The Eighth International Conference on Business Information Security (BISEC-2016), 15th October 2016, Belgrade, Serbia

DETECTING MALICIOUS ANOMALIES IN IOT: ENSEMBLE LEARNERS AND INCOMPLETE DATASETS

IGOR FRANC
Belgrade Metropolitan University, Faculty of Information Technologies and SECIT Security Consulting, igor.franc@metropolitan.ac.rs

NEMANJA MAČEK
School of Electrical and Computer Engineering of Applied Studies, Belgrade and SECIT Security Consulting, nmacek@viser.edu.rs

MITKO BOGDANOSKI
Military Academy General Mihailo Apostolski, Skoplje, Macedonia, mitko.bogdanoski@ugd.edu.mk

ALEKSANDAR MIRKOVIĆ
eSigurnost Association, Belgrade and SECIT Security Consulting, amirkovic@secitsecurity.com

DRAGAN ĐOKIĆ
Belgrade Metropolitan University, Faculty of Information Technologies, dragan.djokic@metropolitan.ac.rs

Abstract: *Anomalies in IoT typically occur as a result of malicious activity. As an example, a point anomaly may occur once network intrusion is attempted, while collective anomaly may result from device being hacked. Due to the nature of the attacks, some anomalies are represented by incomplete captured instances or imbalanced captured datasets. For example, features may have some values missing from the row or may contain both categorical and numerical values. Once pre-processed, these datasets become suitable training sets for any machine learning classifier that detects anomalies. However, there are situations where pre-processing takes large amount of time in the operating phase or simply is not executable due to the nature of the data. For example, a feature that contains unknown number of categorical values, such as strings, cannot be converted into finite number of binary features before the training. In this scenarios, basic machine learning methods, such as Support Vector Machines or Decision Trees either fail to operate or provide poor classification performance. Unlike basic, ensemble learners manage these data instances efficiently and provide good anomaly detection rates. This paper analyses the performance of ensemble learners on incomplete IoT intrusion datasets, represented by point anomalies.*

Keywords: *Datasets, Anomaly, Ensemble, Boosting, GentleBoost*

1. INTRODUCTION

Anomalies are patterns in data that do not conform to a well-defined notion of normal behavior. As an example, anomalies in a simple two-dimensional dataset are points that are sufficiently far away from normal regions, i.e. regions that most points belong to. “Anomaly detection refers to the problem of finding patterns in data that does not conform to expected behavior” [1]. That being said, anomalies are detected by defining a region that represents normal behavior and declaring any pattern in the data that does not belong to this normal region as an anomaly. Although anomaly detection seems to be straightforward, several factors make this apparently simple task very challenging: defining a normal region that encompasses every possible normal behavior is difficult; normal behavior may evolve with time and current notion might not be representative in the future; the boundary between normal behavior and anomalies is often not precise and the lack of labeled instances for training may cause a problem. Consequently, the algorithm suitable for anomaly detection in all domains does not exist. The choice of technique suitable for the specific problem is based on the nature of the input data, the amount of labeled instances in the training set and the type of anomaly.

The nature of the input data refers to the types of features that describe instances in the training set. Features can be binary, categorical (they can take one from the finite number of values) or continuous. Multivariate instances may contain features of same type or a mixture of different data types [2]. According to the amount of labelled instances in the training set, one of the following techniques is used: supervised detection (all instances are labelled), semi-supervised techniques (either the instances belonging to normal or anomalous behaviour are labelled) and unsupervised techniques, that requires no training data, as they are based on assumption that normal instances are far more frequent than anomalies in the test data. Anomalies can be classified as point, contextual and collective anomalies [1]: point anomalies are individual data instances that are anomalous, contextual anomalies are data instances that are anomalous in a specific context, and collective anomalies are collections of related data instances that are anomalous with respect to the entire data set.

