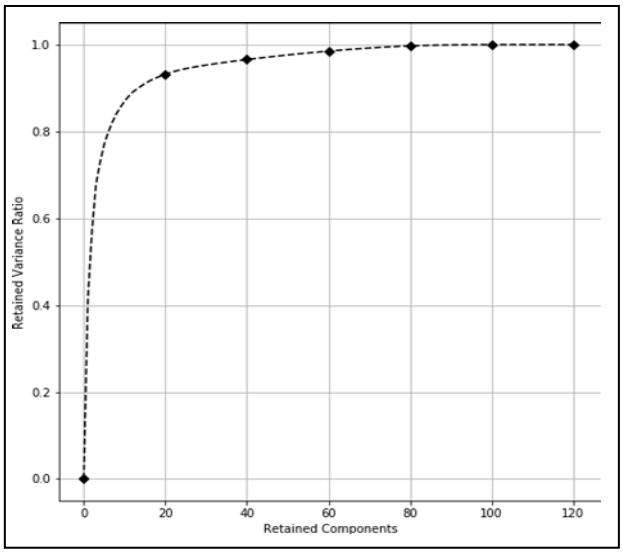
classified as normal here as well as at the binary classification stage form the combined population of benign data points. Every other data point is categorized first as malicious and then further into the type of attack.

5 Experiment and Results

In this research, we have used scikit-learn and keras with a tensorflow backend for the construction of our models and we have used matplotlib for generating graphs. All of the models have been trained on Google Colaboratory using a Tesla K80 single core GPU with 25.51 GB RAM, although RAM usage throughout our experiments was limited to 10 GB.

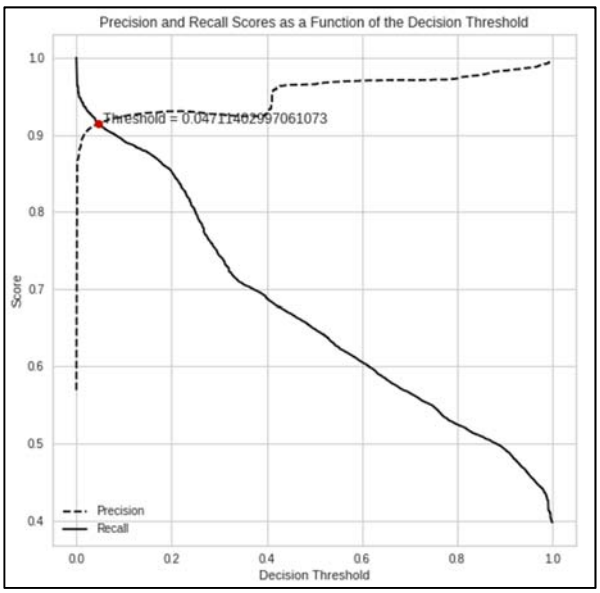
During feature engineering, we applied PCA for dimensionality reduction. The following graph shows the retention of variance corresponding to the number of components retained.

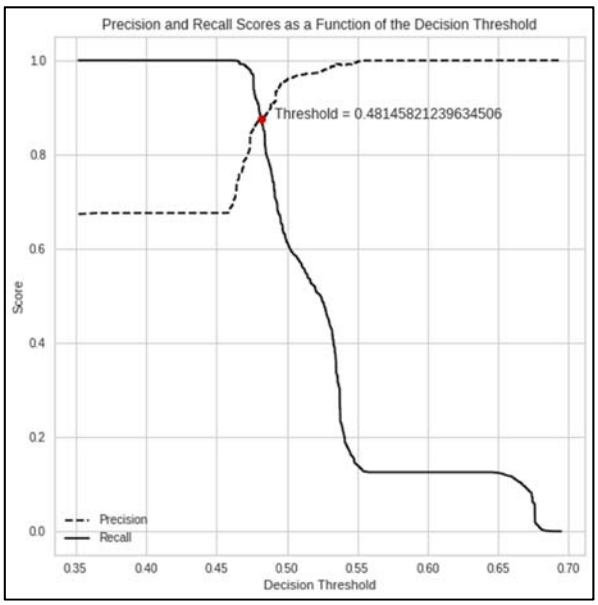
Fig.4 Variance Retention vs. Number of Components.

