Final Project

Identifying Phishing URLs using Machine Learning

Due:

In this project, you will have an opportunity to apply machine learning techniques to identify phishing URLs. Automation plays an important role in improving efficiency in security operations. Identifying patterns in data and codifying them as rules is fundamental to writing software that automates repetitive tasks. A human analyst can identify patterns in data that is not complex enough but the task becomes challenging for high dimensional data. In these situations, machine learning is a suitable approach. Machine learning is a technique to learn complex rules from a set of data called *training set* and use those *rules* to predict future data.

In this project, you will be given a dataset that contains 4000 URLs of which 50 % of them are benign and the other 50% is malicious. The data also contains for each URL the time that it got reported. Your first task is to identify features from the dataset that can be used to tell apart a benign URL from a malicious one. You will then use those features to train three different classifiers:

1. Support Vector Machine (SVM)
2. Logistic Regression
3. Decision Tree

Input / Output Specifications

Your program should read from a CSV file that has following columns:

url, seen_time, id

seen_time – reported time of the URL

id – unique identifier for that entry

Your output should be the predicted value for each of the entries:

```
0
1
0
0
1
.
.
.
0
```

Where 0 indicates malicious and 1 indicates benign URL.

Programming Language

You must use Python for this assignment. No other languages allowed. This is for ease of grading. For machine learning aspects, you must use the scikit-learn library (http://scikit-learn.org/stable/).

Submission

1. Python source code following the input output specification.
2. You must also submit Python source code for the training data that we provided showing cross-validation and test accuracy scores.
3. A summary of your observations on output from each of the three classifiers. The summary also should include your comment on the accuracy of the output. How and if the accuracy can / cannot be improved. The summary should include a **visualization** of output from three classifiers. You must also include 5 misclassified examples from each of the two classes and explain empirically why they got **misclassified.**

Python files should be in a directory named Code. The Code folder should also include a README file with instructions to execute your code. Zip the Code and summary file in PDF format and name your folder as Final_Project. Submit Final_Project.zip to Canvas.