In the Internet of Things, anomalies in the network traffic may indicate an ongoing malicious activity, such as network intrusion, eavesdropping, or a Thing in IoT being hacked or compromised in another way. While on the typical network various anomaly based intrusion detection systems use pre-processors to convert feature values into numerical values, IoT scenario may be a little bit different. Attacks in IoT differ due to vast variety of devices, which further causes some anomalies to be represented by incomplete feature vectors or imbalanced datasets. Features may have some values missing or may contain categorical and numerical values. Although pre-processors are typically used to resolve these issues, sometimes they cannot be implemented due to the nature of data: one cannot employ conversion from categorical to numerical features if number of categorical values is prior unknown. In these case basic machine learning methods fail to operate – Support Vector Machines operate with standardized numerical datasets, while Decision Trees cannot perform with sufficient precision as some nodes cannot be traversed down to the leafs due to lack of values. However, ensemble learners operate with sufficient precision and provide high anomaly detection accuracy. Having that said, within this paper, the efficiency of supervised ensemble machine learning methods on incomplete synthetic IoT intrusion datasets, represented by point anomalies and healthy traffic.

2. MACHINE LEARNING ALGORITHMS

Tom Mitchell's widely quoted formal definition of machine learning [3] can be rephrased to the supervised anomaly detection context: "a learner learns to classify events (task *T*) into normal events and anomalies; performance measure *P* of this task is the classification accuracy, and the experience *E* is the training set of rules". Supervised learning algorithms build a model from a training set (given in the form of feature vectors) with class label assigned to each instance. Once trained, supervised algorithms assign class labels to previously unseen examples of the same task [4]. Within the context of anomaly detection, typically a two-class problem is being solved and having that said, class labels given to the instances indicate normal or anomalous data.

In theory, every machine learning method has its own advantages and disadvantages, which can be perceived on the basis of how these methods operate. Decision trees recursively create a classification tree with decision nodes based on a subset of values of a corresponding feature and classifications on its leaves. Support vectors map learning examples from an input space to a new high dimensional (potentially infinite) feature space in which examples are linearly separable (see Image 1). Naive Bayes apply Bayes' rule using assumption about mutual independence of features, and k-Nearest Neighbours assign class labels according to a classification of *k* closest training examples in feature space. However, there is no machine learning method that is the best for every problem (the Generalization Conservation Law [5] or the No Free Lunch Theorem [6]).

Machine learning methods used for classification can be divided into [7]: basic methods (artificial neural networks [8], Support Vector Machines [9, 10], decision trees [11, 12], Naive Bayes [13, 14]), hybrid methods (for example, a hybrid of decision trees and Naive Bayes – a regular univariate decision tree, where leaves contain a naive Bayes classifier built from the examples that fall at that leaf [15]), incremental methods (Naive Bayes updatable), hybrid incremental methods (Hoeffding Tree [16]), basic ensembles (random forest [17]), hybrid ensembles (stacking) and hybrid incremental ensembles (Ada Hoeffding option tree).

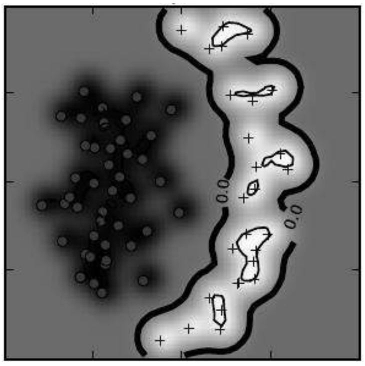

Image 1: SVM's RBF kernel maps data from input space to high dimensional feature space

3. ENSEMBLES

Ensemble machine learning methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone [18, 19], as shown on Image 2.

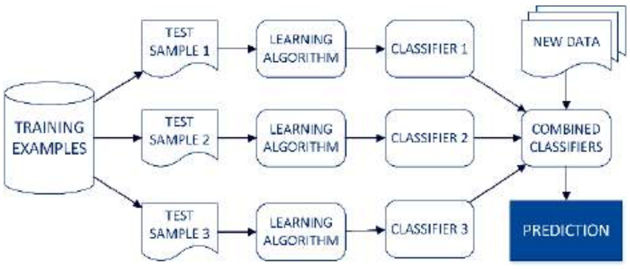

Image 2: Ensemble machine learning

Most common types of ensembles used in experiments reported in the literature are built by boosting, bagging and stacking. Boosting incrementally builds an ensemble: each new model instance is trained in order to emphasize the training instances that previous models have misclassified. Each model in the bagging ensemble votes with equal weight, and in order to promote model variance, bagging trains each model in the ensemble using a randomly drawn subset of the training set. For example, the random forest [17] combines random decision trees with bagging to achieve very high classification accuracy [20]. In some cases, boosting has been reported to provide better accuracy than bagging, but is less prone to over-fitting.

