Frankie Fazlollahi
Martin Coder
Professor Ghosh
CSCI 183
5 June 2022

Network Intrusion Detection Classification

Introduction

Network intrusion is the unauthorized penetration of a network by a third party. Intrusions can be passive, where the intrusion is stealthy and goes undetected; or active, in which the intrusion results in effected change to assets of the domain. Intrusions can be harmless or nearly harmless, where the intruder simply leaves some indicator of their penetration; or they can be malicious, in which the intruder gains critical information from the network. Intrusions can be a one-time thing, or they can recur indefinitely until detected. Depending on the information at risk, the harm that network intrusions pose to network administrators can be immense: it is estimated that on average, a malware attack will cost a company $2.5 million, and that the global annual cost of cyber crime is $6 trillion per year. In addition to the high cost of cyber crime, it is also extremely pervasive, and attacks are only growing in frequency. There exist over 1 billion malware programs in the world, and 560,000 new malware programs are detected every day. To make things even more challenging, intrusions can also be fairly inconspicuous, as from the network's perspective a malicious connection may not look much different from a normal one. For these reasons, systems which can detect and flag intrusions in real time are extremely valuable.

Since malicious connections are not obviously identifiable, machine learning can be used to more reliably and accurately detect them. In this project, we used machine learning to detect and differentiate cyber attack threats from normal user behavior. Behavior anomaly detection techniques are essential for cyber network security. It can help notice unusual user behavior and can detect and prevent theft of data or intellectual property. Behavior anomaly detection provides real-time detection of cyber attack threats.

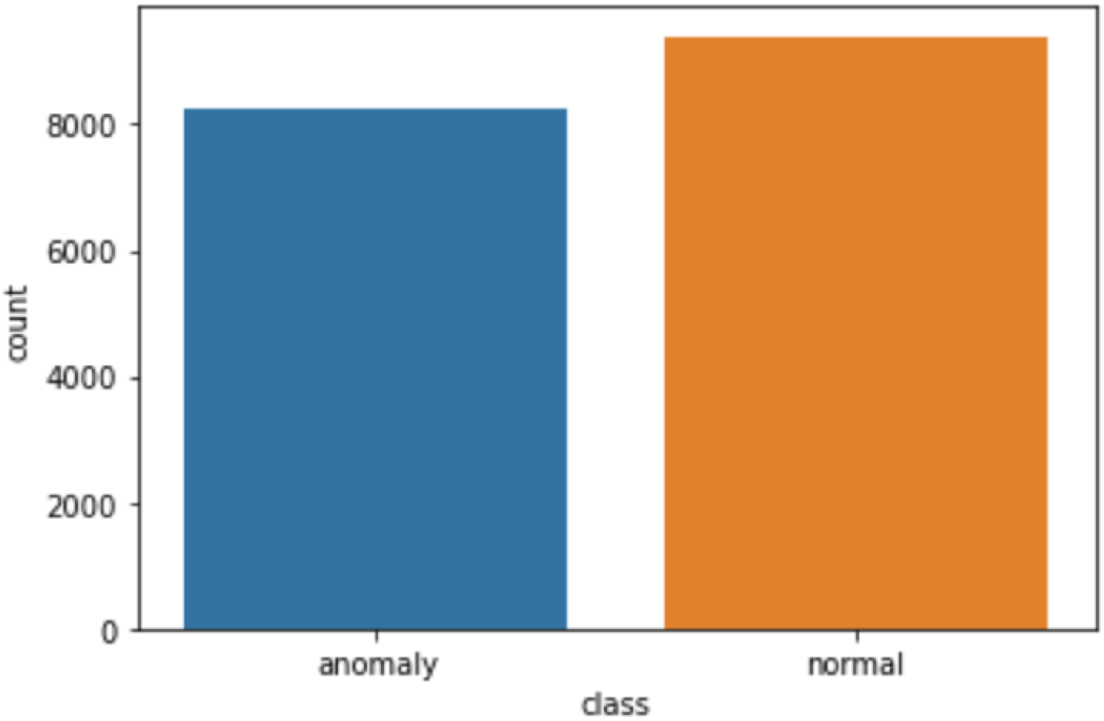
The Dataset

To detect anomalous behavior we used a dataset that labels each connection as either normal or anomalous. A connection is a sequence of TCP packets starting and ending at some time duration between which data flows to and from a source IP address to a target IP address under some well-defined protocol. The dataset consists of a wide variety of intrusions simulated in a military network environment, specifically a typical US Air Force LAN. Each connection consists of 41 qualitative and quantitative features.

