ARTIFICIAL INTELLIGENCE REPORT

ABHINAV ERNAM	2018A1TS0450P
NIVEDITHA K	2020A7PS0067P
TOSHIT JAIN	2020A7PS0146P

Under the guidance of:

Dr. Vishal Gupta,

Assistant Professor,

Department of Computer Science & Information Systems

BITS PILANI

Table of Contents

Brief Overview

As education moves into the online realm, students find themselves browsing the internet to clear doubts, for researching about projects or learning beyond the curriculum. This has accelerated the educational process by bringing information at your fingertips, it has also exposed students at a young age to negativity and the dark sides of the internet. We have attempted to address the problem of content supervision in two ways.

Malicious websites are a huge hazard to students browsing the internet for educational content. We aim to build a model which will classify malicious and benign websites. We have done this by building a neural network from scratch as well as using Machine Learning Techniques through the SkLearn library

Literature Review

A Machine Learning Approach for Detecting Malicious Websites using URL Feature

Machine Learning approach can be used to classify URLs as malicious or benign. Several classification models can be built using classification algorithms and feature selection techniques have been used to check only on important features. In this paper, a wide range of Machine Learning algorithms (for URL detection) such as SVM, KNN, Random Forest, Decision Tree and Naïve Bayes are used. It is found that Random Forest achieves the highest accuracy. This paper uses a feature selection named Recursive Feature Elimination. The Association Rule Mining was performed by using algorithms like Apriori, FP Growth and Decision Tree Rule Making. After iterations of training and testing, it is found that Random Forest produces an accuracy of 96%.

Machine Learning & Concept Drift based Approach for Malicious Website Detection

Using the URL provided by the user, the approach collects Lexical, Host-Based, and Content-Based features for the website. These features are fed into a supervised Machine Learning algorithm as input that classifies the URL as malicious or benign. Classification has been done using Random forests, Gradient Boosted Decision Trees and Deep Neural Network classifiers. Hackers and miscreants are aware of the standard features that are obtained from the URL. This provides them with the opportunity to modify the properties of malicious URLs to escape detection. This phenomenon, where the relationship between the input data and the target variable changes over time, is referred to as concept drift. The algorithm for concept drift detection focuses on finding the difference in data distribution between the old training data and the newly collected feature vectors data. For every benign URL in the new dataset, it finds the least distant malicious URL feature vector in the old dataset. We observe that Gradient Boosting Algorithm is the most effective in classifying malicious and benign URLs with an accuracy of 96.4% for a maximum depth (longest path from root to leaf for any estimator tree in the ensemble) of 4. We observe that the effectiveness of these algorithms in detecting malicious websites begins to decrease when a concept drift occurs.

The Main Idea

For the malicious websites data set we have built both a neural network model as well as machine learning models to classify the URL as benign or malignant. The first step is data preprocessing in order to remove the redundant data followed by oversampling to remove class imbalances. We have implemented machine learning models in the order of : Random Forest, Logistic Regression and Gradient Boosting. The accuracy of each model has been tested. We have also programmed a 4-layer neural network which has been written from scratch.

Data Preprocessing

Five steps were followed:

- Imbalance Checking : The data was observed to be skewed towards the benign class.
- Redundant classes : Only the top 5 classes were considered when a data entry had a large number of classes.
- Redundant data entries : Redundant data entries were removed using a heat map.
- Binary values for the attributes were attributed so that we could now create a numerical array
- SMOTE is used; technique through which we can over sample and under sample the data to remove class imbalances.

Model Building

a. Random Forest

Random forest is a supervised machine learning algorithm that is commonly used to solve classification and regression problems. It creates decision trees from various samples, using the majority vote for classification and the average for regression. One of the most essential characteristics of the Random Forest Algorithm is that it can handle data sets with both continuous and categorical variables, as in regression and classification. For classification difficulties, it produces superior results.