Stacking builds a hybrid ensemble by training a learning algorithm (combiner) to combine the predictions of several other learning algorithms. All of the other algorithms are trained using the available data and a combiner is further trained to make a final prediction using all the predictions of the other algorithms as additional inputs. In practice, a single-layer logistic regression model is often used as the combiner.

4. GENERATING IOT DATASET

The IoT dataset used in this research is built from traffic captured on the simulated network of Things, consisting mostly of mobile devices. All devices had their traffic rerouted through a single gateway where it has been captured using PCAP library on Linux operating system. Synthetic dataset consists of normal, healthy traffic recorded during one day period and variety of simulated attacks, ranging from vulnerability analysis to penetration attempts and successful exploitations, executed with variety of open source and commercial software products. Both healthy and malicious traffic have been recorded separately and cleansed from other protocol and service leftovers (partial noisy data removal), thus leaving clean normal and anomalous PCAP files, which reassembles a scenario for supervised anomaly detection. QoSilent Argus software was used to extract features values from PCAPs and create data instances which were labelled and shuffled into a separate training and test sets. Features used in this research do not include source and destination IP addresses. However, they include flags, connection states, protocols, port numbers and lots of statistical data. Once the feature extraction was done, a sneak peek into the generated CSV revealed the following facts that point up to incompleteness of the dataset.

1. Feature flags is categorical, but have fields with no values (blanks). Although somewhat convertible to numerical values, a change in the Argus software may result in need to change in the pre-processor.

2. Source and destination ports have decimal, hexadecimal and textual values. Examples of the values include: “51305”, “0xb6”, “netbios-dgm”, see Table 1 for more examples. As number of textual values is not documented in the software documentation, unknown number of textual values cannot be converted into finite number of numerical values during the training phase unless the model will be retrained on frequent basis.

Table 1: Example values of port features

Sport value	Sport type	Dport value	Dport type
33100	Decimal	https	String
0x0008	Hex	0x0100	Hex
35360	Decimal	domain	String
37159	Decimal	https	String
15039	Decimal	http	String

3. Source and destination TOS, TTL, TCP window advertisements and several other numerical features have blanks, which makes them virtually inconvertible into a numeric values, if a range of numerical values they make take is unknown or undocumented.

4. Source and destination diff service have both numerical and categorical values. Categorical values are not documented as well as the range of numerical values, thus making this field inconvertible to numerical values.

Although one might try to implement conversion to numerical values in the data pre-processor (for example, filling blanks with -1 or splitting features with mixed values into two or three features), aforementioned statements indicate that in this scenario it is not possible due to unknown or undocumented ranges. This leaves a learner to be trained and evaluated with incomplete datasets – sets from which aforementioned features are removed.

5. EXPERIMENTS

Performance of the basic and ensemble machine learning algorithms solution is experimentally evaluated using MATLAB R2016a with Statistical and Machine Learning Toolbox, version 10.2. Within this research the following machine learning algorithms available in aforementioned toolbox have been used to train and test IoT datasets: basic methods (decision tree and Support Vector Machines), bagging and boosting ensembles (Adaboost [21], RUSBoost [22], LogitBoost [23] and GentleBoost [24], all using C4.5 decision tree as a base learner). A training dataset consisting of one thousand lines with 41 features and a class label was imported into MATLAB. When imported into the Classification Learner, 11 features were discarded due to the feature values inconsistencies (inability of being pre-processed, as stated in Section 4 of this paper) and several additional categorical for SVMs. Five-fold cross validation was applied and the testing results for each classifier are listed below.

1. *Complex tree*.

- C4.5 decision tree using Gini’s diversity index as split criterion, with 100 splits and no surrogate decision splits.

- Overall accuracy of the classifier is 73.7%.

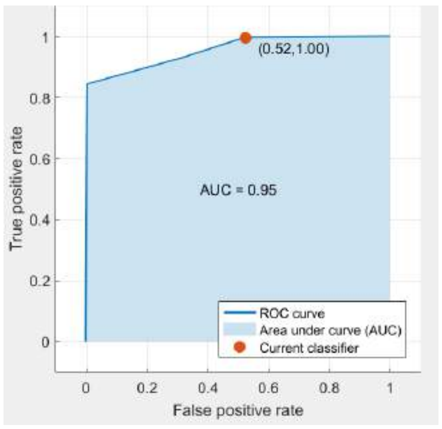

Image 3: Complex Tree ROC Curve

2. *Fine Gaussian SVM*.

