CHAPTER 2
LITERATURE REVIEW

2.1 Ensemble Learning Systems

Ensemble learning systems, which can also be referred to as *Multiple Classifier Systems* is an approach to solve a problem by combining multiple base learners [28]. Base learners or individual learners are usually simple models that are then combined through a strategy, the basic architecture of an ensemble system is depicted in Figure 2.1. The group of base learners can be identical, which makes the group a *Homogeneous Ensemble*, or the group can consist of multiple different learning models, in this case, a *Heterogeneous Ensemble* [29].

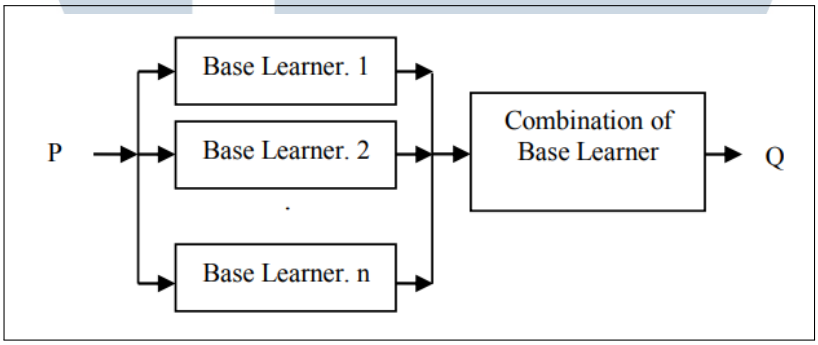

Figure 2.1. Basic Structure of an Ensemble

Ensemble learning systems are said to overcome 3 main problems that single learning systems have. These problems are the *Statistical Problem, Computational Problem,* and *Representation Problem*. The statistical problem is evident in high *Variances*, the computational problem through *Computational Variance* and the representation problem through *Bias* [30]. In modern studies, ensemble learning systems are mostly used due to their ability to reduce noise, or in other terms *Smoothing*. As depicted in Figure 2.2, through the combination of multiple models, a smoother decision boundary is established [31].

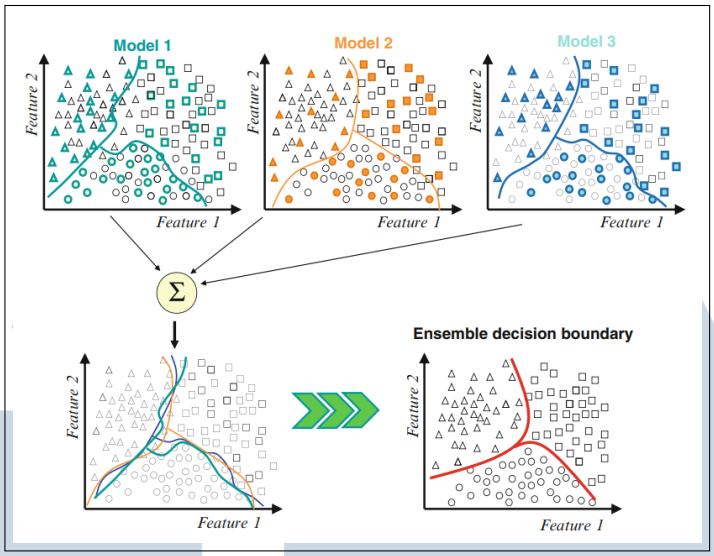

Figure 2.2. Reduction of Variability using Ensemble Systems

When combining multiple outputs and learners, some algorithms are applied. There are a large variety of algorithms, but the main standard classes of ensemble learning includes Bagging, Boosting and Stacking.

Bagging or *Bootstrap AGGregating* is the combination of a bootstrapping and aggregation process. This algorithm applies a method where all the base learners are identical and run in parallel. Each base learner will be trained with a different subset of the dataset and once an output for each learner is generated, the algorithm uses a majority vote to produce the final decision[32]. Figure 2.3 shows an example where the ensemble is made of decision trees.

