CHAPTER 3

PERFORMANCE EVALUATION OF PREDICTIVE CLASSIFIERS AND ITS VARIANTS TOWARDS XSS DETECTION

3.1 INTRODUCTION

Nowadays web is a common platform for all categories of people involving e-payment, e-business, e-service and many more. All these applications require an interface with the users during the access of their session. The behaviour of the applications may vary from time to time and application to application. Detecting the nature of the application is a hectic job which is to be done carefully. Wasserman (2008) spots that the web anomaly is more common and researchers are in a need to concentrate on its data input operation that deviates to an abnormal operation. Handling the web information involves a lot of risk since the user is unaware of the anomaly. So it is necessary to classify the normal and abnormal information. It is found from the past study that the ML and DM algorithms are more suitable to predict based on the history. Moreover the results of those algorithms are independent of the domain in which they work over and depend only on the accuracy of the collected dataset.

The focus of this work is to identify whether a given web page is benign or malicious towards XSS in order to secure the web application. Malicious is an activity which is a slight deviation that collapses the objective of the application. XSS attack is one amongst the top 10 vulnerabilities (Source: www.owasp.org). Lee et al. (2000) have devised a framework for modelling and constructing intrusion detection system which was the first
customized detector for web systems. This framework was the first to teach the procedure of collecting the characteristics based on the data input, which paved way for the development of the Recommender Systems (RS) in the recent years.

Lots of information and knowledge are found around the globe in terms of web pages which grows year after year. Without an automatic extraction method, it is difficult to extract the hidden knowledge and information trapped between huge bytes of data. Classification Algorithms being a part of ML predicts based on the pre-known circumstances for analysing the future unknown circumstances. To show the significant differences between the classifiers, various statistical tests have been conducted using the dataset obtained from the XSSed (source: www.xssed.com) website.

3.2 CLASSIFIER AND ITS VARIANTS

Many techniques are found to extract the patterns from a huge database. Han & Kamber (2001) have identified that classification is one of the predictive techniques that predicts the group membership in a two-step process. The most successful factor is that the algorithms do not worry about the nature of the dataset. This feature paved a way for choosing the classification algorithms for this research work.

Likarsh et al. (2009) gave a proposal to detect malicious content present within Javascript tag using classification algorithms considering the obscure features. The classifier itself acts as a filter and provides a laudatory solution. Similarly Rieck et al. (2010) initiated a method named CUJO (Classification of Unknown Javascript Code) filtered patterns and devised a technique for capturing malignant code on the required computer.
From the analysis, it is found that classification algorithms are most suitable to detect malicious content by performing analysis with various factors. Rule based classifier, DT, Nearest Neighbour (NNe), Artificial Neural Network (ANN) are some of the prominent classifiers present. These algorithms try to classify the tuple during the arrival of a tuple.

The most common DM algorithms like Simple Bayes (SB), Naïve Bayes (NB), Iterative Dichotomiser3 (ID3) and C4.5 (represented as J48 in WEKA) that induces to provide a DT. SB and NB which are generalized as Bayes Classifiers do not construct a decision tree. DTs are simple and easy to interpret. Since DTs are constructed only with the limited attributes obtained from the web page, the case of over fitting does not exist.

3.3 FRAMEWORK FOR CLASSIFIERS TO DETECT XSS

Figure 3.1 shows the framework for construction and evaluation of the classifier and its variants. The framework consists of two major phases. Initially the pre-processed dataset is labelled into positive and negative tuples based on the class label. The first phase is to model the various classifiers based on the training set of data. The classification of data by the classifiers is conducted in the second phase. Then the performances of the classifiers are compared and the one with the highest accuracy is chosen for the future processes.
3.3.1 Attributes

Web page is the only source of input in this research work. By varied analysis, it was found that certain attributes of the web page pave a way for being malignant. Certain features like URL tag and its contents, script tag and its contents and few listed words are among the feature extraction criteria. The presence of these features is identified for a tuple with attributes
to store the associated binary value. Identification of perfect attributes is the major challenge and it is done through ideal and precise literature survey with an unspoilt content damage over the page.