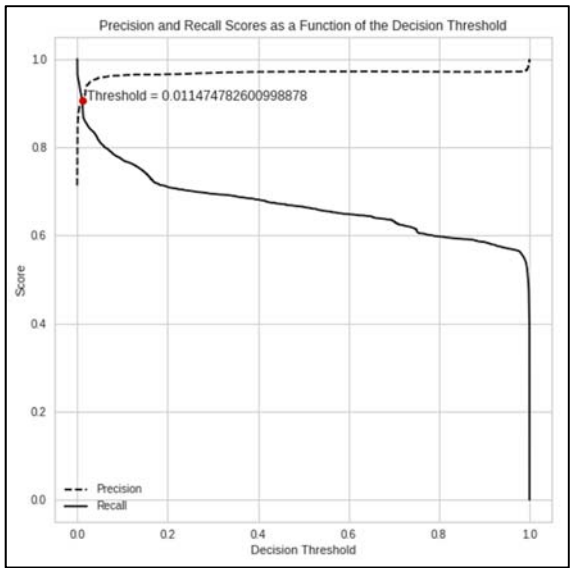


For binary classification we have compares 3 methods: Random Forest Classifiers and boosting algorithms AdaBoost and XGBoost. To obtain the optimal set of parameters for each of these classifiers, grid search was run on a suitable search space with recall as the optimizing metric. This was followed by comparing plots (Figure 5) of precision and recall and finding the optimal threshold for a point to be classified as malicious. Threshold tuning was based on the fact that we weren't looking for very high precision values at this stage; recall was of primary importance. Experiments suggested that threshold values that brought precision and recall closest together could be considered as optimal threshold points as the relation between these 2 metrics is almost inverse in nature, which means that increasing recall via threshold tuning would result in reduces precision.

Fig.5 Precision and recall scores as a function of the decision threshold of Random Forest, AdaBoost, and XGBoost respectively.



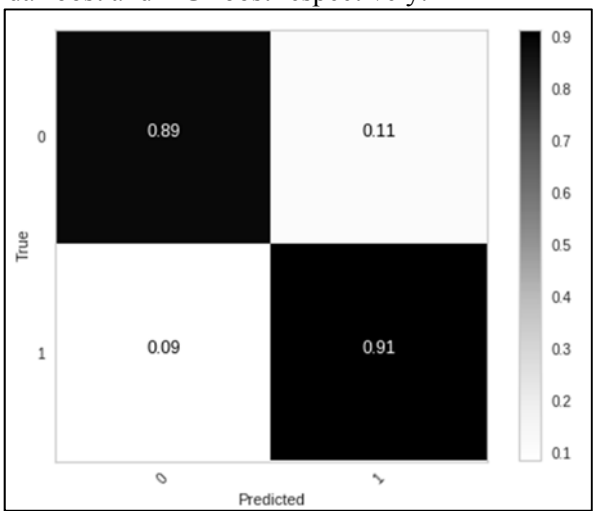


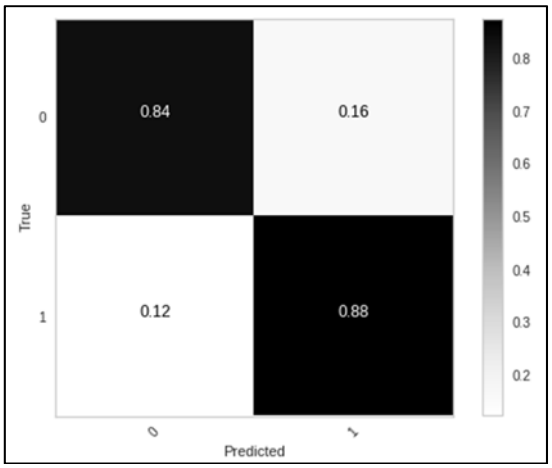


Random Forest Classifiers gave the best classification output overall (Fig.6) It and XGBoost had a recall of 91%, while AdaBoost gave a recall of 88%. Random Forest Classifiers gave a maximum precision of 89% while AdaBoost and XGBoost gave a precision of 84% and 88% respectively.