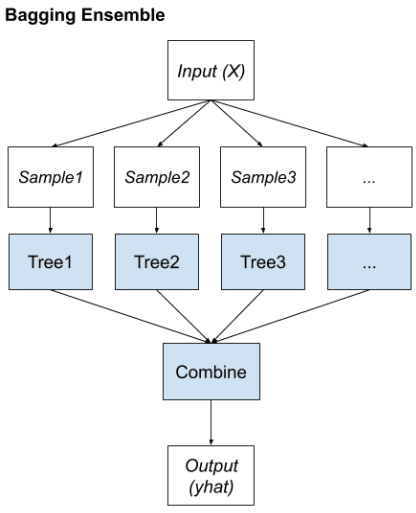

Figure 2.3. Example of a bagging ensemble with decision trees base learners

In contrast to bagging, boosting applies a step-by-step approach that re-weights the data samples on each step. The weights are adjusted after each phase of training based on the accuracy of the previous learners' results. The boosting algorithm unlike bagging, has the learners run sequentially rather than in parallel [33], this difference is also depicted in Figure 2.4.

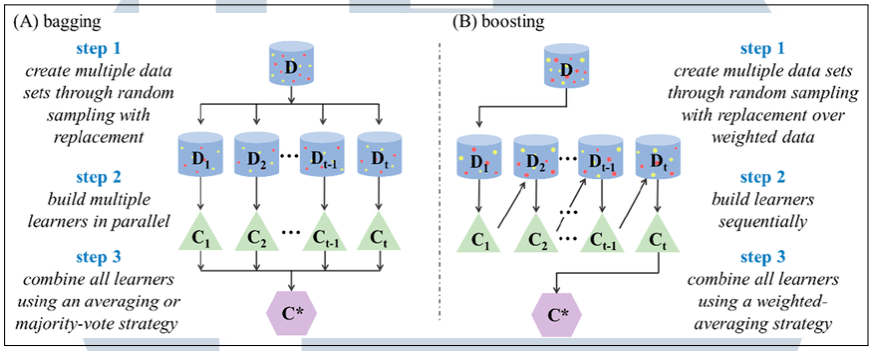

Figure 2.4. Main differences between Bagging and Boosting

The last main ensemble class is called Stacking. Stacking allows multiple types of models as the base learners to be run in parallel, from there the results from each learner is then combined altogether with another model. The first base learners that are in parallel are usually referred to as level 0 learners, while the combining model is referred to as the level 1 learner, or the meta-learner. To further explain how stacking is implemented, Figure 2.5 shows how the base structure of a stacking ensemble model.

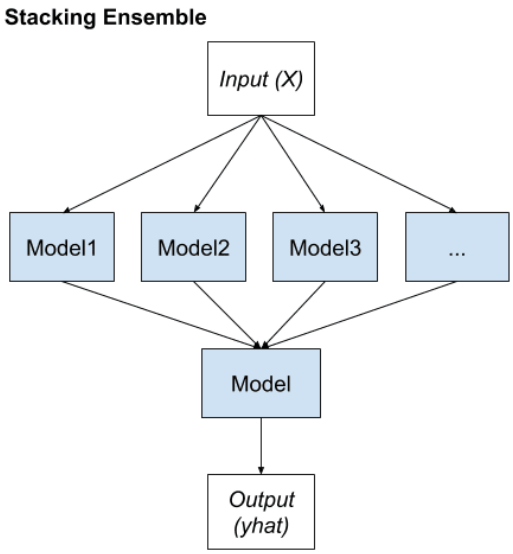

Figure 2.5. Basic structure of a stacking ensemble learning model

2.2 Decision Trees

Decision trees are simple approaches to prediction and classification problems, its defining characterizing is the recursive nature of the subsets involved within the trees. The top of the tree or the *root node* holds the *target* of the problem, and on each level, a decision needs to be made, this recursive pattern continues until the final decision is made, represented by a *leaf node*[34]. The basic structure of a decision tree is shown in Figure 2.6.