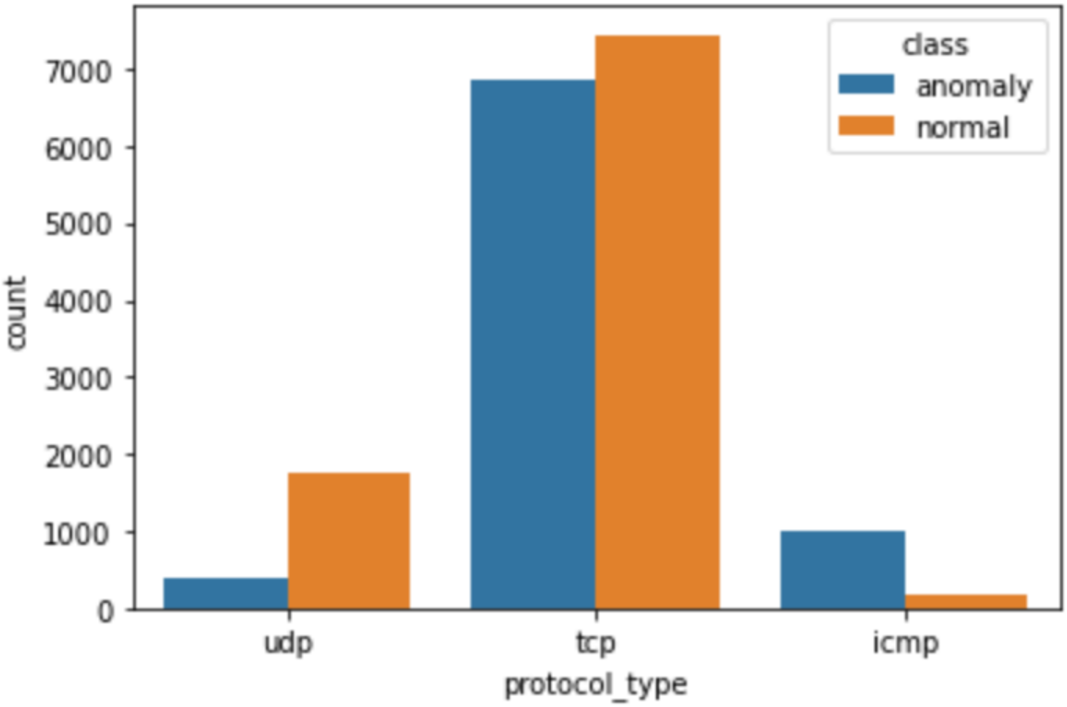
Analysis on the Dataset

When observing the values of each of the features of the dataset, we noticed that features "is_host_login" and "num_outbound_cmds" both only had one unique value, zero, across all examples. This is redundant since a feature with only one value will not affect our model. Therefore, we decided to remove these features from the dataset.

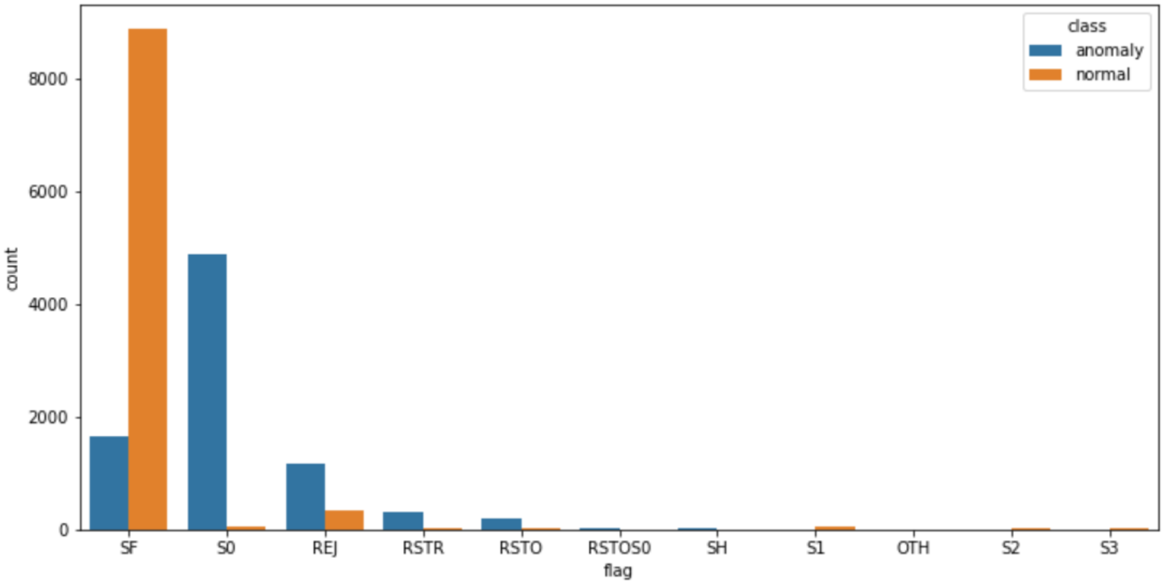
Next, we proceeded to visualize the distribution of the target class. As can be seen in the graph below, there is a slight imbalance in the target column "class" of the dataset. However, since there is not a significant difference between the amount of examples labeled "normal" and examples labeled “anomaly”, we determined that the distribution was fine and that we did not need to use oversampling or another method for handling imbalanced data.