- Standardized data, Gaussian kernel function, manual kernel scale mode 1.4.

- Overall accuracy of the classifier is 71.6%.

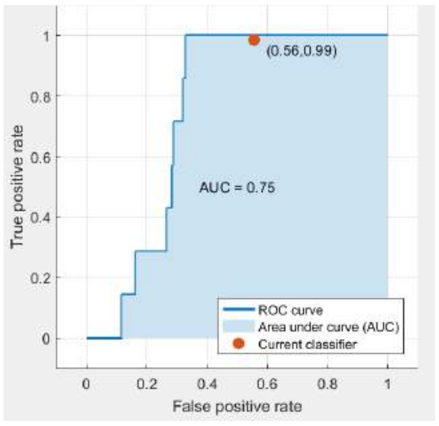

Image 4: Fine Gaussian SVM ROC Curve

3. *Bagged trees*.

- Decision tree as a learner, 20 splits, 30 learners, 0.1 learning rate, subspace dimension 1.

- Overall accuracy of the classifier is 89.0%.

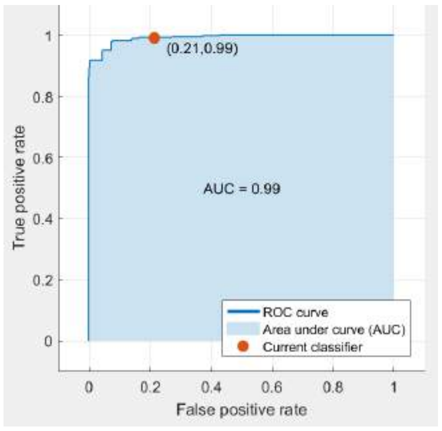

Image 5: Bagged Trees ROC Curve

Boosted trees are further examined with 20 splits, 30 learners, 0.1 learning rate and subspace dimension 1.

4. *AdaBoost*.

- Overall accuracy of the classifier is 94.8%.

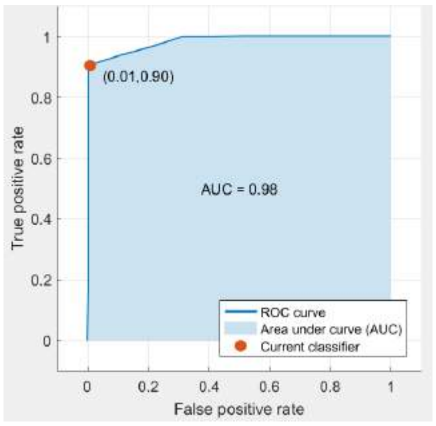

Image 6: AdaBoost ROC Curve

5. *RUSBoost*.

- Overall accuracy of the classifier is 73.7%.

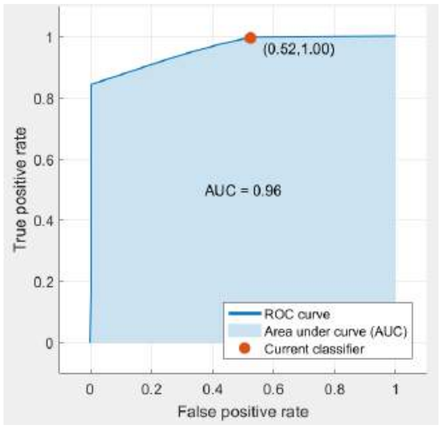

Image 7: RUSBoost ROC Curve

6. *LogitBoost*.

- Overall accuracy of the classifier is 91.9%.

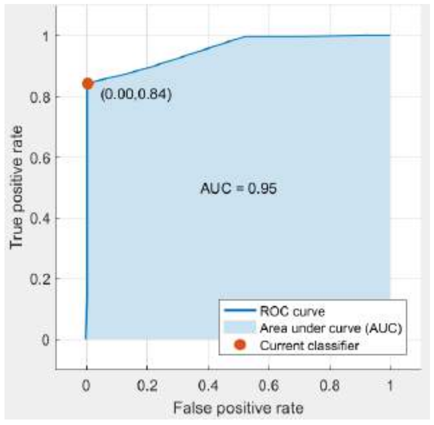

Image 8: RUSBoost ROC Curve

7. *GentleBoost*.

- Overall accuracy of the classifier is 98.6%.

