Internship Report for May - June 2018

Yash Saboo

2015-2019 Batch

T.Y.C.S.

15070121170

**MORE DETAILS THE PROJECT**

**Problem Statement:**

To develop a Google Chrome Web Extension which can detect phishing/imitation websites and alert users about the same based on previously browsed websites.

**Technology Used and Technical Details:**

In the month of May, all my tasks involved implementation of machine learning techniques to predict if a website is Phishing or Legitimate.

Following concepts were used:

1. *Models:* Decisions Trees, Random Forests, XGBoost
2. *Learning:* Supervised
3. *Data:* Unbalanced
4. *Metrics:* Accuracy, Receiver Operating Characteristic(ROC) Curve, Confusion Matrix
5. *Data Mining Techniques:* Term Frequency – Inverse Document Frequency (TFIDF), One Hot Encoding

These technologies were learnt and implemented, as and when required, using various resources available on Internet.

**TASKS DONE**

**Task 1: Preprocessing Data**

Machine learning algorithms learn from data. It is critical that one feeds them the right data for the problem one wants to solve. Even if one has good data, one needs to make sure that it is in a useful scale, format and even that meaningful features are included.

* **Formatting:** The data collected from feature extraction script was saved in Comma Separated Format (CSV) file. The machine learning model that I was going to implement had a constraint that it only takes Pandas DataFrame or Numpy array as an input. Thus, the data was not in a format that was suitable for a model to work with and had to be transformed into that format.
* **Cleaning:** Cleaning data is the removal or fixing of missing data. There were occasions where the data was incomplete or did not carry any value. These instances had to be resolved, and were taken care by filling up the missing entries with either 0 or 1 or infinite value (such as -9999) as per the feature’s description and range values.
* **Binarize Data:** One can transform data using a binary threshold. All values above the threshold are marked 1 and all equal to or below are marked as 0. It was useful when I had to make mark certain similarities and dissimilarities.
* **Text Mining:** The machine learning models are generally not designed to accept string data values, so the features which had string as their data type had to be transformed into integer data type. When the string value is converted to integer value, one has to make sure that the information that string provided is replicated in integer values too and not lost. This was done using following two techniques:
  + **TFIDF:** In [information retrieval](https://en.wikipedia.org/wiki/Information_retrieval), tf–idf or TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a [document](https://en.wikipedia.org/wiki/Document) in a collection or [corpus](https://en.wikipedia.org/wiki/Text_corpus). It is often used as a [weighting factor](https://en.wikipedia.org/wiki/Weighting_factor) in searches of information retrieval, [text mining](https://en.wikipedia.org/wiki/Text_mining), and [user modeling](https://en.wikipedia.org/wiki/User_modeling). The tf-idf value increases [proportionally](https://en.wikipedia.org/wiki/Proportionality_(mathematics)) to the number of times a word appears in the document and is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general. In my case, each website was a document and set of websites was a corpus.
  + **One Hot Encoding:** One hot encoding is a process by which categorical variables are converted into a form that could be provided to ML algorithms to do a better job in prediction. The categorical value represents the numerical value of the entry in the dataset. As the number of unique entries increases, the categorical values also proportionally increase. The categorical values start from 0 goes all the way up to N-1 categories.

**Task 2: Build the Model**

* XGBoost model was used for binary classification of a website. To understand XGBoost, I had to understand working of Decision Trees and Random Forests.
* The overall details of the model are bound by a confidentiality clause but I will give a brief information about the models:
  + **Decision Trees**

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, decision tree algorithm can be used for solving regression and classification problems too. The general motive of using Decision Tree is to create a training model which can use to predict class or value of target variables by learning decision rules inferred from prior data (training data).

* + **Random Forest**

Random forest algorithm is a supervised classification algorithm. As the name suggest, this algorithm creates the forest with a set of decision trees. In general, the more trees in the forest, the more robust the forest looks like. In the same way in the random forest classifier, the higher the number of trees in the forest gives the high accuracy results.

* + **XGBoost (eXtreme Gradient Boosting)**

Boosting is an ensemble technique where new models are added to correct the errors made by existing models. Models are added sequentially until no further improvements can be made.

Gradient boosting is an approach where new models are created that predict the residuals or errors of prior models and then added together to make the final prediction. It is called gradient boosting because it uses a gradient descent algorithm to minimize the loss when adding new models.

The XGBoost implements the gradient boosting decision tree algorithm. This algorithm goes by lots of different names such as gradient boosting, multiple additive regression trees, stochastic gradient boosting or gradient boosting machines. XGBoost is designed for speed and performance. This approach supports both regression and classification predictive modeling problems.

**Task 2: Training, Testing and Optimizing the Model**

XGBoost model was finally implemented. Following are the training and testing statistics of the model:

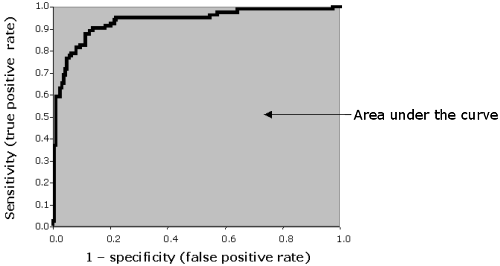
* Total Websites: 5,125
* Test Accuracy: 92.52%
* Split Ratio: 0.7
* Training Websites: 3587
* Testing Websites: 1538

As the data was unbalanced, accuracy was not the appropriate metric to measure the model’s performance. Thus, ROC curve was used as the metric for model performance measure.

* **ROC Curve**

In statistics, a receiver operating characteristic curve, i.e. ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability of detection in machine learning. The false-positive rate is also known as the fall-out or probability of false alarm and can be calculated as (1 − specificity).

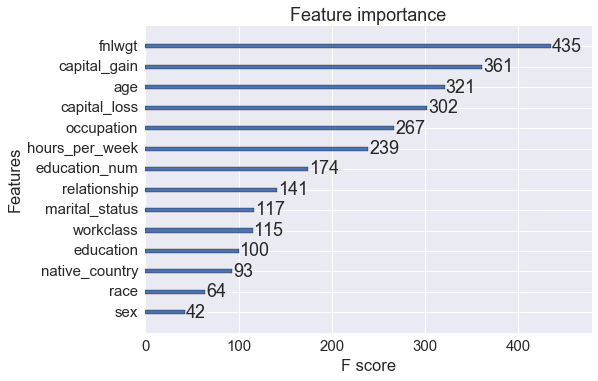
The area under the ROC curve (AUC) of a test can be used as a criterion to measure the test's performance ability. Larger the AUC, better is the performance of the model.



The above curve is ROC Curve of some model. (I can’t paste my project’s ROC because of confidentiality clause) The AUC for my ROC curve was 0.98 which is excellent.

* **Feature Importance Map**

XGBoost provides a functionality called “Feature Importance Map” which helped me identify which important features. It, indirectly, helped me find features which were not contributing in the classification process. These features will be accordingly dealt with while hyperparameter tuning.



The above “Feature Importance” chart is NOT of my model, but it looked like this chart (I can’t paste my project’s chart because of confidentiality clause). The score present for each feature represents its importance. Higher the value, higher the importance.

All these files are uploaded regularly on the Bitbucket of Symantec using Git.

**FUTURE WORK**

My current task is perform model’s hyperparameter tuning to increase AUC of ROC Curve. Once that is completed, next task includes finding the threshold of Hamming Distance which would help in classifying Phishing and Legitimate websites using the algorithms designed for Layer 1 and 2. They will be described in the next report (June - July).

Thank You.