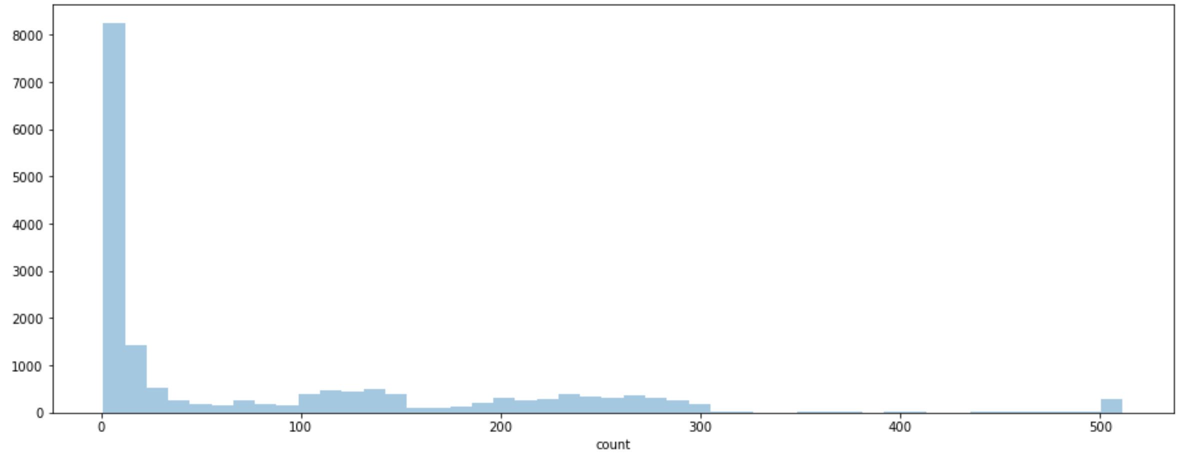
Then we visualized the distribution of the feature “protocol_type” in respect to the class to see if there are any trends from the type of protocols used and if the connection was an anomaly or not. From the data we found that 80% of the traffic was TCP, 12% was UDP, and the remaining was ICMP. Moreover, most of the ICMP traffic were anomalies, while most of the UDP traffic was normal. The distribution for TCP was about equal.