XGBoost had the maximum area under curve (AUC) of 0.96275, while Random Forest Classifiers and AdaBoost had an AUC of 0.94587 and 0.94139 respectively (Fig.7).

Fig.6 Confusion matrices of Random Forest, AdaBoost and XGBoost respectively.





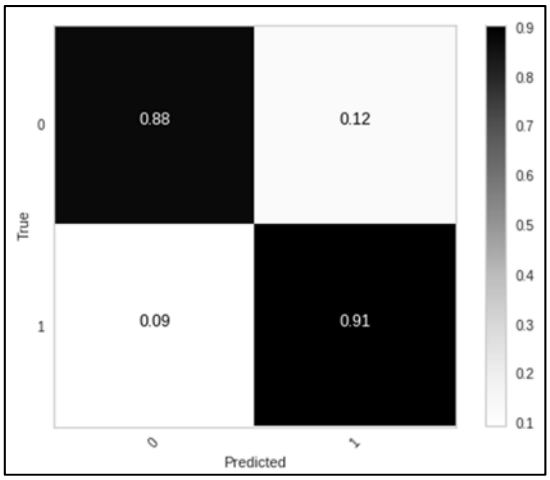
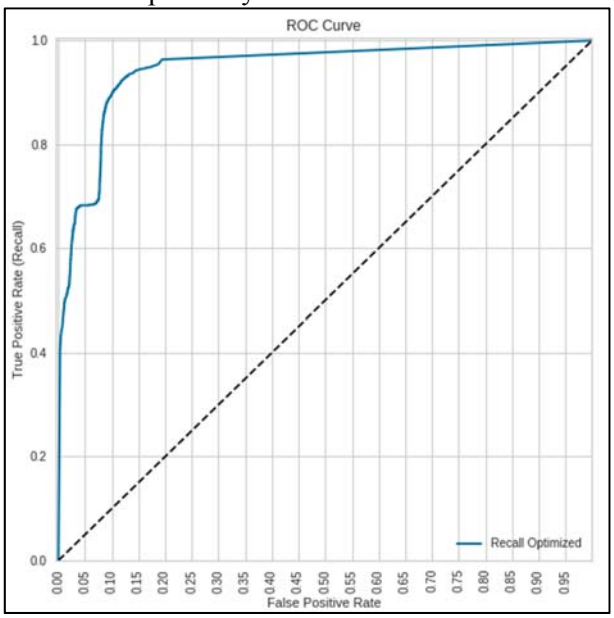
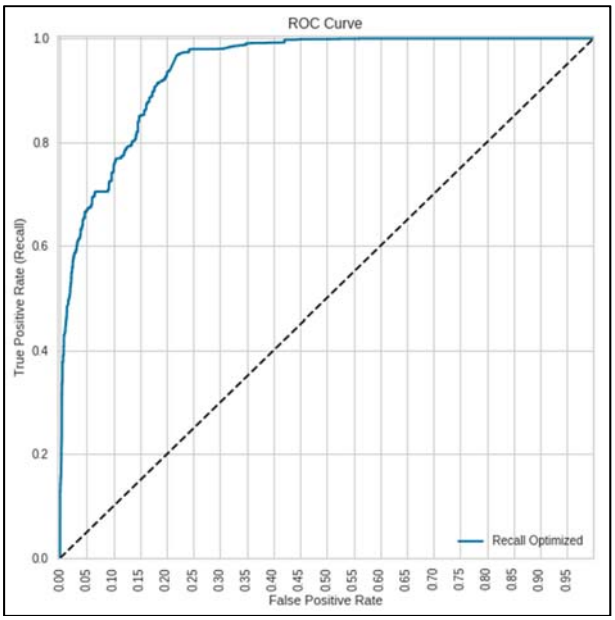
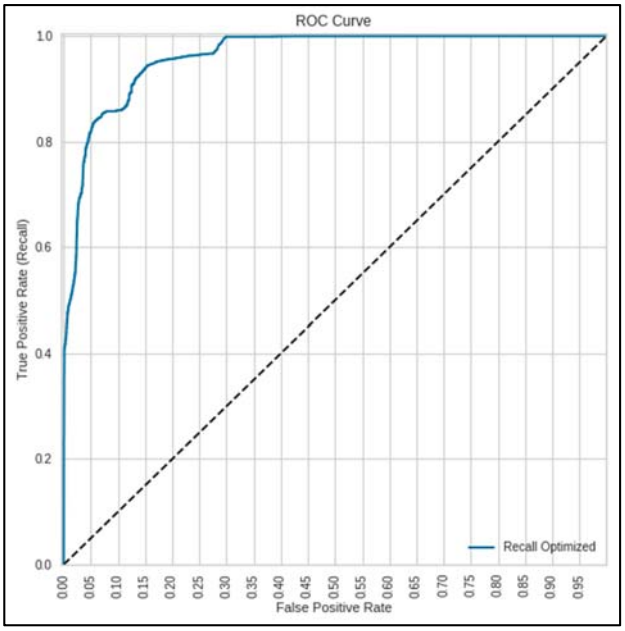