### 3.3.2 Data Pre-Processing

Prior to the process of applying algorithms over the dataset, it is processed using WEKA 3.7 as per the requirements of the system. The transformed webpage as tuples are taken as input in due course. At the initial stage, all the tuples are grouped by class label.

### 3.3.3 Classifier Construction

The classifiers namely SB and NB works on the Bayes theorem which are known to be statistical classifiers that predicts class membership probabilities. Equation (3.1) determines the Bayes theorem. The highest posterior probability class is the prediction outcome. Algorithms like ID3 and J48 are Boolean functions. These algorithms work on measures like Entropy (Equation (3.2)) and Gain (Equation (3.3), Equation (3.4) and Equation (3.5)). Entropy is a measure of amount of information in an attribute. The possibility to improve the classification is when the entropy is higher. Information Gain (IG) is a measure of the difference in entropy from earlier to later after an attribute split which aims to reduce uncertainty. The highest Gain is used to split the subset associated with the attribute. Classifiers in WEKA are constructed by a due course to learn a model using the training data.

**Bayes Theorem**

\[
P \frac{H}{X} = P \frac{X}{H} \frac{P(H)}{P(X)} \quad (3.1)
\]

where \(P(H/X)\) represents the posterior probability.
P(X/H) is the likelihood.

P(H) is the class prior probability.

P(X) is the predictor prior probability.

X is the predictor class of the web page.

H is the hypothesis or attribute selected for analysis from the tuple.

**Entropy**

\[
\text{Entropy } A = - \sum_{x \in X} P(x) \log_2 P(x)
\]  

(3.2)

where A is not an attribute but an entire dataset.

x represents the classes namely positive and negative for the given set.

**Mutual Information Gain (Gain)**

\[
\text{Gain } A = I(p, n) - \text{Entropy } A
\]  

(3.3)

\[
I(p, n) = - \frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}
\]  

(3.4)

\[
\text{Entropy } A = - \sum_{i=1}^{m} \frac{pi+ni}{p+n} I(p_i, n_i)
\]  

(3.5)

where I (p, n) represents the expected information needed to satisfy a given training set.

‘p’ represents the total number of positive tuples in the training set.

‘n’ represents the total number of negative tuples in the training set.

‘\(p_i\)’ represents the number of positive tuples covered by the \(i^{th}\) value of the attribute ‘A’, that is to be included in the rule antecedent.

‘\(n_i\)’ represents the number of negative tuples covered by the \(i^{th}\) value of the attribute ‘A’ that is to be included in the rule antecedent.

‘m’ represents the number of values of an attribute ‘A’.
3.3.4 Classification of Test Tuples

As a web page enters to the browser, it is treated as a testing set. The tuple is made to undergo the classification process and it is identified based on the class label. The result of class label is examined for XSS detection process that happens as a parallel process of classification. Finally, the classifier accuracy is calculated using the Equation (3.6).

\[
\text{Accuracy} = \frac{\text{Number of correctly classified tuples}}{\text{Total number of test tuples}}
\]  

(3.6)

3.4 EXPERIMENTS AND RESULTS

3.4.1 Dataset

The experiments are conducted with datasets that are organised by the due course of survey using WEKA 3.7. The malignant web page was found from XSSed archive web site and benign data are retrieved through Google search engine. The engine was processed with variety of input keywords referring to various domains. List of keywords used in Google is presented in Table 3.1.
Table 3.1 Sample Keywords for Benign Data

<table>
<thead>
<tr>
<th>DOMAIN</th>
<th>KEYWORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search Engine</td>
<td>Google, Yahoo, Web Search Engine</td>
</tr>
<tr>
<td>Food</td>
<td>Indian Food, Hotel, Importance of Food, Diet, lose Weight, Gain Weight, Calories</td>
</tr>
<tr>
<td>Attack</td>
<td>Massive Attack, Heart Attack, Military</td>
</tr>
<tr>
<td>XSS</td>
<td>XSS Cheat Sheet, XSS prevention, sql Injection</td>
</tr>
</tbody>
</table>

About 24 attributes are extracted for formation of a tuple and about 500 sample web pages are collected for classification. The 24\textsuperscript{th} attribute is the class label used to predict the nature of the web page. A portion of the data set before and after pre-processing is shown in the Figure 3.2 (Source: www.incapsula.com) and Figure 3.3 respectively.