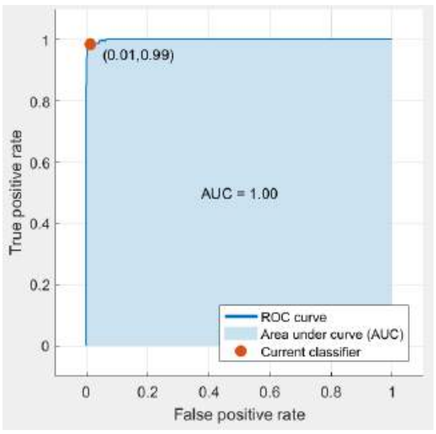

Image 9: GentleBoost ROC Curve

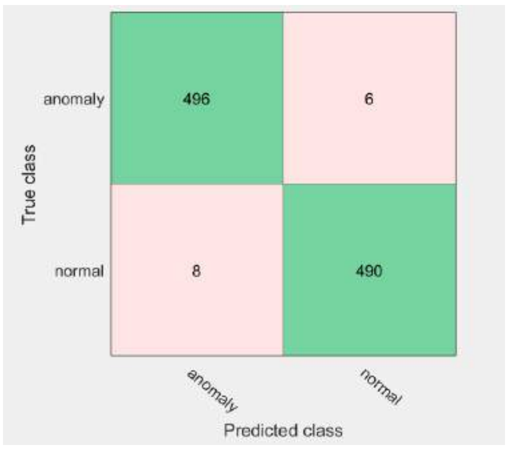

Image 10: GentleBoost confussion matrix

Results are further summarized in Table 2.

Table 2: Experimental evaluation summary

Classifier	Accuracy	AUC
Complex Tree	73.7%	0.95
Fine Gaussian SVM	71.6%	0.75
Bagged Trees	89.0%	0.99
AdaBoost	94.8%	0.98
RUSBoost	73.7%	0.96
LogitBoost	91.9%	0.95
GentleBoost	98.6%	~1

Although we have already stated and explained why basic machine learning methods are not expected to operate with high classification accuracy, and ensembles were expected to, another question arises: why did the GentleBoost provide such level of accuracy on the dataset and have outperformed other ensembles?

Unlike the commonly used AdaBoost algorithm, the weak classifier in the GentleBoost algorithm is a soft-decision classifier with continuous output. This enables the strong classifier's score to be smoother and favourable for computing derivatives [25]. More formally, while other boosting algorithms minimize the overall test error as much as possible at each step, GentleBoost features a bounded step size. Let $w_{t,i}$ denote update weights, y_i desired outputs, $h_t(x)$ weak learners, x the samples and α_t the minimizer. Variable f_t is chosen to minimize:

$$\sum_i w_{t,i}\left(y_i - f_t\left(x_i\right)\right)^2, \tag{1}$$

and no further coefficient is applied. GentleBoost will choose:

$$f_t\left(x\right) = \alpha_t h_t\left(x\right) \tag{2}$$

exactly equal to y, while steepest descent algorithms will try to set $\alpha_t=\infty$. According to empirical observations, this causes good performance of GentleBoost, even with incomplete datasets, as large values of α can lead to poor generalization performance [26, 27]. Second, the GentleBoost algorithm outperforms other boosting methods in that it is more robust to noisy data and more resistant to outliers.

6. CONCLUSION

The Internet of Things involves the increasing prevalence of objects and entities (Things) provided with unique identifiers and the ability to automatically transfer data over a network. Much of the increase in IoT communication comes from computing devices and embedded sensor systems used in industrial communication, home and building automation, vehicle to vehicle communication and wearable computing devices. IoT security is the area of endeavour concerned with safeguarding connected devices and networks in the IoT. As we have stated before, anomalies in the IoT traffic may occur as a result of malicious activities, such as attempt to hack or otherwise compromise a mobile device. These anomalies can be represented by a finite number of numerical or categorical features. In most scenarios pre-processors are able to convert all this data to numeric values and most of the machine learning algorithms are able to perform supervised anomaly detection after. However, due to a large number of different devices and variety of attacks, data may be incomplete and unsuitable to train basic learners such as decision trees or Support Vector Machines, or, even if trained, they will provide poor classification performance. Ensemble learners manage to build models and classify these data instances, which has been experimentally proven in this paper. Another issue that has arisen from the experimental evaluation presented in this paper is superiority of GentleBoost algorithm over other boosted ensembles on incomplete datasets resulting from soft-decision classification with continuous output and robustness to noisy data.