Fig.7 ROC curves of Random Forest, AdaBoost and XGBoost respectively

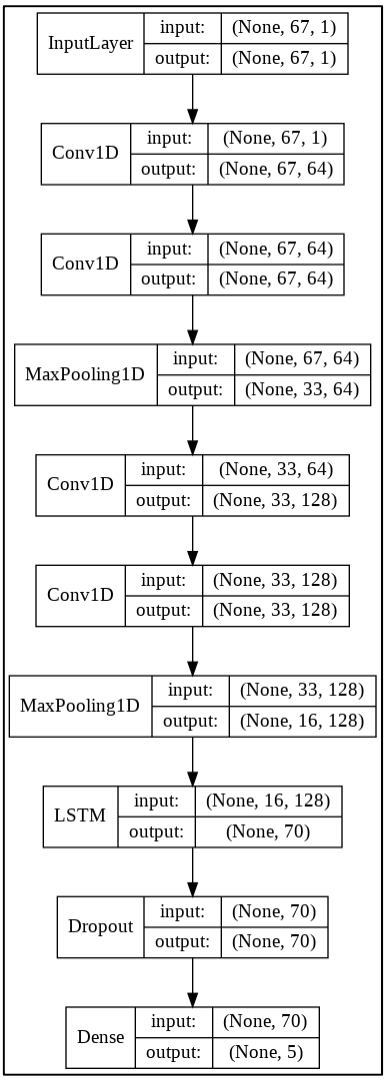






For multi-class classification, multiple neural network architectures were considered comprising of CNNs, RNNs, LSTMs, etc. The following architecture, as suggested by [21] yielded best results when trained on 100 epochs.

Fig.8 Architecture of the ANN



The loss convergence and accuracy over a course of 100 epochs are shown by the following graph.