Cross site scripting (XSS) is a common attack vector that injects malicious code into a vulnerable web application. XSS differs from other web attack vectors (e.g., SQL injections), in that it does not directly target the application itself. Instead, the users of the web application are the ones at risk.

A successful cross site scripting attack can have devastating consequences for an online business's reputation and its relationship with its clients.

Depending on the severity of the attack, user accounts may be compromised, Trojan horse programs activated and page content modified, misleading users into willingly surrendering their private data. Finally, session cookies could be revealed, enabling a perpetrator to impersonate valid users and abuse their private accounts.

Cross site scripting attacks can be broken down into two types: stored and reflected.
Stored XSS, also known as persistent XSS, is the more damaging of the two. It occurs when a malicious script is injected directly into a vulnerable web application.

Reflected XSS involves the reflecting of a malicious script off of a web application, onto a user's browser. The script is embedded into a link, and is only activated once that link is clicked on.

What is Stored Cross Site Scripting

To successfully execute a stored XSS attack, a perpetrator has to locate a vulnerability in a web application and then inject malicious script into its server (e.g., via a comment field).

One of the most frequent targets are websites that allow users to share content, including blogs, social networks, video sharing platforms and message boards. Every time the infected page is viewed, the malicious script is transmitted to the victim's browser.

Figure 3.2 Training Dataset before Pre-Processing

3.4.2 Transformed Dataset

The pre-processed data set is then taken as input and the necessary attributes leading to maliciousness are extricated from the web page without affecting the web page. A single line data as shown highlighted in Figure 3.3 is obtained as output for a web page that is supplied as an input.
Figure 3.3 Training Dataset after Pre-Processing

3.4.3 Classifier Evaluation

The model was constructed using the algorithms with the full packed training set of data. The classifier is evaluated using two test options namely 10-fold cross validation (10 CV) and split 50% (SF) for estimation of error. In the 10 CV, the total dataset is subdivided into ten parts and the first
nine parts are considered as training set and the remaining 10\textsuperscript{th} part is considered as test set. The process is iterated for ten times altering the test and train data set. The SF equally partitions the dataset into training and test set. The classifier is constructed using the training set and the efficiency of the classifier is estimated using the test set.

3.4.4 Performance Measures

For DM and ML algorithms particularly to the statistical classification problem, a table format visualizes the performance of the algorithm which is an error matrix or a confusion matrix or a matching matrix. The matrix is required in order to show the significant differences in the performance of classifiers. The classifier returns either a 0 or 1 that is denoted as ‘N’ or ‘P’ which is used for the 2 X 2 table formulation with False Positive (FP), False Negatives (FN), True Positives (TP) and True Negatives (TN). The following statistics namely True Positive Rate (TPR) (Equation (3.7)), False Positive Rate (FPR) (Equation (3.8)), Precision (PR) (Equation (3.9)), DR (Equation (3.10)) and FS Measure (Equation (3.11)) are calculated respectively.

True Positive Rate (TPR)

\[
TPR = \frac{TP}{P} = \frac{TP}{TP+FN} \tag{3.7}
\]

False Positive Rate (FPR)

\[
FPR = \frac{FP}{N} = \frac{FP}{FP+TN} \tag{3.8}
\]

Precision or Positive Predicted Value (PR)

\[
PR = \frac{TP}{TP+FP} \tag{3.9}
\]
Detection Rate (DR)

\[
DR = \frac{TP}{TP + FN}
\]  

(3.10)

F-Score (FS)

\[
FS = 2 \times \frac{PR + DR}{PR + DR}
\]  

(3.11)

Since the class attribute of the dataset considered for the experimentation is nominal or categorical, Kappa Statistic (KS) (Equation 3.12) is the one of the best normalized measure suited for evaluation which calculates the inter-rater agreement for categorical data. Mean Absolute Error (MAE) (Equation 3.13) is a measure to identify the closeness between the predictions and the actual outcomes of the sample data whereas Root Mean Squared Error (RMSE) (Equation 3.14) performs the same for the whole population and aggregates the magnitude of the predictive errors. Relative Absolute Error (RAE) (Equation 3.15) and Root Relative Squared Error (RRSE) (Equation 3.16) are calculated using the MAE and RMSE divided by the corresponding errors obtained by prior probabilities of the classes observed for the data.