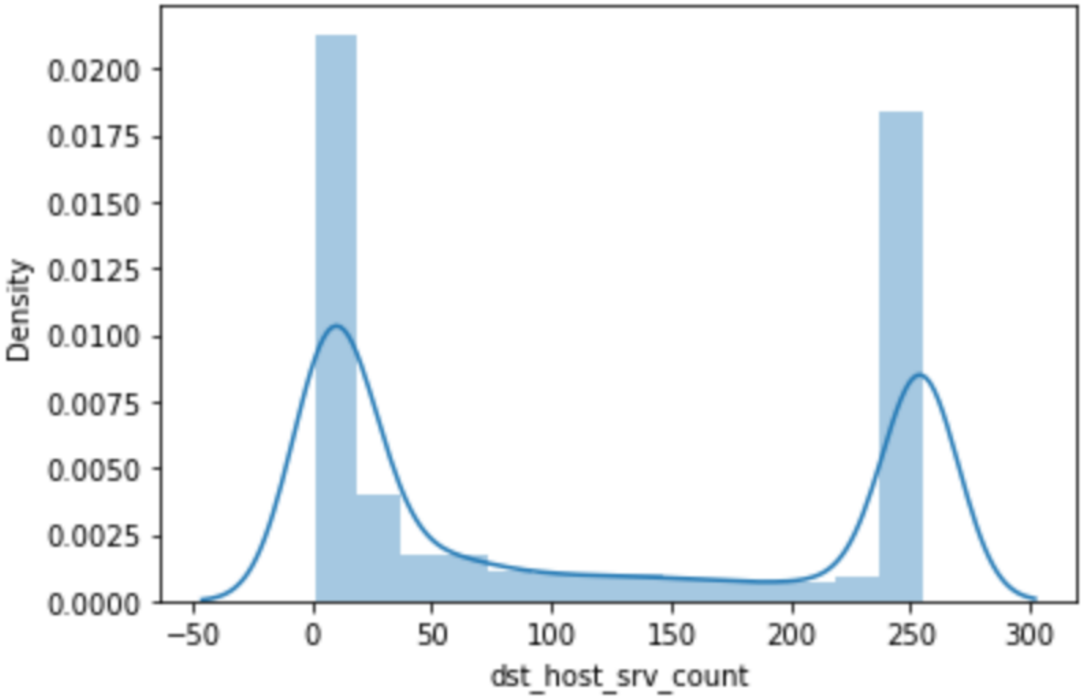


We also analyzed the traffic distribution on the basis of the "flag" feature. The distribution was uneven with most of the examples having SF (Sign Flag). Most of the traffic with SF was normal, but the traffic with the S0 flag were anomalies.