Fig.9 Loss convergence of the model

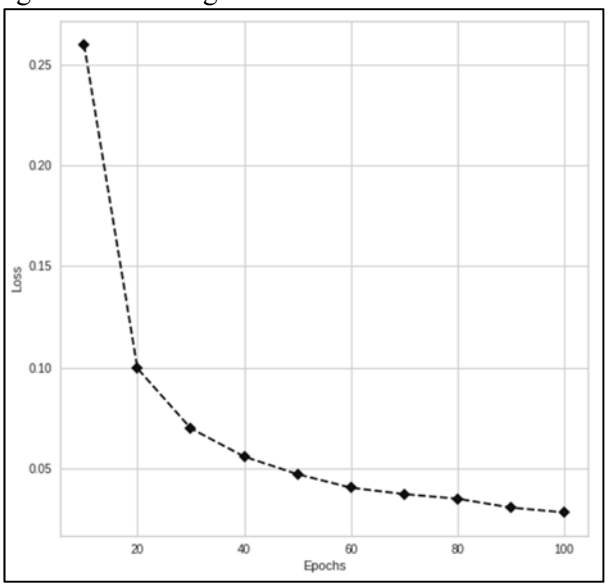
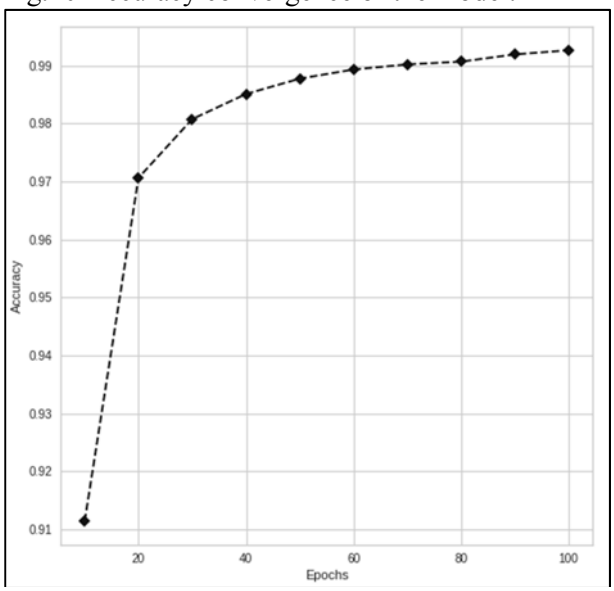
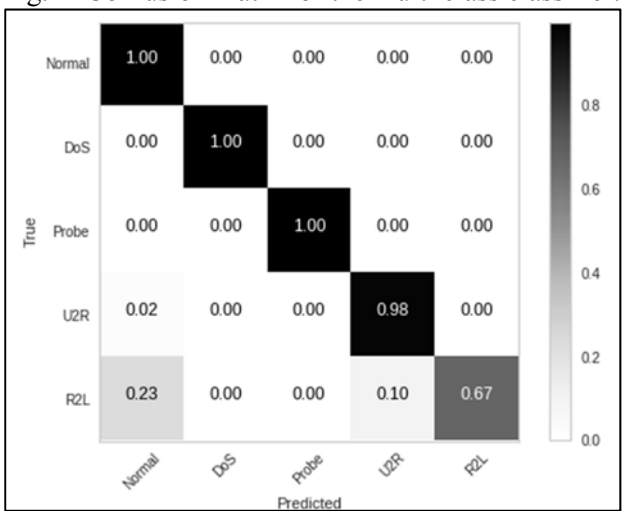


Fig.10 Accuracy convergence of the model.



The combined model, along with all binary classifiers in consideration combined with the multi class classifier gave the following results.

Fig.11 Confusion matrix of the multi class classifier.



6 Further Analysis

In this section, we resort to using various methods to help understand the importance of each feature viz a viz the information it provides in aiding the classification process. Three standard techniques have been used for the same, which are mentioned and further elaborated upon below:

6.1 Decision Tree Classifiers

For categorical variables, we used the weighted Gini index criterion after doing one hot encoding on the variables. Each individual categorical variable was separated into binary attributes. The weighted Gini index was calculated using the following formula:

$$W_G(A) = \sum_{a_i \in A} w(a_i) G(a_i)$$

$$w(a_i) = \frac{entries\ with\ value\ a_i\ in\ column\ A}{total\ entries\ in\ dataset}$$

