COMS 4995 - Applied Machine Learning

Project Deliverable #1 - Proposal

Background and context

A computer network is protected from unauthorized users, including possibly insiders, using software that detects network intrusions. The goal of the intrusion detector learning job is to create a predictive model, or classifier, that can differentiate between good, typical connections and undesirable connections, sometimes known as intrusions or attacks.

Data flows to and from a source IP address to a target IP address under a well-defined protocol using a series of TCP packets that begin and stop at specific times. Every connection has a name that indicates whether it is normal or an attack, with a single attack type. The size of each connection record is around 100 bytes.

There are four basic types of attacks:

- R2L: unauthorised access from a distant computer, such as password guessing;
- DOS: denial-of-service, such as syn flood;
- U2R: unauthorized access to local superuser (root) privileges, e.g., various "buffer overflow" attacks;
- probing: surveillance and other probing, e.g., port scanning.

Dataset

NSL-KDD
https://www.kaggle.com/code/linchihyi/nsl-kdd/input

Dataset Information

KDDTrain+.ARFF The full NSL-KDD train set with binary labels in ARFF format

KDDTrain+.TXT The full NSL-KDD train set including attack-type labels and difficulty level in CSV format

The full NSL-KDD train set including attack-type labels and difficulty level in CSV format

KDDTrain+_20Percent.ARFF A 20% subset of the KDDTrain+.arff file

KDDTrain+_20Percent.TXT A 20% subset of the KDDTrain+.txt file

KDDTest+.ARFF The full NSL-KDD test set with binary labels in ARFF format

KDDTest+.TXT The full NSL-KDD test set including attack-type labels and difficulty level in CSV format

KDDTest-21.ARFF A subset of the KDDTest+.arff file which does not include records with difficulty level of
21 out of 21

KDDTest-21.TXT A subset of the KDDTest+.txt file which does not include records with difficulty level of 21
out of 21

Improvements to the KDD'99 data set

The NSL-KDD data set has the following advantages over the original KDD data set:
It does not include redundant records in the train set, so the classifiers will not be biased towards more frequent records.

There are no duplicate records in the proposed test sets; therefore, the performance of the learners are not biased by the methods which have better detection rates on the frequent records.

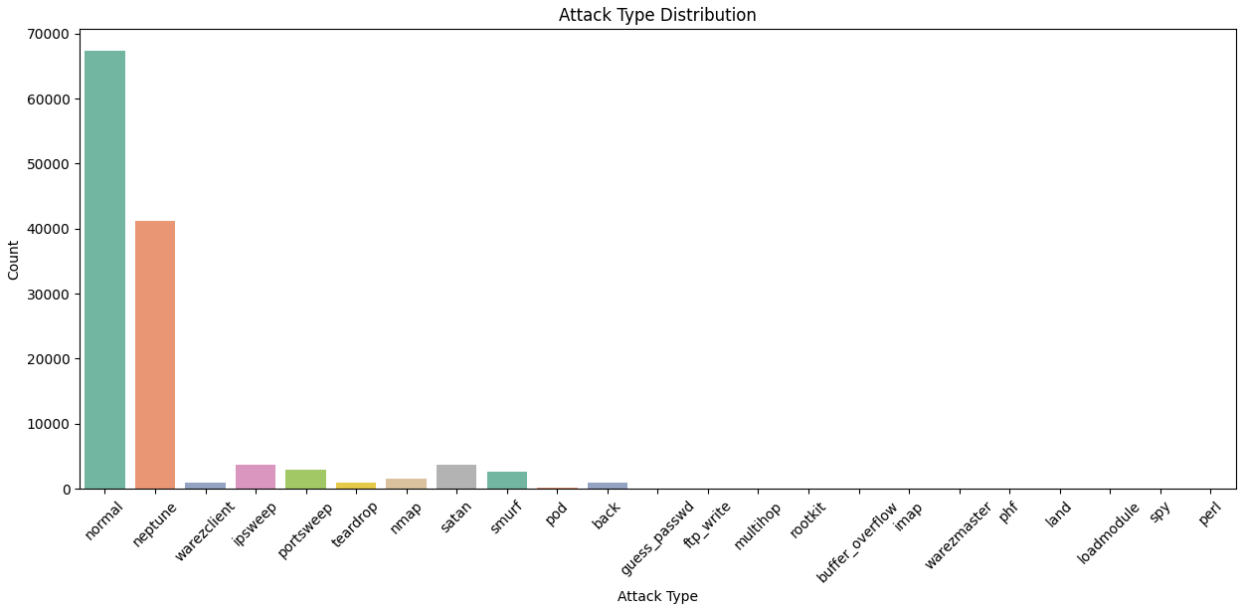
The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set. As a result, the classification rates of distinct machine learning methods vary in a wider range, which makes it more efficient to have an accurate evaluation of different learning techniques.

The number of records in the train and test sets are reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research works will be consistent and comparable.

Dataset Information

The dataset contains 125972 samples with 42 features. Among the 42 features, we will use the feature ‘`attack`’ as our target variable.

The distribution of our target variable is as follows:



Machine Learning techniques

[Data engineering techniques]

- SMOTE (To deal with class imbalance)
- Feature engineering (Adding new features through making combinations of the existing features)
- Feature Selection and Dimensionality Reduction techniques.

Since our target variable is highly imbalanced, we intend to use the oversampling technique ‘SMOTE’ to overcome the class imbalance problem. Also, considering the semantics of each predictor variable, we are going to try making new features by using combinations of the existing features to create features that are more correlated with the target variable. Also, since we are planning to do feature engineering and the initial number of predictor variables is 42 which is quite a lot, we also intend to try reducing the number of predictor variables through feature selection or dimensionality reduction techniques.

[Machine Learning Models]

- Tree-based models (Random Forest, XGBoost, Catboost, Explainable boosting machine)
- Support Vector Machines (SVM) with kernels

- Deep Learning-based models

We intend to try out machine learning models that are listed above. As for the tabular data, tree-based models are known to work better than the deep learning-based models. So, testing out the performance of tree-based models is our first priority. Then, rather than using the naive Support Vector Machines (SVM), we use SVMs with non-linear kernels so that we achieve better performance. Then, we intend to try out deep learning-based models, which are known to map complex non-linear relationships well, on our dataset. For all models, we intend to finetune the models with sophisticated changes on the hyperparameters so as to achieve the best performance.