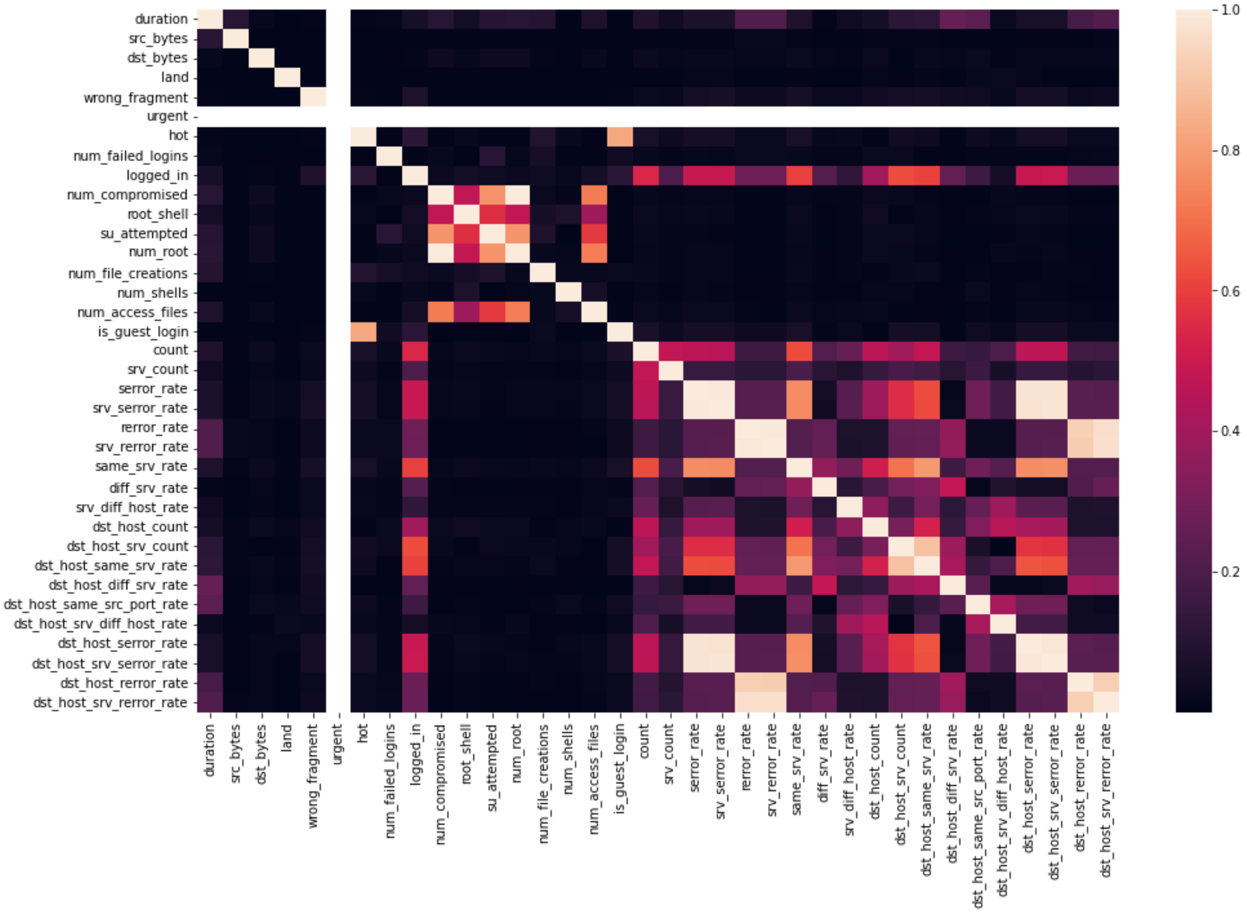


From looking at the "count" and "dst_host_srv_count" features we found that most of the traffic recorded was unique, and that the count of most of the connections having the same destination host and using the same service was either very low or very high.





After taking a look at these specific features, we wanted to see how all of the features relate to each other, if at all. So, we decided to create a correlation heatmap to show which features are related to each other and how strong the relationship is.

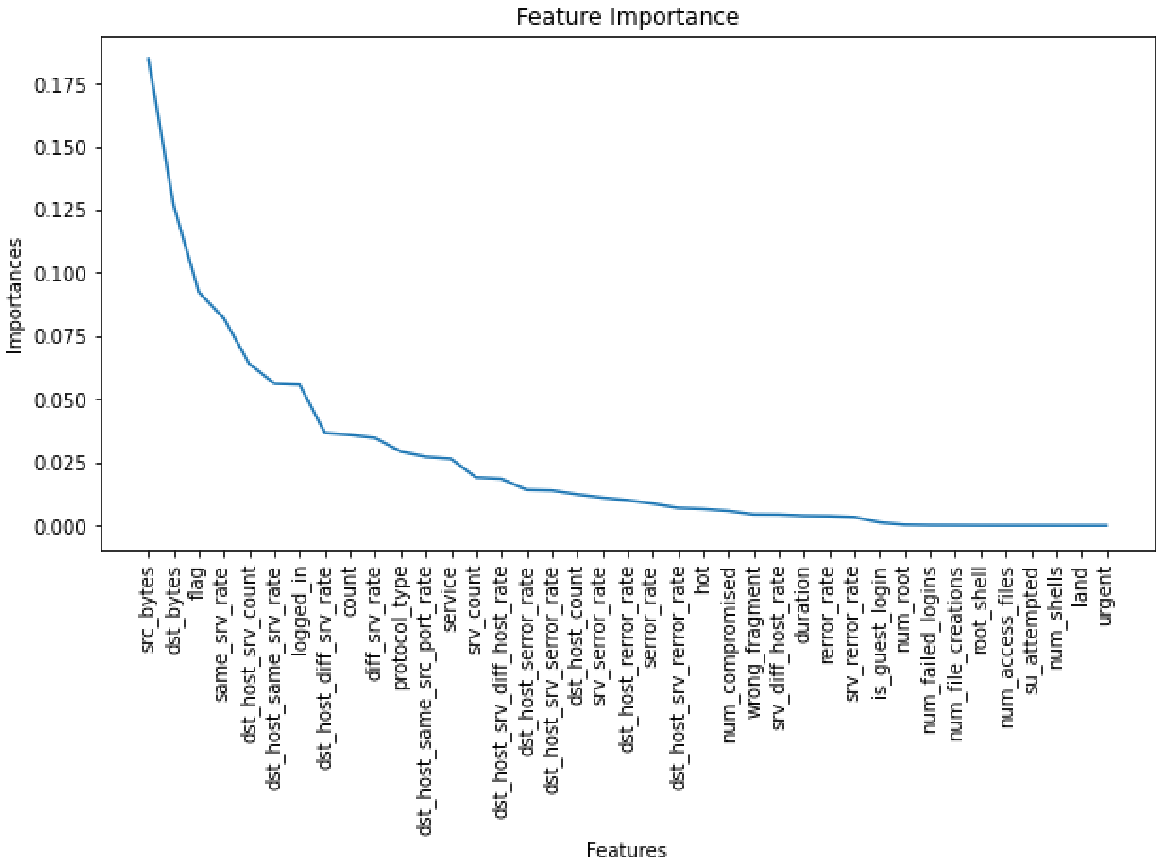


From the above correlation heatmap we can see that most of the data has very low correlation. This is a good characteristic for our machine learning process since a group of highly correlated features will not provide much additional information, but will increase the complexity of the algorithm, increasing the potential for errors.