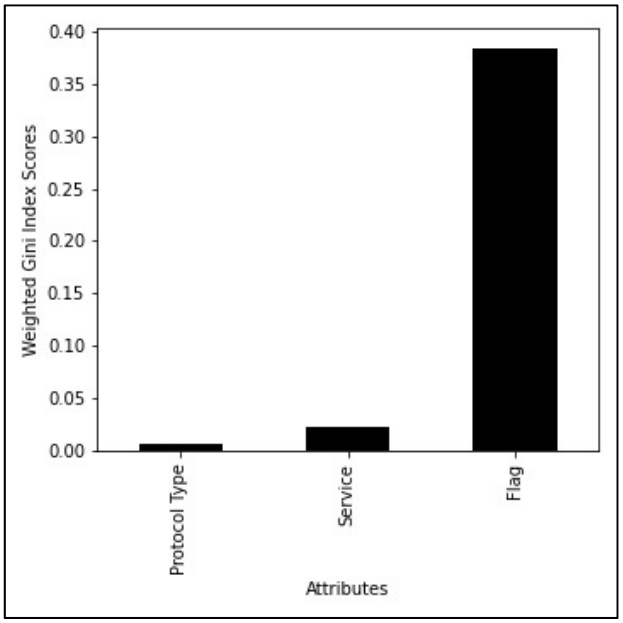
$$G(a_i) = Gini\ Index\ calculated\ for\ a_i$$

(2)

Categorical variables in the dataset have been inputted into a decision tree classifier to evaluate the importance of each in the classification process. This is measured as the total reduction of the Gini importance criterion caused due to each feature. The Gini Importance, also known as the Mean Decrease in Impurity is calculated as the number of times a feature is used to split a node, weighted according to the number of samples split at each node in a decision tree.

The conclusion in this regard is that while the Flag and Service variables are crucial in detecting intrusions, while the Protocol Type is not so important owing to its low Gini scores.

Fig.12 Feature Importance's using Decision Tree Classifier



6.2 Univariate Selection

For nominal features, we compared their importance using a chi square test (χ^2) which is commonly used for testing relationships between categorical variables. The null hypothesis of the Chi-Square test is that no relationship exists on the categorical variables in the population; they are independent.

We see from the following graph that these are the top 10 features we have identified in order of their importance.

Fig.13 Univariate Selection – Top 10 Features

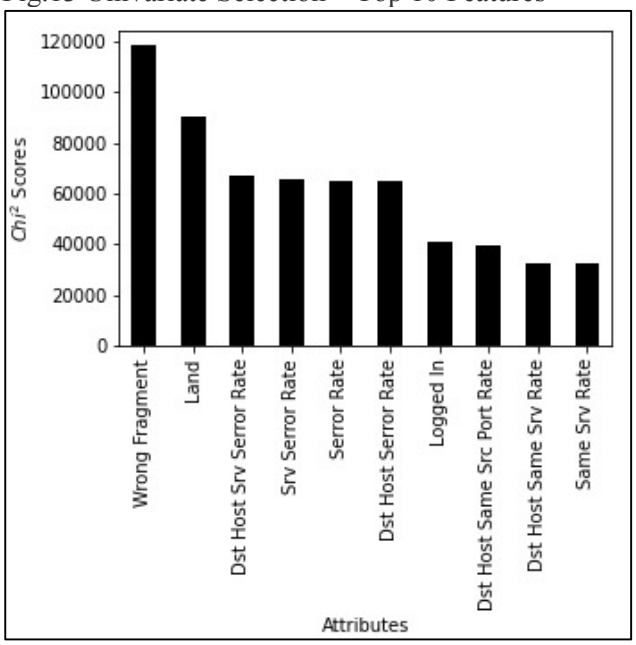
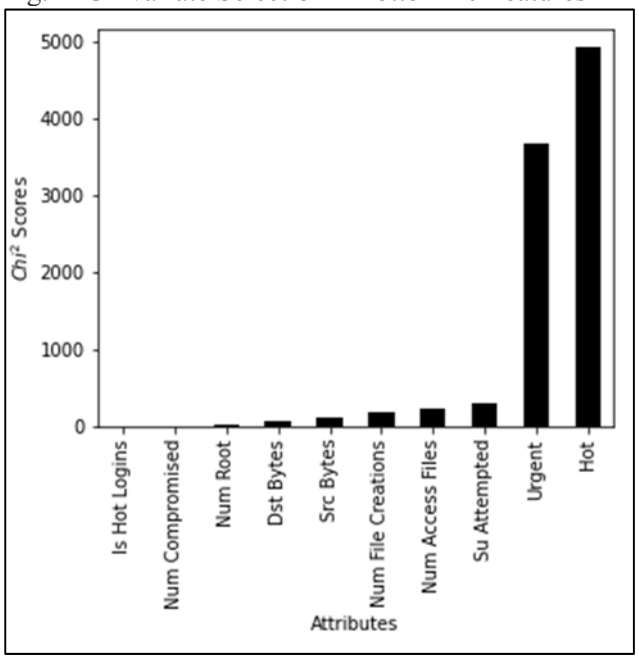