In this project, the number of decision trees (n_estimators) is set as 100. We've imported sklearn and used the RandomForestClassifier() procedure directly. The overall accuracy of the algorithm is around 97%.

b. Logistic Regression

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. Logistic regression predicts the output of a categorical dependent variable. It gives the probabilistic values between 0 and 1. As with the previous Random Forest model, we've imported sklearn and used Logistic Regression() procedure directly. The overall accuracy is around 79%.

c. Gradient Boosting

One of the most powerful algorithms in the field of machine learning is the gradient boosting technique. As we all know, machine learning algorithm faults can be divided into two categories: bias error and variance error. Gradient boosting is one of the boosting strategies that is used to reduce the model's bias error. In Gradient Boosting, each predictor corrects its predecessor's error. We've imported sklearn and used GradientBoostingClassifier() directly. The overall accuracy is around 96%.

d. Neural Network

A node layer contains an input layer, one or more hidden layers, and an output layer in artificial neural networks (ANNs). Each node, or artificial neuron, is connected to the others and has a weight and threshold linked with it. If a node's output exceeds a certain threshold value, the node is activated, and data is sent to the next tier of the network. Otherwise, no data is sent on to the network's next tier.

In our project, we use the neural network architecture: 4 layer neural network [30, 16, 8, 4, 1] resulting in an accuracy of 67%.

The neural network has the following components :

1. Forward Propagation : To calculate the activations of each layer and make a prediction based on the weights and biases. For all the hidden layers a ReLU activation function has been used. This ReLU activation function will output the input if it is positive, else will output zero

2. Cost function : We have used a cross entropy loss function to minimise the losses. The cross entropy function is as follows:

$$L = -\frac{1}{m}\sum_{i=1}^{m} y_i \cdot \log(\hat{y}_i)$$

3. Back Propagation : This is used to fine tune the weights and biases of the different layers by minimising the loss functions. We do this by finding the gradients of parameters and adjusting the values with respect to the gradients
4. Prediction : Finally we predict the accuracy of our model by running out testing and training data sets and comparing the predicted values from the model and the ground truth labels

Pseudocode For First Three Models

Model1:Random Forest
set n_estimators to 100
call ek.RandomForestClassifier() procedure
set clf1 to above output
Train clf1
call fit() procedure with X_train,y_train as input
Test clf1
call score() method with X_test,y_test as input

Model2:Logistic Regression
call LogisticRegression() procedure
set clf2 to above output
Tarin clf2
call fit() procedure with X_train,y_train as input
Test clf2
call score() method with X_test,y_test as input

Model3: Gradient Boosting
set n_estimators to 100
call ek.GradientBoostingClassifier() procedure
set clf3 to above output
Train clf3
call fit() procedure with X_train,y_train as input
Test clf3
call score() method with X_test,y_test as input

