



DDoS Attack Identification

Machine Learning - Course Project

Divyam Choudhary
Prince Butani

(MS2022011)
(MS2022006)



What is DDoS Attack?

- DDoS (Distributed Denial of Service) is a category of malicious cyber-attacks that hackers or cyber criminals employ in order to make an online service, network resource or host machine unavailable to its intended users on the Internet.
- Targets of DDoS attacks are flooded with thousands or millions of superfluous requests, overwhelming the machine and its supporting resources.



Data Pre-processing

- The data is a big data with around 8.9 GB of size. Such a huge data is difficult to process and also lot of data might be not useful.
- Since the data set is very huge and to reduce the size of it, all the columns were analyzed and the data types were changed from float64 to float32 or int32, whichever was applicable.
- Many of the columns has infinite values, negative values and zero values.



Data Pre-processing

- The column features with more than 95% of the values as zero are dropped.
- Many of the columns have infinite values and thus those values are replaced with zero.
- Duplicate values were checked and were removed.

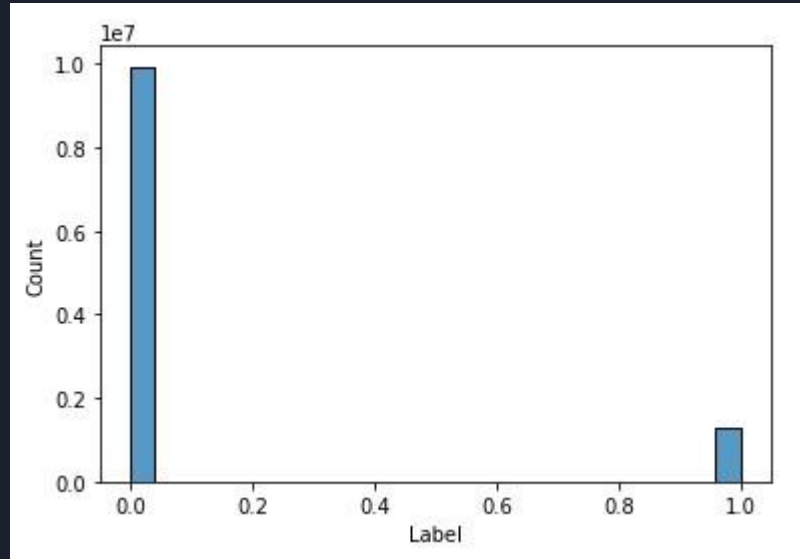


Box plot of 'Fwd Seg Size Min' showing the data distribution



Data Pre-processing

- Box plots were made to study deviation values of the columns
- The columns with lot of deviation values are:
 1. Init Fwd Win Byts
 2. Subflow Bwd Byts
 3. Subflow Bwd Pkts
 4. Subflow Fwd Byts
 5. Init Bwd Win Byts
 6. Fwd Act Data Pkts
 7. Active Mean
- The data was hence reduced from 8.9 GB to 4.08 GB after all these pre-processing



Histogram plotted using the seaborn library, depicting the imbalanced nature of the data set



Models

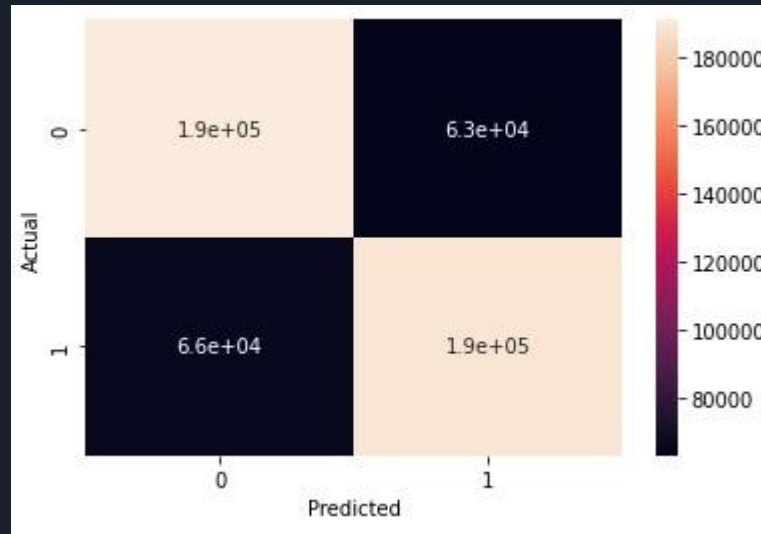
Different Machine learning models were implemented to train and test the data set and measure the performance of each model:

1. Logistic Regression Model
2. Random Forest Model
3. XGBoost Model



Logistic Regression

- The data had 11.36% of Malicious values and 88.64% Benign values
- Random under-sampling was carried out such that, the majority class instances are discarded at random until a more balanced distribution is reached.
- The data was then divided into train and test data in the ratio of 4:1 with a test size of 0.2
- After fitting the model, it didn't converge with the max iteration of 1000 and was increased to 10,000
- The accuracy, as well as the F1 score obtained here, is 74%.



Confusion Matrix showing the actual and predicted values



Random Forest Classifier

Model Name	Model Details	Dataset Details	Validation Acc	Submission Score
RF-50-5.0	1018605 data points with 3-fold Grid Search cross validation, comparing estimator sizes of 50 and 100.	Inf cols dropped	0.95	0.76879
RF-100-5.1		Inf cols dropped + zero cols removed	0.95021	0.76692
RF-100-5.2		Inf cols dropped + minmax normalization	0.955002268556997	0.34542
RF-100-6.0		Inf cols replaced with col max	0.9499187024309469	0.76644



Comments on RF models uptill now

- We get good accuracy on validation set in Grid Search.
- Even the validation set of 254651 data points give accuracy above 90% for all models.
- Issue is with test data submissions, the scores are low.
- Minmax normalization doesn't perform well on RF as it is not needed for categorical data if any.
- The some columns in our dataset may be categorical but in numerical format. Normal RF handles it much better than minmaxed.

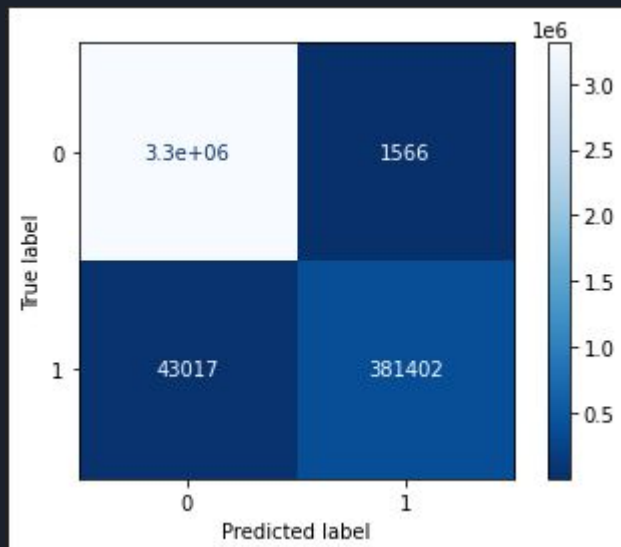



XGboost Models

Model Name	Model Details	Dataset Details	Validation Acc	Submission Score
RF-50-5.0	1018605 data points with 3-fold Grid Search cross validation, comparing estimator sizes of 50 and 100.	Inf rows dropped	0.95	0.76879
XGB-MD6-1.0	Randomly Under Sampled dataset shape Counter({0: 1273257, 1: 1273257}), Default hyper parameters		0.943	0.84823
XGB-7.1-v1	Stratified k-fold validation, max_depth=3, rest hyper parameters default		0.944	0.94284
XGB-8.0-v1	Stratified k-fold validation, max_depth=3, rest hyper parameters default	Inf rows dropped. 49 columns. All parallel columns removed.	0.944	0.90415