Fig.14 Univariate Selection – Bottom 10 Features



6.2 Correlation Plot

Correlation plots are used to determine the degree of correlation between every pair of features in a dataset. Each element corresponds to the Pearson's correlation coefficient between the two variables. Naturally, it is a symmetric matrix. Higher the correlation coefficient between two features, the stronger the linear relationship between them. Hence, each coefficient can be considered as a measure of the linear dependency between two variables. Ideally, we want a linearly independent set of features to maximize information present in the data and minimize redundancy. Assuming a threshold of 0.95 (or 95% correlation), we can identify 5 pairs of highly correlated features.

Fig.15 Correlation Heatmap with 95% Threshold

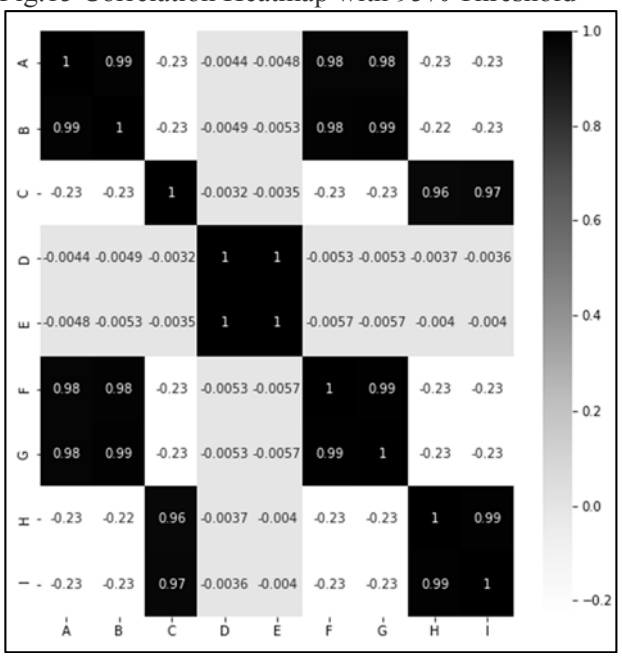


Table 2. Legend for Correlation Heatmap

A	Dst_Host_Serror_Rate
B	Dst_Host_Srv_Serror_Rate
C	Dst_Host_Srv_Rerror_Rate
D	Num_Compromised
E	Num_Root
F	Serror_Rate
G	Srv_Serror_Rate
H	Rerror_Rate
I	Srv_Rerror_Rate

7 Conclusion and Future Scope

The two-stage classification algorithm we considered here has resulted in much better evaluation results than the methods we explored. A reduced feature space ensures forward propagation through the model to work fast so that detection overhead is minimized.

Based on the further analysis, a total of 9 features: 8 numerical and 1 nominal, can be eliminated, reducing our feature space from 122 to 111. As the elimination has helped reduce redundancies in the dataset, we can expect our

model to perform better due to advanced deep learning algorithms on the new dataset. There is further scope for removal based on correlation plot data which would require further analysis.

Future scopes for this architecture include using GANs for generating adversarial samples that can compensate for the lesser proportion of U2R and R2L samples and generate a more global feature space that would result in better model parameters and would detect a larger proportion of attacks. A number of other over sampling techniques can be compared to the results we have obtained in this paper. The model can be trained on the new feature space for an expected enhanced performance.

