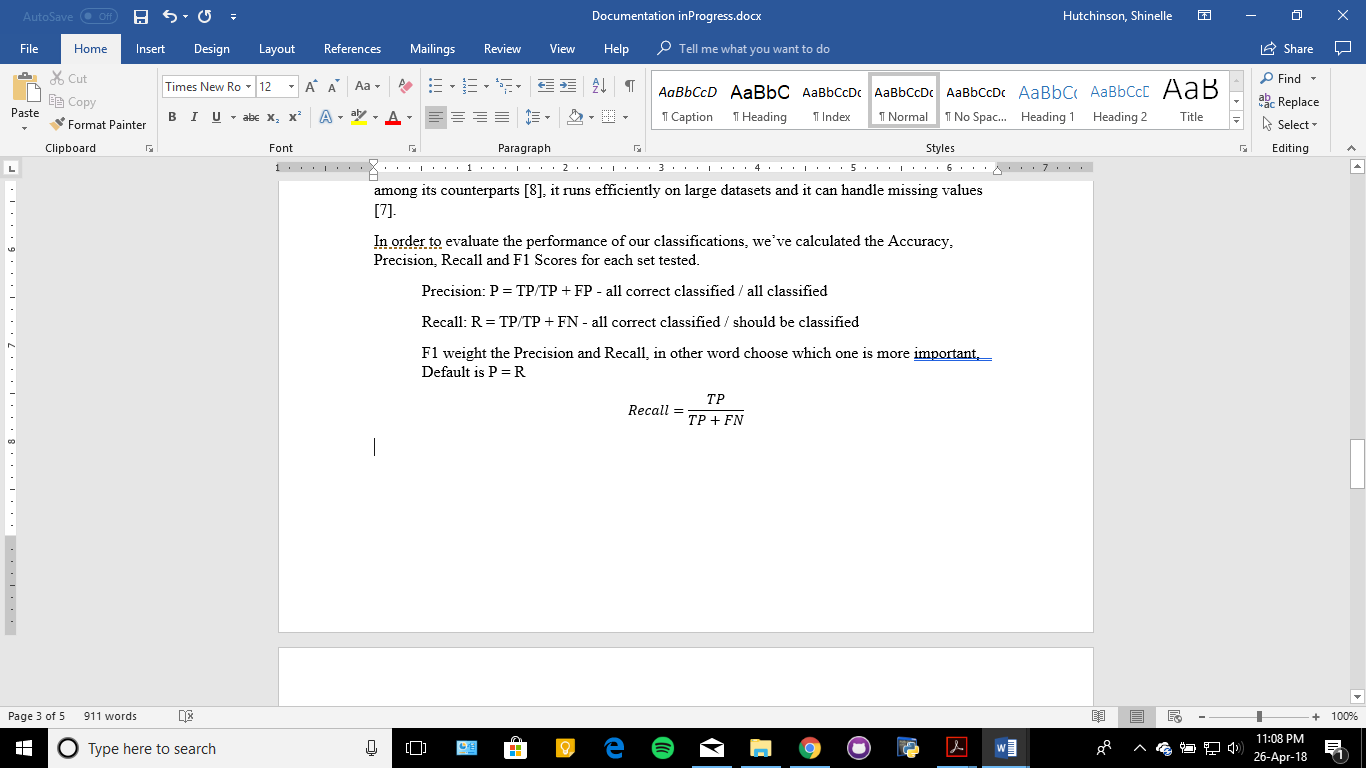
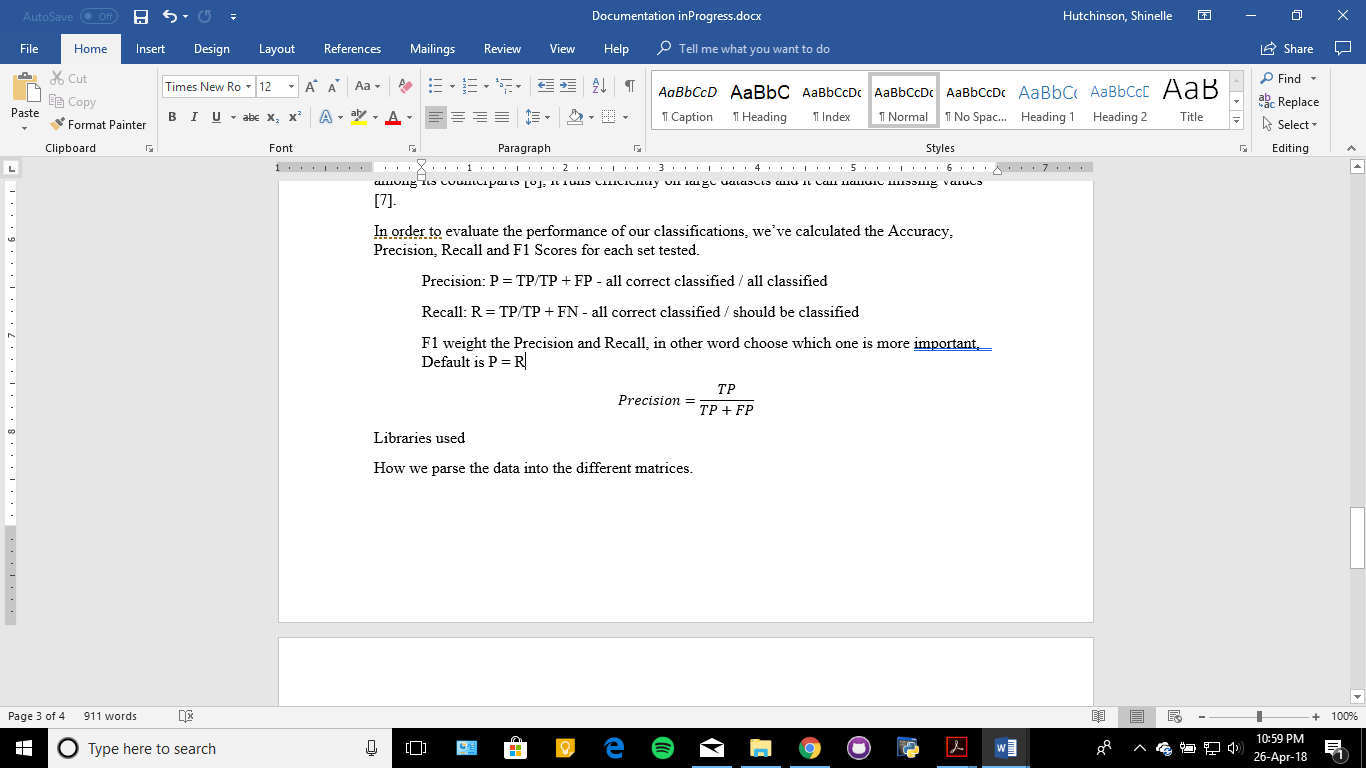
Recall: R = TP/TP + FN - all correct classified / should be classified