XGB-7.1-v1

- Mean Accuracy: 94%



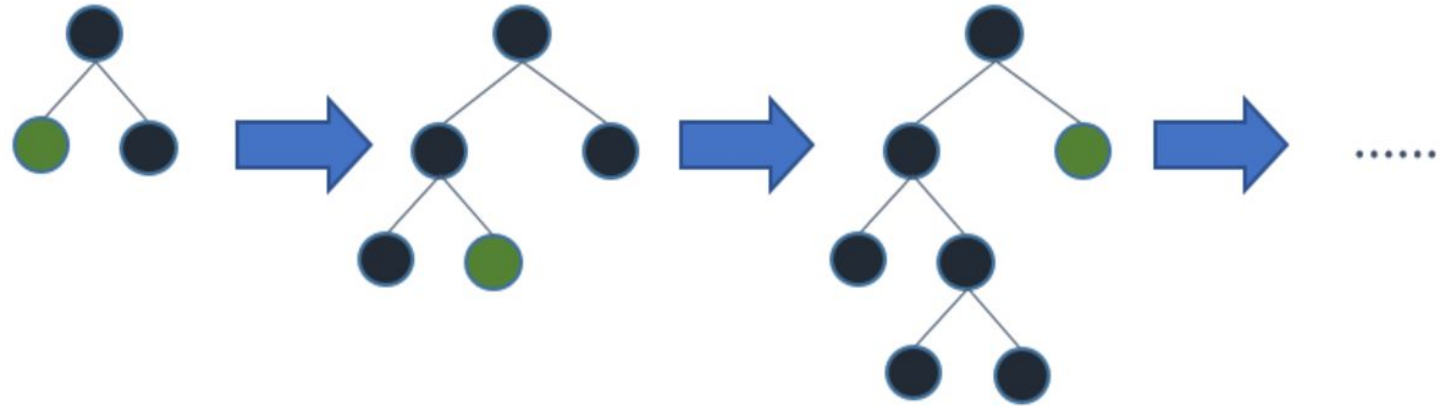
- 
- Gradient Boosting Decision Tree (GBDT) is a popular machine learning algorithm. It has quite effective implementations such as XGBoost as many optimization techniques are adopted from this algorithm
 - However, the efficiency and scalability are still unsatisfactory when there are more features in the data.
 - For this behaviour, the major reason is that each feature needs to scan all the data instances to estimate the information gain of all possible split points, which is very time-consuming.



LightGBM

- To tackle this problem, the LightGBM (Gradient Boosting Machine) uses two techniques, namely Gradient-Based One-Side sampling (GOSS) and Exclusive Feature Bundling (EFB).
- So GOSS excludes the significant portion of data instances with small gradients and only uses the remaining data to estimate the information gain.
- Since the data instances with large gradients play a more important role in the computation of information gain, GOSS can obtain quite accurate information gain with a relatively much smaller dataset.

LightGBM grows the tree vertically while another tree-based learning algorithm grows horizontally; LightGBM grows trees leaf-wise, and it chooses max delta loss to grow