Kappa Statistic (KS)

\[
KS = 1 - \frac{1 - p_o}{1 - p_e}
\]  

(3.12)

Mean Absolute Error (MAE)

\[
MAE = \frac{1}{N} \sum_{i=1}^{N} |w_i - \theta \Theta_i|
\]  

(3.13)
Root Mean Squared Error (RMSE)

\[ RMSE = \frac{1}{N} \sum_{i=1}^{N} (w_i - \theta \Theta_i)^2 \]  

(3.14)

Relative Absolute Error (RAE)

\[ RAE = \frac{\sum_{i=1}^{N} |w_i - \theta \Theta_i|}{\sum_{i=1}^{N} |\theta - \Theta_i|} \]  

(3.15)

Root Relative Squared Error (RRSE)

\[ RRSE = \frac{\sum_{i=1}^{N} (w_i - \theta \Theta_i)^2}{\sum_{i=1}^{N} (\theta - \Theta_i)^2} \]  

(3.16)

The term \( w \) represents the expected output, \( \theta \) represents the actual output and \( \nu \) represents the mean value of \( \theta \). \( p_o \) is the relative observed agreement and \( p_e \) is the probability of expected agreement and \( N \) being the number of samples.

Once the model is built using classification algorithms, the test tuples are classified. Based on the number of correctly classified test tuple, classifier accuracy is calculated. The time required for training the different instances is computed. Table 3.2 shows the various results obtained including accuracy and the training time involved for 10 CV and SF test option.
Table 3.2 Comparison of Classifiers based on 10 CV and SF

<table>
<thead>
<tr>
<th>CRITERIA</th>
<th>TEST OPTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10-FOLD CROSS VALIDATION</td>
</tr>
<tr>
<td></td>
<td>SB</td>
</tr>
<tr>
<td>Time taken to build the model (Seconds)</td>
<td>0.0010</td>
</tr>
<tr>
<td>Kappa Statistic</td>
<td>0.4601</td>
</tr>
<tr>
<td>Mean Absolute Error</td>
<td>0.1990</td>
</tr>
<tr>
<td>Root Mean Squared Error</td>
<td>0.3437</td>
</tr>
<tr>
<td>Relative Absolute Error (%)</td>
<td>69.1255</td>
</tr>
<tr>
<td>Root Relative Squared Error (%)</td>
<td>90.7933</td>
</tr>
<tr>
<td>Prediction Accuracy (%)</td>
<td>85.6784</td>
</tr>
</tbody>
</table>

From the results observed, the time taken criteria cannot be considered for comparison because all the classifiers had utilized negligible time for the model construction. On analysing both the test options, J48 yields better accuracy than other classifiers. Accuracy is not the only measure to compare the performance of the binary classifiers and hence KS can be computed. Usually if the KS value is greater than 0.8, the classifier is considered as an efficient classifier and the KS value always ranges between
-1 and +1. With respect to KS, the classifier J48 has obtained a value 0.6 on two considered test options nearing to 0.80 which clearly depicts that J48 classifier is better amongst the other classifiers. With respect to MAE, ID3 scores a value of 0.1449 and 0.1633 under 10CV and SF respectively which is a minimum error value interpreted. In order to find the high infrequent errors in prediction, RMSE is computed. In general, larger the difference between RMSE and MAE implies high inconsistencies in error size. From the calculated values, RMSE do not show a larger difference between them leading to a conclusion that the classifiers used for experimentation do not report high impact infrequent errors.

3.5 SUMMARY

In this chapter, the performance of various classifiers in WEKA are analysed with respect to different combinations of error estimate measures and accuracy. It is observed that the use of J48 brings considerable significant classification than ID3. Bayesian classifiers both behave in a similar way and the results are found close proximity with each other. From the Table 3.2, under the two test conditions it was observed that J48 encourages the accuracy by 3% in 10 CV and 2 to 5% in SF validation amongst the classifiers. From the results, it is identified that J48 can be opted for XSS Detector phase rather than other algorithms.