Preprocessing the Data

After understanding our data we proceeded to do some preprocessing. First we split the training data into 70% for training and 30% for testing our model. Secondly, we encoded the features that had values that were not numeric. Thirdly, we normalized the data because we did not want to risk any of our machine learning estimators behaving poorly since the individual features did not look like standardly normally distributed data.

Next, we wanted to get the importance of features. To do this we used sklearn’s RandomForestClassifier and sorted the features by their importance.

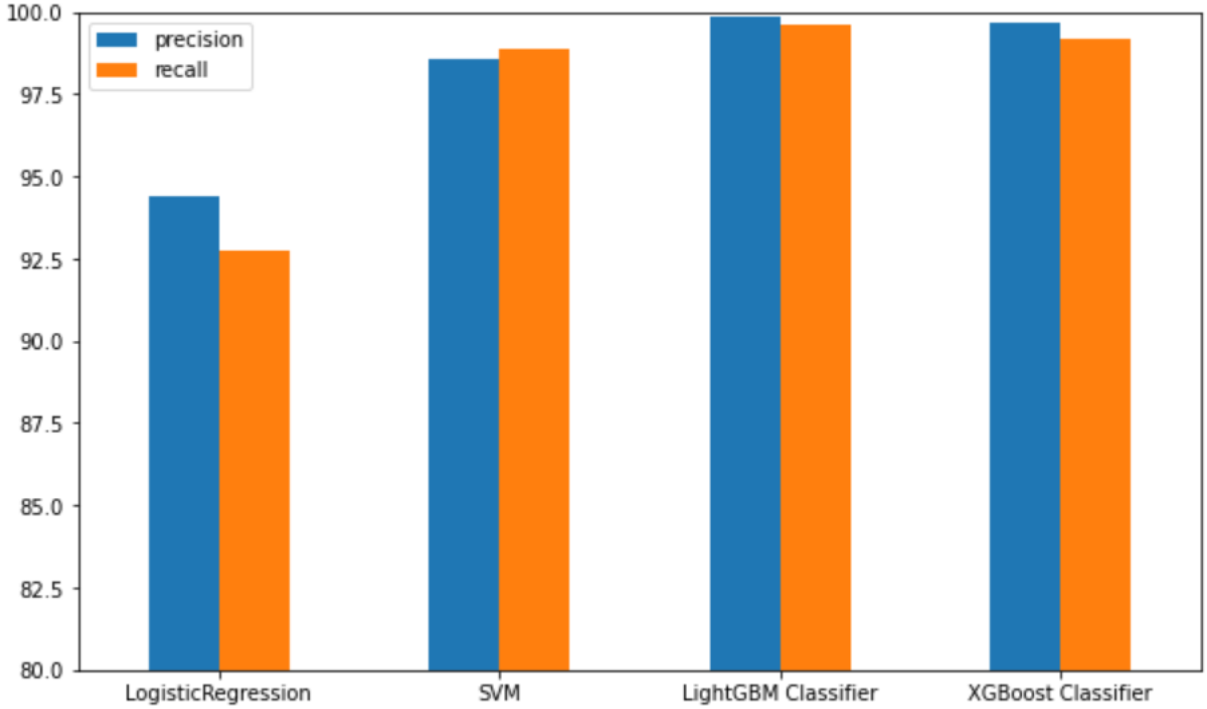


We then used sklearn's RFE (Recursive Feature Elimination) for feature selection.