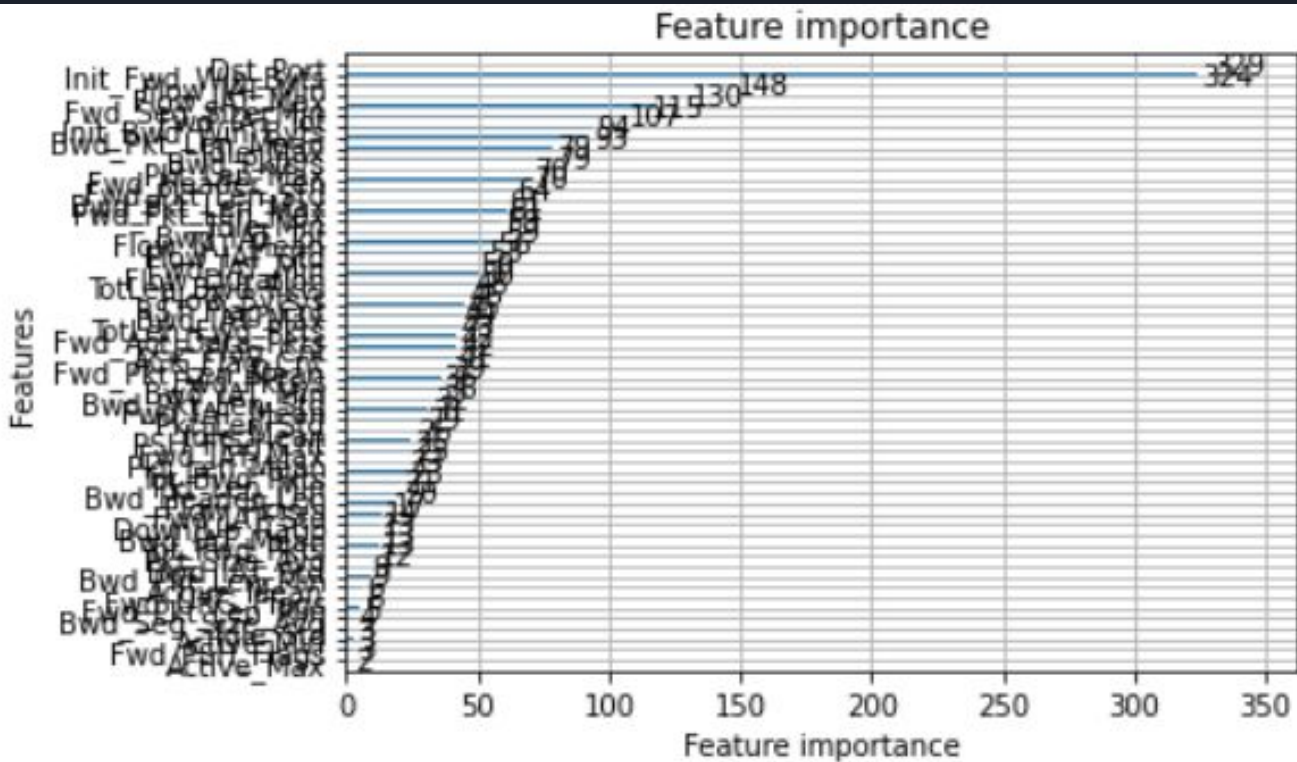


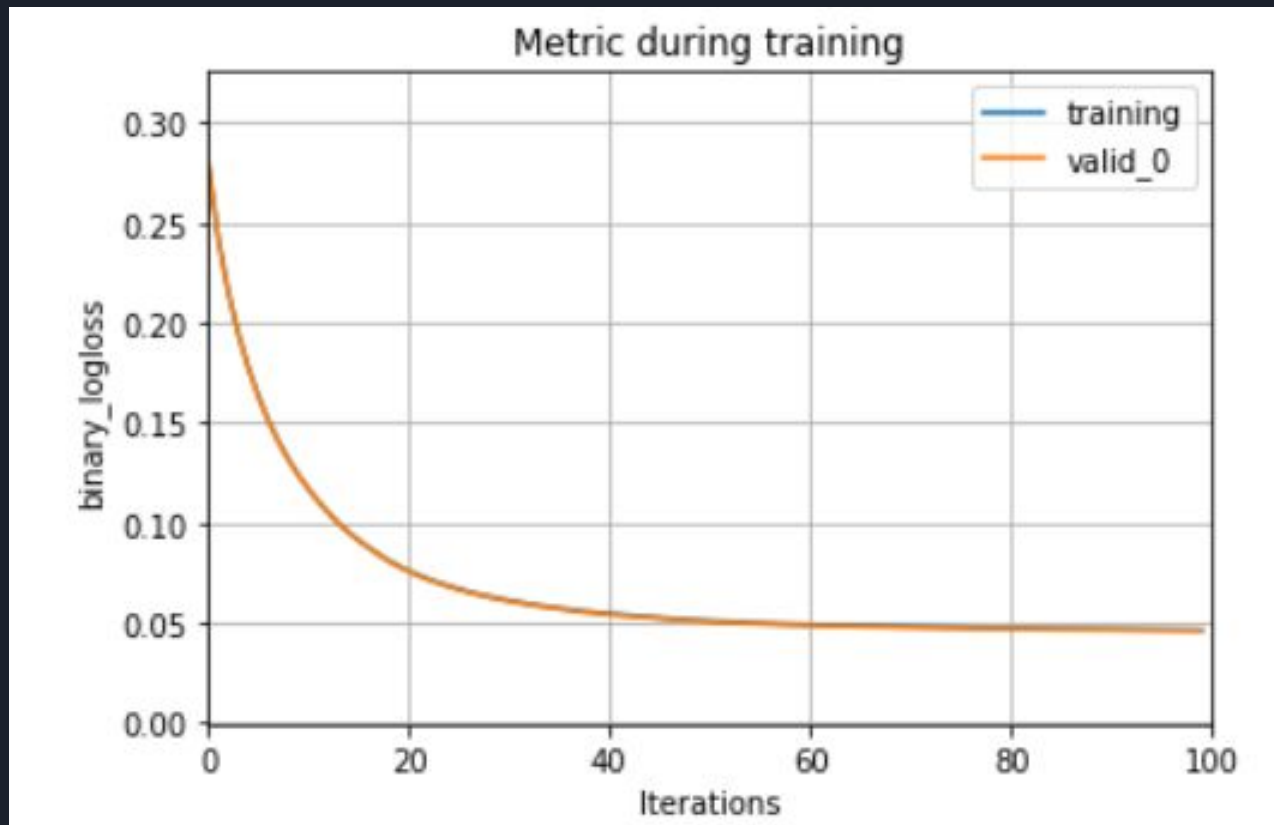
Leaf-wise tree growth

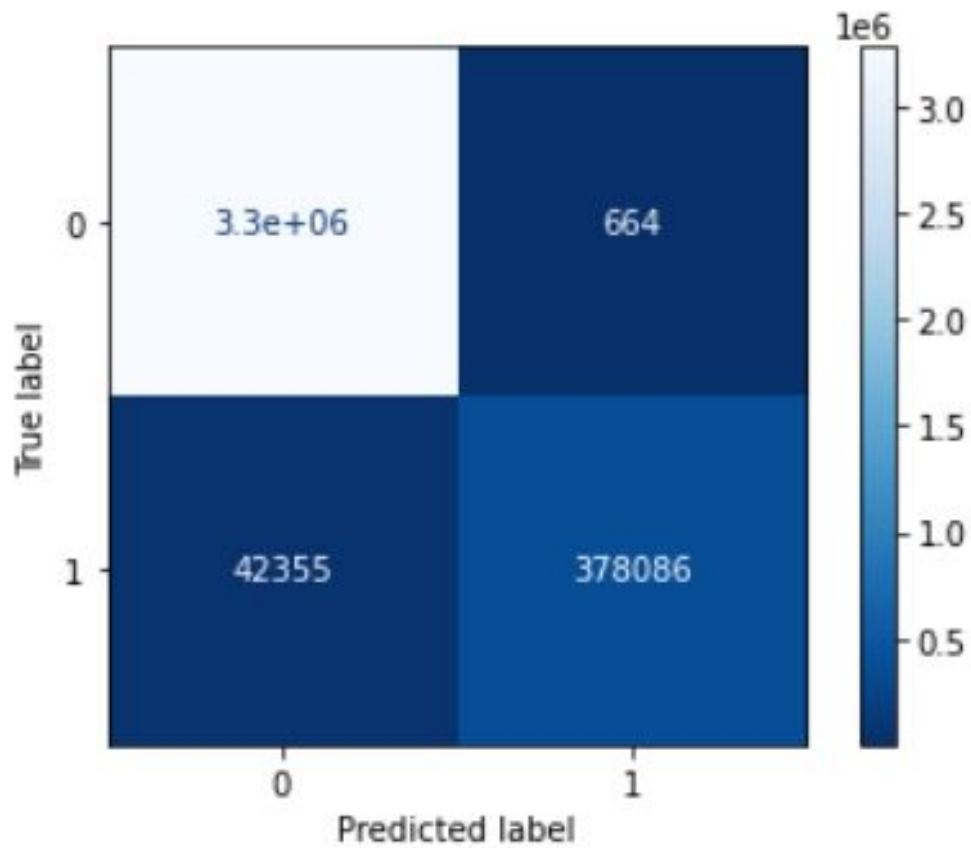



Advantages of LightGBM

- Faster training speed with higher accuracy
- Lower memory usage
- Better accuracy than any other boosting algorithm specially handles the overfitting very well when working with a small dataset
- Compatibility with large datasets
- Parallel learning support.









	precision	recall	f1-score
0	0.99	1.00	0.99
1	1.00	0.90	0.95
accuracy			0.99
macro avg	0.99	0.95	0.97
weighted avg	0.99	0.99	0.99

The Accuracy after submitting to kaggle is 90.115 %