END

Procedure to Run Code

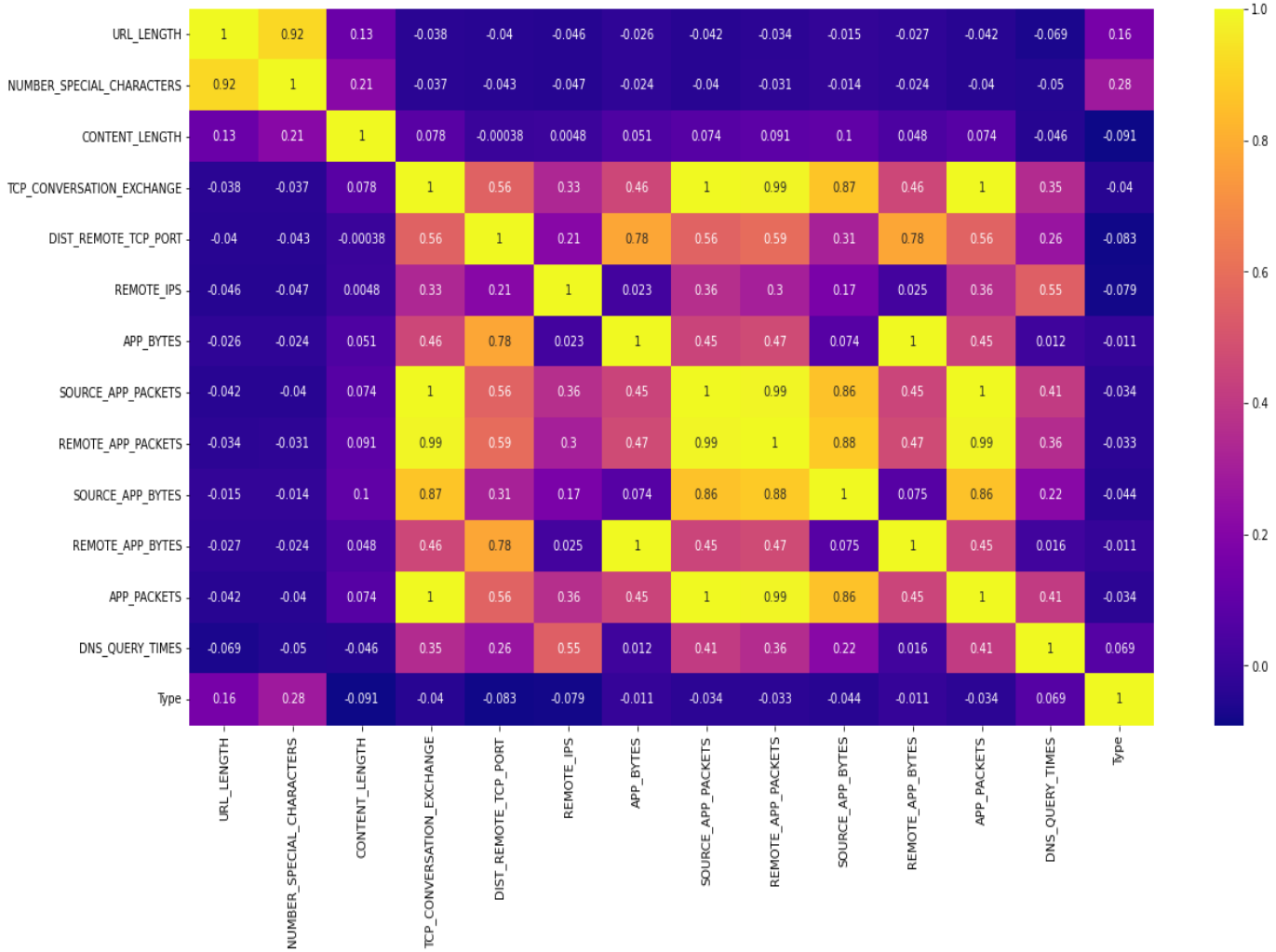
- The google drive folder with the dataset and code can be accessed here.
- Create a google colab notebook and upload the file
- Create a folder and upload the dataset used
- If the code doesn't directly run, change the directory of the dataset to the one present on drive
- Press Shift+Enter to run the individual code cells or Ctrl + F9 to run all cells

Results & Plot

After building the four models, the accuracy is calculated and tabulated as follows:

Model	**Score**
Random Forest	0.974
Logistic Regression	0.794
Gradient Boosting	0.966
Neural Network	0.67

Heatmap:



Limitations

As elaborated by Singhal et al. [2], there is a limitation to this system. Often malicious programmers change the URLs of malicious websites over time since they can be detected, which is known as concept drift. We find the closest malicious feature from the old data to the new benign data to conclude whether the concept drift has occurred. Unfortunately, this would require a consistently changing dataset which we don't have access to and hence cannot be implemented.

Applications

1) This can be used by educational devices to prevent children from accessing malicious content
2) Common man can use it to make sure he doesn't visit malicious websites which can bring viruses on his laptop and spoil the hardware
3) With extensions to the code, it can be used by cybersecurity firms to prevent hacks

References

[1] A. S. Manjeri, K. R., A. M.N.V. and P. C. Nair, "A Machine Learning Approach for Detecting Malicious Websites using URL Features," 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), 2019, pp. 555-561.

[2] S. Singhal, U. Chawla and R. Shorey, "Machine Learning & Concept Drift based Approach for Malicious Website Detection," 2020 International Conference on COMmunication Systems & NETworkS (COMSNETS), 2020, pp. 582-585.

****END****