Trying Various Models

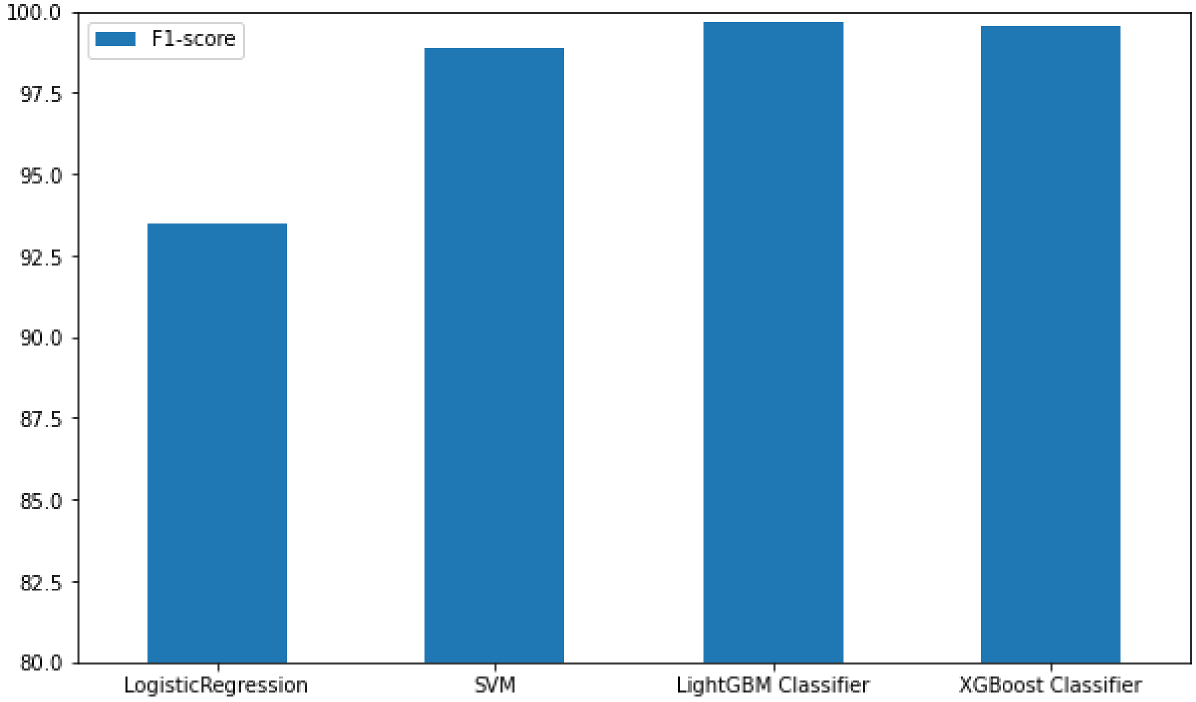
For classification we wanted to use some models that we covered in class. So two of the models we used were logistic regression and support vector machine (svm). We also wanted to try some models that are popular today and commonly used in practice, so we decided to also try LightGBM (Light Gradient Boosting Machine), which is a gradient boosting framework that uses tree based learning algorithms developed by Microsoft, and XGBoost (Extreme Gradient Boosting), which is a scalable, distributed gradient-boosting decision tree algorithm.

We used 10-fold cross validation on the training data for each of the classification models above. Below are the mean precision and recall results for the various models.



Recall is the more important measure to consider since it is significantly more harmful to falsely classify an intruder as a normal user than to classify a normal user as an intruder. As we can see, all the models performed well on the training data, with LightGBM and XGBoost performing the best.

We then trained the models on 70% of the training data and predicted on the remaining 30% to evaluate their performance.



Above are the F1-scores of each of the models based on the prediction and true values of 30% of the training data. Each of the models performed very well on the subset of the data we used for testing. Logistic regression performed the worst of the models with still a strong F1-score of about 93. SVM, LightGBM, and XGBoost all performed extremely well with LightGBM performing the best, slightly over XGBoost, and SVM not far behind.

Results

Overall, we are very satisfied with the results from each of our models. None of the models overfitted on the training data as they each performed better on the subset we used for testing than they did for training. LightGBM performed the best according to all of our performance measurements, so we decided to use it to predict on the real test dataset that does not contain labels.

Works Cited

Dataprot. “A Not-So-Common Cold: Malware Statistics in 2022.” Dataprot, 9 May 2022, dataprot.net/statistics/malware-statistics.

PurpleSec. “2022 Cyber Security Statistics: The Ultimate List Of Stats, Data and Trends.”

R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," 2010 IEEE Symposium on Security and Privacy, 2010, pp. 305-316, doi: 10.1109/SP.2010.25.

H. Jiang, Z. He, G. Ye and H. Zhang, "Network Intrusion Detection Based on PSO-Xgboost Model," in IEEE Access, vol. 8, pp. 58392-58401, 2020, doi: 10.1109/ACCESS.2020.2982418.