Malicious URL Detection Model Comparison

Nicholas Benyo, Henry Dihardjo, Samuel Sandell, Ian Suwandi

Abstract:

In this project, we are asked to experiment with a real-world dataset of our choice, and use machine learning algorithms to get the result. We are expected to experiment with the machine learning libraries we have learned and write a report of our observations. After implementing and observing performance of several different algorithms for our dataset, our results are presented below.

Keywords:

Machine Learning

Accuracy

Supervised Learning

I INTRODUCTION

We aim to evaluate both the efficacy and interpretability of multiple machine learning models. Decision trees and logistic regression provide fairly interpretable results but often lack the same performance of an ensemble model. The results will be reported and compared. Strengths and weaknesses will be evaluated with the intent to isolate an "ideal" model for the given dataset/data domain.

Expected result

True or not, it is often held that the more interpretable/inferential models achieve lower accuracy/predictive power. What we do not know is whether the difference in performance is substantial enough to elect a black box model (random forest) versus a more understandable model. We expect the more interpretable models to be less accurate, but perhaps not enough to sacrifice the user's ability to understand the output.

II BACKGROUND

Phishing is a malicious website that behaves like a legitimate one to get sensitive data like credit card number or even bank account password. It uses social engineering and technical deception to extract private information from the web user. Generally, phishing web pages have similar page layouts, blocks and fonts that impersonate legitimate web pages in an endeavor to influence users to obtain sensitive data. Phishing is a type of fraud that has been quickly growing and has been one of the dangerous threats within the web.

Examination of Data, Rule Generation and Detection of Phishing URLs Using online Logistic Regression presents a logistic regression-based approach for detecting phishing URLs. The paper uses online logistic regression, which updates the predictor upon more training data becoming available. This allows the model to adapt to changing patterns of what constitutes phishing URLs. The model uses both lexical and host-based features to build the feature set. A dataset containing examples of both safe and malicious URLs from the DMOZ Open Directory Project and Phishtank was built to train and test the model. The paper found that that the logistic regression model outperformed other machine learning algorithms, including random forest, J48 decision tree, and naive Bayes, for classifying the URLs in terms of accuracy. This conclusion held true for multiple splits of the dataset. This paper is relevant to our project because it involves classifying phishing data and uses a logistic regression model.

III DATASET

Dataset: https://archive.ics.uci.edu/ml/datasets/phishing+websites

The dataset used for the project is composed of instances of websites, some of which are phishing websites. The dataset contains features and labels for each instance to be used for classification. The dataset has information from 2456 different websites, collected from different sources. Instances of legitimate websites were taken from Google, while instances of phishing sites were collected from Phishtank and Millersmiles archive. Each instance in the

dataset has features relevant to whether or not the site is a phishing site. The features of the dataset include information relating to the URL anchor, the request URL, the server form handler, the URL length, the symbols in the URL, the prefixes and suffixes in the URL, IP data, the sub domain, web traffic of the site, and the domain age of the site. Each instance also has a class value, denoting whether or not it is a phishing site. Sites included in the dataset can be one of two classes: legitimate, and phishy.

IV METHOD

Multiple models were evaluated for predictive power. Under the black box section of this experiment, six separate methodologies were implemented. The first multi-layer perceptron model was tested, with single layer of neurons using lbfgs solver and logistic activation function. The second iteration stacked a second layer of neurons using the same solver and activation function. After MLP, SVM models using both linear and RBF kernel functions were tested. Following SVM, KNN-3 and KNN-5 were implemented. Next, Naïve Bayes Classification using the models from Gaussian and Bernoulli, and also Logistic Regression model were implemented and tested. Finally, a random forest model was built with a limit of 100 trees and no max depth.