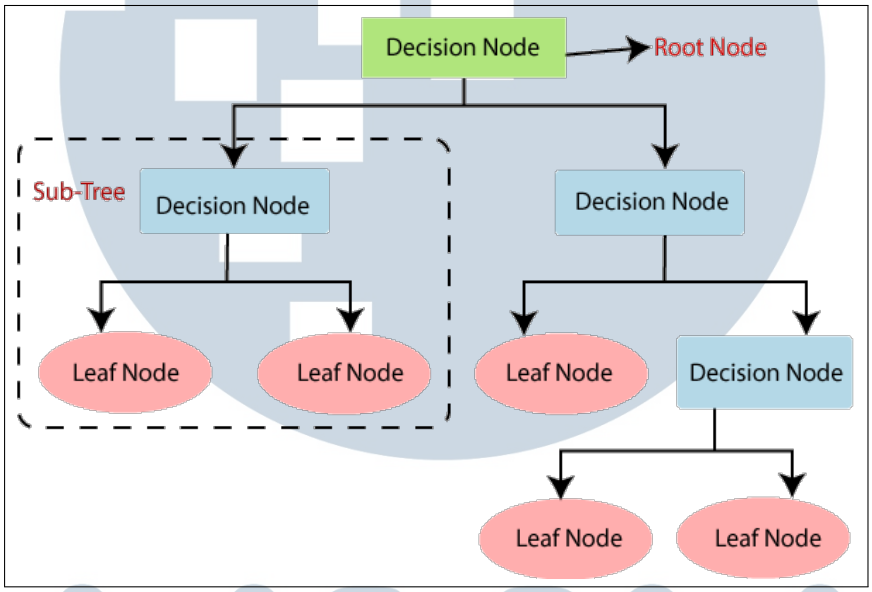

Figure 2.6. Basic Structure of a Decision Tree

2.3 Naive Bayes Classification

The Naive Bayes approach is to have the assumption that all features are independent. This method of classification can also be represented using the formula denoted below.

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \tag{2.1}$$

Here, H represents the hypothesis while X represents the data available. Both of these factors are used to determine the probability (represented by the P) [35]. From this basic equation, it is also said that Naive Bayes is most optimal for data that will be classified into two classes. As more classes are needed, the more the classifier degrades[36].

2.4 Simple Logistic Regression

Logistic regression is another optimal two-class classifier. This classifier produces a single output and has a logistic property. To express the single output, it can be said that to get the single outcome of y_i where $i = 1, ..., n$ [37]. Similar to other classifiers, Logistic Regression uses the *Objective Function*. This function maximizes the logistic likelihood of the proposed system. This can be represented through the given formula [38].

$$L = \sum_{i=1}^{n} (1 - x^i) log(1 - Pr[1|f_1^i, ..., f_m^i]) + x^i log(Pr[1|f_1^i, f_2^i, ..., f_m^i]) \quad (2.2)$$

In the given formula, m would be the number of features. Another representation of how to implement the logistic regression model is shown in Figure 2.7

Additive logistic regression

Model generation

for j = 1 to t iterations **do**:

for each feature vector **f**[i] **do**:

1: set the target value for the regression to

$$c[i] = (y[i] - P(1|\mathbf{f}[i]))/ [P(1|\mathbf{f}[i]) * (1 - P(1|\mathbf{f}[i])]$$

2: set the weight w[i] of instance **f**[i] to :

$$P(1|\mathbf{f}[i]) * (1-P(1|\mathbf{f}[i])$$

3: fit a regression model r[j] to the data with

class values c[i] and weights w[i]

Classification

Predict class 1 if $P(1|\mathbf{f}) > 0.5$, otherwise predict class 0

Figure 2.7. Pseudo-code for a Binary Logistic Regression Model

2.5 Evaluation Metrics

In order to validate a machine learning model and measure its performance, there are some metrics that need to be used. These metrics are called the Evaluation Metrics. However, different models have different metrics that need to be ideally used. This exploration mainly focuses binary classification models, therefore the metrics to be used are based on the concept of a class output, or a probability output. With that being said, the metrics used throughout the exploration include the calculation for an Accuracy Score, Precision Score, F-1 Score, Recall Value and ROC Area.