If a machine learning algorithm is good at recall, it doesn’t mean that algorithm is good at precision. That’s why we also need F1 score which is the (harmonic) mean of recall and precision to evaluate an algorithm.

Simply looking at a classifier’s precision and recall scores would be not constitute a substantial evaluation. As such, we include the F1 score, which is the weighted average of precision and recall scores. The formula to calculate F1 Score is given by Equation 3 below.

//F1 weight the Precision and Recall, in other word choose which one is more important, Default is P = R





Accuracy, simply put, is the number of correct classifications made out of all instances in the test data. The formula to calculate accuracy is given by Equation 4 below

The Receiver Operating Characteristic (ROC) curve is ideal for representing binary classifications like this one. The curve is plotted with the False Positive Rate (FPR) on the x-axis and the True Positive Rate (TPR) or recall, on the y-axis. The formulas to calculate FPR and TPR are given by Equations 5 and 6 below.

**SubSect: Results**

In this section we provide details on the performance of our classification model. As previously mentioned, thirty (30) initial features were broken down into five subsets and investigated. The overall performance of these subsets is reported in Table 1.

-Performance

Sets C, D and E were able to achieve great precision and recall rates, above 92.4%. Set E outperformed the other sets, achieving 97.09% precision and 94.14% recall. This set produced results that surpass some previous research [1] and closely follow others [3]. High precision and recall rates are directly related to high quality feature selection. Set A underperformed with 50% accuracy and 44.65% precision, while Set B produced average results, obtaining 76.77% accuracy and 71.51% precision. The increasing performance of each set can be easily seen in Fig. 2.

Each set contained varying amounts of features. The relation between feature count and performance is shown in Fig. 3. We can observe one profound characteristic: the number of features is not as important as the importance of the features in question. Set B contained 21 features to obtain 76.77% accuracy while Sets C and E used 8 and 16 features respectively and achieved over 94% accuracy.

The ROC Curve is useful in showing the relationship between False Positive Rate and True Positive Rate. The ideal case occurs when the curve has the shortest distance to the upper left corner of the graph. The Area Under the Curve (AUC) is also an important indicator of classifier performance. We see the Random Forest classifier achieves the best AUC for Set E, with 0.00000111.

Table I: Classification Performance

|  | ***Precision*** | ***Recall*** | ***F1 Score*** | ***Accuracy*** | ***FPR*** | ***TPR*** |
| --- | --- | --- | --- | --- | --- | --- |
| Set A | ***44.65*** | ***56.18*** | ***0.4975*** | ***50.42*** | ***0.0218*** | ***0.5618*** |
| Set B | ***71.51*** | ***77.87*** | ***0.7455*** | ***76.77*** | ***0.0286*** | ***0.7787*** |
| Set C | ***94.03*** | ***92.4*** | ***0.9321*** | ***94.12*** | ***0.0454*** | ***0.924*** |
| Set D | ***96.22*** | ***93.92*** | ***0.9506*** | ***95.73*** | ***0.2407*** | ***0.9392*** |
| Set E | ***97.09*** | ***94.14*** | ***0.9459*** | ***96.2*** | ***0.5404*** | ***0.9414*** |

Fig. 4 ROC Curve for Random Forest Classifier.

**Discussion:**

Set A was chosen based on the URL’s online presence rather than its structure. The performance of this set simply reiterates that we cannot determine, without a doubt, whether or not a website is phishing based solely on how long the site has been online, how many people visit the site and whether or not a user can visit the site from doing a Google search. These features are not substantial on their own.

Why Set B underperformed?

It was hoped that having binary input would make classification simpler for Random Forest. As evident by Set B’s performance, precision and recall scores were still good, though not desirable.

Why Set C performed so well?

In contrast, Set C obtained surprisingly better results although only containing features that could take on 3 possible outcomes. This means that some features had a value of zero, meaning that feature was suspicious, and not definitely phishing or legitimate.

Comment on feature importance and Set E outperforming the other sets.

Our feature importance ranking was done using Random Forest’s feature\_importances\_ attribute. This resulted in 16 features with a rating 0.01 or greater. These 16 features held the most weight and so was expected to perform extremely well. As evident by Set E’s performance, using only the most important features are enough to increase the accuracy, precision and recall.

Compare our results to previous works.

**SubSet Limitations**:

We were restricted in the number and type of features under consideration. Had we been able to alter these parameters, we could have included current phishing URLs as reported by PhishTank. Another limitation of our method is that some of our subsets were built based on our judgement of feature importance, this includes Sets A, B and C.

**Conclusion:**

We were able to improve the accuracy, precision and recall of the Random Forest classifier by using select features from all 30 features. This highlights the significance of first determining which features are most useful in determining whether or not a URL is phishing. Our findings give several possibilities for improvement. As a further step, we would include more features, find more important features, and combine all important features in hopes of achieving near perfect accuracy, precision and recall. We would also design our own URL parse so as to be able to include more current phishing URLs. In order to improve method of selecting important features for testing, we would employ forward feature selection and backward feature elimination methods.

Better method of selecting feature sets, eg. By using forward feature selection and backward feature elimination methods.