## Group Memebers: Congyou Yangxie, Yuannan Yang, Siqi Cao

## **1. Problem definition: What are you trying to do? Where do you get data? (2 pt.)**

**Problem definition:**

Our goal is using a dataset to train a model and the model is capable of distinguishing between abnormal connection (intrusions or attacks) and normal connection, to predict incoming packets’ label (Similar to an Intrusion Detection). We get the data from the NSL-KDD dataset (<https://www.kaggle.com/hassan06/nslkdd>).

## 

## **2. A brief introduction on the data and a preliminary study about your data characteristic. (4 pt.)**

**Introduction to dataset:**

The NSL-KDD dataset which is an improved version of the previous KDD 1999 Dataset. This dataset contains four sub datasets(KDDTest+, KDDTest-21, KDDTrain+, KDDTrain+\_20Percent). The KDDTest-21 (without the most difficult traffic records) and KDDTrain+\_20Percent(20% of entire train dataset) are subsets of the KDDTrain+ and KDDTest+. There are 148517 records of internet traffic of simple intrusion detection in the entire dataset, which contain 43 features per record. 41 features are for traffic input and the last two are labels (normal or attack) and Score (the severity of the traffic input).

Four different classes of attacks exist in the dataset: DoS (Denial of Service), Prob, U2R (User to Root), R2L (Remote to Local). The features types in this dataset can be broken down into 4 types: 4 Categorical, 6 Binary, 23 Discrete, 10 Continuous.

This dataset is labeled and straightforward and has several advantages: The number of selected records from each difficulty level group is inversely proportional to the percentage of records. 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. Besides, 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.

## 

## **3. What models/techniques do you plan to use? (4 pt.)**

**Data preprocessing:**

All features are made numerical using One-Hot-Encoding. The features are scaled to avoid features with large values that may weigh too much in the results. One-Hot-Encoding can transform all categorical features into binary features. We plan to use the LabelEncoder and OneHotEncoder from sklearn.preprocessing to transform features. According to .arff files, the test dataset has fewer features than the train dataset. We have to add these missing features to the test dataset before training. The next step is replacing the label column with numbers (0 for normal, 2 for probe, 3 for R2L, 4 for U2R).

The last step is to standardize the features by removing the mean and scaling to unit variance, in order to make sure that some features can be compared to other features. Plan to use the StandardScaler() from sklearn.preprocessing.

**Feature Selection:**

In this part, we’re going to focus on eliminating redundant and irrelevant data by selecting a subset of relevant features that fully represents the given problem. Plan to separately use PCA (Principal Component Analysis), SVD (Singular Value Decomposition), LDA(Linear Discriminant Analysis) ANOVA F-test, and RFE (Recursive Feature Elimination) to select features. Then test each method with our model planned to figure out the best one.

1. PCA: sklearn.decomposition.PCA
2. SVD: sklearn.decomposition.TruncatedSVD.
3. LDA: sklearn.discriminant\_analysis.LinearDiscriminantAnalysis
4. ANOVA F-test: sklearn.feature\_selection.f\_classif. ANOVA F-test analyzes each feature individually to determine the strength of the relationship between the feature and labels. Using the SecondPercentile method to select features based on percentile of the highest scores.
5. RFE: sklearn.feature\_selection.RFE. The goal of RFE is to select features by recursively considering smaller and smaller sets of features.

**Model Selection:**

Supervised learning is the strategy we want to use in this part. Decision Tree, Logistic Regression and SVM (Support vector machines) are the candidates.

1. Decision Tree: sklearn.tree.DecisionTreeClassifier.
2. Logistic Regression: sklearn.linear\_model.LogisticRegression. In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the ‘multi\_class’ option is set to ‘ovr’, and uses the cross-entropy loss if the ‘multi\_class’ option is set to ‘multinomial’
3. SVM: sklearn.svm.LinearSVC. This class supports both dense and sparse input and the multiclass support is handled according to a one-vs-the-rest scheme.

**Prediction & Evaluation (validation):**

Using the test data to make predictions of the model. Multiple scores are considered such as: accuracy score, recall, f-measure, confusion matrix.

1. Accuracy. sklearn.metrics.accuracy\_score. To compute the test dataset’s accuracy, comparing the predicted labels with actual labels in test datasets.
2. ROC (receiver operating characteristic) curves and AUC: sklearn.metrics.roc\_auc\_score. This class computes the area under the ROC curve, to reflect the relationship between true positive rate and false positive rate.
3. PRC (Receiver Operating Characteristic): sklearn.metrics.f1\_score. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0.

## **4. Need to include at least 2 factors to compare…**

We are going to compare Feature selection and Model selection.

Comparison of feature selection: we would compare the multiple results of evaluation between a model with no feature selection and a model with feature selection.

Comparison of model selection: we would compare the multiple results of evaluation using different feature selection models.

## **5. Reference**

<https://towardsdatascience.com/a-deeper-dive-into-the-nsl-kdd-data-set-15c753364657>

[https://github.com/CynthiaKoopman/Network-Intrusion-Detection/](https://github.com/CynthiaKoopman/Network-Intrusion-Detection/blob/master/DecisionTree_IDS.ipynb)

<https://www.kaggle.com/hassan06/nslkdd>