2.5.1 Confusion Matrix

The basis to the evaluation metrics used revolves around the values that are presented in a confusion matrix. Therefore an understanding of what the confusion matrix is becomes imminent. The confusion matrix has a size of N x N, with N being the number of classes, this matrix represents the findings of a classification process that compares the predicted and actual results. As an example, in a binary classification problem, a 2 by 2 matrix will be drawn as shown in an example in Figure 2.8. To easily decipher this, TP (True Positive) represents the correct positive predictions, FP (False Positive) represents the incorrectly predicted positives, FN (False Negative) represents the incorrect negative predictions, and lastly TN (True Negative) represents the correct negative predictions [39]. For the purpose of this exploration, the Negative value indicates that the application is benign, while the positive indicates that the application is a malware.

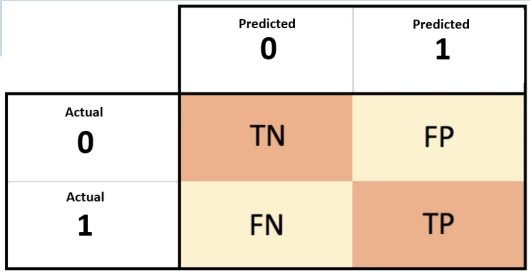

Figure 2.8. Confusion Matrix

2.5.2 Accuracy

Accuracy describes the measure of correctness a prediction has. The mathematical calculation shows the sum of two 'TRUE' predictions (True Positive and True Negative values) being divided by the total number of data used. This formula is as shown below [40].

$$ACC = \frac{TP+TN}{TP+TN+FN+FP} = \frac{TP+TN}{P+N} \tag{2.3}$$

2.5.3 Precision

Precision would be defined as the count of correctly predicted positive data points. The formula to calculate this value requires the amount of correctly pre-

dicted points, also called the True Positives, divided by the amount of all the positive data points. This formula can be written as shown in the equation below. This equation will produce a result between 0 and 1, in which if the value is closer to 1, then the model can be said to have a higher precision [41]. A higher value of precision will also mean there are less chances of a false positive.

$$ACC = \frac{TP}{TP + FP} \tag{2.4}$$

2.5.4 Recall

Recall would be the measure of the ability of a model to correctly identify a true positive from all the actual positives. To calculate this metric, the amount of True Positives is divided by the number of True Positives added with the number of False Negatives, this is shown in the equation below. The value calculated will range between 0 and 1, the value of 1 would represent that there are no False Negatives.

$$ACC = \frac{TP}{TP + FN} \tag{2.5}$$

2.5.5 F-1 Score

To find out the overall performance of a classification model, the F-1 Score can be calculated. Also referred to as the harmonic mean of the precision score and recall score. While the recall score is prioritised to prevent false negatives, and the precision score is prioritised to prevent false positives, but when both errors need to be prevented, the F-1 Score would be the most ideal. Here the formula, as shown below, combines the two to measure how well a model can correctly classify an observation into its proper class.

$$F-1 = 2x\frac{Precision * Recall}{Precision + Recall} \tag{2.6}$$

2.5.6 ROC Area

The ROC Curve, also known as the Receiver Operating Characteristic Curve is drawn to depict the relationship between the True Positive Rate, and the False Positive Rate within a threshold. To summarize an ROC Curve, the area under said graph is taken. This area, now called the ROC Area or AUC, sums up the ability of

a model in ranking predictions. A higher AUC value indicates that the model will have a higher probability in properly detecting a prediction.

To draw an ROC Curve, there are two main values that need to be calculated, these values are the True Positive Rate (TPR) and the False Positive Rate (FPR). TPR, also referred to as a measure of sensitivity, defines the amount of positive predictions that are correct within all the points predicted. TPR is calculated using the following formula [42]:

$$TPR = \frac{TruePositive}{FalseNegative + TruePositive} \tag{2.7}$$

On the other hand, FPR represents the other end of the spectrum where is measures the amount of data that has a negative point, but was predicted to be positive. To calculate it, the formula is as follows [40]:

$$FPR = \frac{FalsePositive}{TrueNegative + FalsePositive} \tag{2.8}$$

From both these values, the range is from 0 to 1, and as mentioned before, the relationship between the True Positive Rate and the False Positive Rate is drawn in threshold. Once all these values are calculated, all the points can be drawn up into a curve, and this curve will be the ROC Curve. Once the curve is drawn up, the area under it can simply be calculated to summarize the curve as a whole.