REFERENCES

[1] V. Chandola, A. Banerjee, V., “Anomaly detection: A survey”, Technical Report, TR 07-017, Department of Computer Science and Engineering, University of Minnesota, August 15, 2007. ACM Computing Surveys (CSUR), 41(3), 15.

[2] P. N. Tan, M. Steinbach, V Kumar, “Introduction to data mining”. Addison-Wesley, 2006.

[3] T. Mitchell, “Machine Learning”, McGraw-Hill Science/Engineering/Math, 1997.

[4] I. Hendrickx, “Local Classification and Global Estimation: Explorations of the k-nearest neighbor algorithm”, PhD Thesis, Tilburg University, The Netherlands, 2005.

[5] C. Schaffer, “A Conservation Law for Generalization Performance”, in Proceedings of the Twelfth International Conference on Machine Learning, pp. 259-265, New Brunswick, NJ: Morgan Kaufmann, 1994.

[6] D. H. Wolpert, “The lack of a prior distinctions between learning algorithms and the existence of a priori distinctions between learning algorithms", Neural Computation, 8, 1341–1390, 1391–1421, 1996.

[7] V. Miškovic, M. Milosavljević, S. Adamović, A. Jevremović, “Application of Hybrid Incremental Machine Learning Methods to Anomaly Based Intrusion Detection”, Proceedings of 1st International Conference on Electrical, Electronic and Computing Engineering IcETRAN 2014, Vrnjačka Banja, Serbia, June 2-5, 2014, pp. VII2.3.1-6.

[8] S. Haykin, “Neural Networks: A Comprehensive Foundation, 2nd ed.”, Prentice Hall, 1998.

[9] V. Shawe-Taylor, N. Cristianini, “Kernel Methods for Pattern Analysis”, Cambridge University Press, 2004.

[10] V. Vapnik, “Statistical Learning Theory”, John Wiley & Sons, 1998.

[11] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, “Classification and Regresssion Trees”, Wadsworth, Belmont, 1984.

[12] R. Quinlan “C4.5: Programs for machine learning”, Morgan Kaufmann Publishers, Inc., 1993.

[13] V. Cherkassky, F. M. Mulier, “Learning from Data: Concepts, Theory and Methods. 2nd ed.”, John Wiley - IEEE Press, 2007.

[14] I. H. Witten, E. Frank, M. A. Hall, “Data Mining: Practical machine Learning Tools and Techniques, 3rdEd”, Elsevier Inc., 2011.

[15] R. Kohavi “Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid”, in KDD (pp. 202-207), 1996.

[16] P. Domingos, G. Hulten, “Mining high-speed data streams” In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 71-80, 2000, ACM.

[17] L. Breiman, “Random Forests”, Machine learning, 45(1), pp. 5-32, 2001.

[18] R. Polikar, “Ensemble based systems in decision making”, IEEE Circuits and Systems Magazine, 6 (3), pp. 21–45, 2006.

[19] L. Rokach, “Ensemble-based classifiers”, Artificial Intelligence Review, 33 (1-2): 1–39, 2010.

[20] L. Breiman, “Bagging Predictors”, Machine Learning, 24(2), pp.123-140, 1996.

[21] B. Kégl, “The return of AdaBoost.MH: multi-class Hamming trees”, arXiv: 1312.6086, Dec. 20. Last time visited: Aug 15, 2016.

[22] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, A. Napolitano, “RUSBoost: A hybrid approach to alleviating class imbalance”, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 40(1), pp.185-197, 2010.

[23] S.B. Kotsiantis, “Logitboost of simple bayesian classifier”, Informatica, 29(1), 2005.

[24] J. Friedman, T. Hastie, R. Tibshirani, “Additive logistic regression: A statistical view of boosting”, The Annals of Statistics, 38(2):337–374, 2000.

[25] X. Liu, T. Yu, “Gradient feature selection for online boosting”, in 2007 IEEE 11th International Conference on Computer Vision, pp. 1-8. IEEE, 2007.

[26] R. E. Schapire, Y. Singer, “Improved boosting algorithms using confidence-rated predictions”, Machine learning, 37(3), pp.297-336, 1999.

[27] Y. Freund, R. Schapire, N. Abe, “A short introduction to boosting”, Journal-Japanese Society For Artificial Intelligence 14(771-780), p.1612, 1999.