The scikit-learn python library was used in the implementation for the machine learning algorithms as well as the performance measures. The following modules were used to implement the algorithms tested in the project:

- MLPClassifier from sklearn.neural_network: multilayer perceptron neural network classifier
- svm from sklearn: support vector machine classifier
- KNeighborsClassifier from sklearn.neighbors: K-nearest neighbors classifier
- RandomForestClassifier from sklearn.ensemble: random forest classifier
- GaussianNB and BernoulliNB from sklearn.naive_bayes: naive bayes classifier
- LogisticRegression from sklearn.linear_model: logistic regression classifier

Performance measures were implemented with functions from scikit-learn's metrics module. Scikit-learn's test_train_split() method was used to separate the dataset into sets for training and testing. Pandas and numpy were used for the manipulation and representation of the dataset.

V RESULTS

Among black box models, random forest was consistently the most performant across all metrics. The random forest model attained an accuracy greater than .96, an F1 score above .97, and an AUC of .9644. The confusion matrix shows a slightly higher false negative rate than false positive with Random Forest.

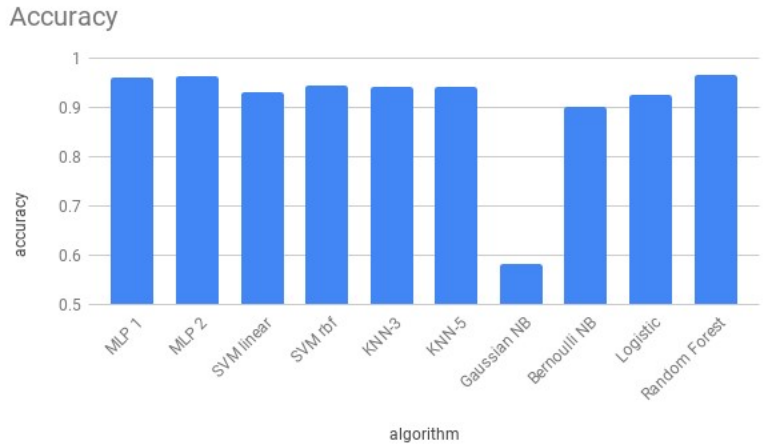
Black-box Model Performance Metrics

	MLP 1	MLP 2	SVM Lin.	SVM RBF	KNN-3	KNN-5	Gaussian NB	Bernoulli NB	Logistic Regression	RF
Accuracy	0.9593	0.9620	0.9299	0.9439	0.9426	0.9408	0.5829	0.9023	0.9244	0.9661
Balanced Accuracy	0.9163	0.9196	0.8526	0.8813	0.8814	0.8789	0.2650	0.8002	0.8442	0.9288
F1	0.9642	0.9669	0.9391	0.9513	0.9497	0.9479	0.4201	0.9141	0.9338	0.9703
Precision	0.9619	0.9578	0.9257	0.9380	0.9448	0.9467	0.9970	0.9119	0.9283	0.9638
Recall	0.9665	0.9761	0.9530	0.9649	0.9546	0.9490	0.2661	0.9163	0.9394	0.9769
AUC	0.9582	0.9598	0.9263	0.9406	0.9407	0.9395	0.6325	0.9001	0.9221	0.9644

Random Forest Confusion Matrix

	Pred. Positive	Pred. Negative
True Positive	910	46
True Negative	29	1226

Accuracy Graph



VI CONCLUSION

One initial assumption was the performance increase of a black-box model may not exceed the value of interpretability. While this always is contingent on the need and purpose of the model, random forest performance is the most performant across the board, though not by large margins (with an exception to Gaussian Naïve Bayes). There is only about a 2 percent difference in accuracy between K-nearest neighbors and random forest and about a 6 percent difference between Bernoulli Naïve Bayes and random forest. This smaller difference in performance warrants a closer look as to whether or not the extra accuracy is worth the poorer interpretability. For applications relating to security such as phishing detection where accuracy is key, it may be prudent to trade off some interpretability for more accuracy and go with the random forest model. However, for an application where accuracy is less of a priority, another more interpretable model could be chosen without too precipitous of a drop-in performance.

REFERENCE

[1] M. N. Feroz and S. Mengel, "Examination of Data, Rule Generation and Detection of Phishing URLs Using online Logistic Regression," *2014 IEEE International Conference on Big Data (Big Data)*, pp. 241-250, October 2014.