References:

[1] A. Divekar, M. Parekh, V. Savla, R. Mishra and M Shirole, Benchmarking datasets for Anomaly-based Intrusion Detection: KDD CUP 99 alternatives, *Proceedings on 2018 IEEE 3rd International Conference on Computing, Communication and Security, ICCCS 2018*, 2018

[2] M. Roesch, Snort – Lightweight Intrusion Detection for Networks, *Proceedings of LISA '99: 13th Systems Administration Conference,* Seattle, Washington, USA, 1999.

[3] G. Wang, J. Hao, J. Ma, L. Huang, A New Approach to Intrusion Detecting using Artificial Neural Networks and Fuzzy Clustering, *Expert Systems with Application,* vol. 37, no. 9, pp. 6225-6232, 2010

[4] S. Shraddha, Intrusion Detection using Fuzzy Clustering and Artificial Neural Network, *Advances in Neural Networks, Fuzzy Systems and Artificial Intelligence,* pp. 209-217

[5] K. Peng, V. C. M. Leung and Q. Huang, Clustering Approach Based on Mini Batch Kmeans for Intrusion Detection System Over Big Data, *IEEE Access,* vol. 6, no. 1, pp. 11897-11906, 2018

[6] S. Potluri and C. Diedrich, Accelerated Deep Neural Networks for Enhanced Intrusion Detection System, *IEEE International Conference on Emerging Technologies and Factory Automation, EFTA,* 2016

[7] C. Yin, Y. Zhu, J. Fei and X. He, A Deep Learning Approach for Intrusion Detection using Recurrent Neural Networks, *IEEE Access,* vol. 5, no. 1, pp. 21954-21961, 2017

[8] J. Kim, N. Shin, S. Y. Jo and S. H. Kim, Method of Intrusion Detection Using Deep Neural Network, in *IEEE*, 2017

[9] Y. LeCun, Y. Bengio and G. Hinton, Deep Learning, *Nature,* vol. 521, no. 1, pp. 436-444, 2015

[10] J. Kim, J. Kim, H. L. T. Thu and H. Kim, "Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection," *IEEE*, 2016

[11] Z. Chiba, A. Noreddine, K. Moussaid, A. E. Omri, M. Rida, Intelligent and Improved Self-Adaptive Anomaly Based Detection System for Networks, *International Journal of Communication Networks and Information Security,* vol. 11, no. 2, pp. 312-330, 2019

[12] P. Manandhar and Z. Aung, Intrusion Detection Based on Outlier Detection Method, *International conference on Intelligent Systems, Data Mining and Information Technology,* Bangkok, Thailand, 2014

[13] J. Vacca, Computer and Information Security Handbook*, Elsevier*, 2009.

[14] M. E. Tipping and C. M. Bishop, Mixtures of Probabilistic Principal Component Analysers, *Journal of the Royal Statistical Society,* pp. 443-482, 1999

[15] K. Rai, M. S. Devi and A. Guleria, Decision Tree Based Algorithm for Intrusion Detection, *Advanced Networking and Applications,* vol. 7, no. 4, pp. 2828-2834, 2016

[16] D. P. Gaikwad and R. C. Thool, Intrusion Detection System using Bagging Ensemble Method of Machine Learning, *2015 International Conference on Computing Communication Control and Automation*, Pune, India, 2015

[17] J. Zhang, M. Zulkernine and A. Haque, Random Forests Based Network Intrusion Detection Systems, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* vol. 38, no. 5, pp. 649-659, 2008

[18] Atthapol Ngaopitakkul, Chaiyan Jettanasen, Dimas Anton Asfani, Yulistya Negara Application of Discrete wavelet transform and Back-propagation Neural Network for Internal and External Fault Classification in Transformer, *International Journal of Circuits, Systems and Signal processing*, pp.458-463, Volume 13, 2019

[19] Dehong Ding, Sisi Zhu, A Method of Forest-Fire Image Recognition based on Ada Boost-BP Algorithm, *International Journal of Circuits, Systems and Signal processing,* pp.312-319, Volume 13, 2019