Mini-Project 3: Network Intrusion Detection

Due Date: 4 pm, Monday, October 22, 2018

Team Members:
Name: Zainiya Manjiyani
Student ID: 219216284

Problem Statement:

Software to detect network intrusions protects a computer network from unauthorized users, including perhaps insiders. The project aims is to build a network intrusion detector, a predictive model capable of distinguishing between bad connections, called intrusions or attacks, and good normal connections

Task Division:

- Model the problem as a BINARY classification problem.
- Compared the accuracy, recall, precision and F1-score for all the models.
- Printed and plot the Confusion matrix as well as ROC curve for each model.
- Did additional features

Models used for comparision:

- Logistic Regression
- Nearest Neighbor
- Support Vector Machine
- Gaussian Naive Bayes
- Fully-Connected Neural Networks
- Convolutional Neural Networks (CNN)

Methodology:

- Prepared data for binary classification problem
- Encoded ctegorical features and Normalized numeric features
- Used different models for comparing accuracy, recall, precision and F1-score
- Printed and plotted Confusion matrix as well as ROC curve for each model
- For tuning neural networks and CNN(Used Conv1D) performance used:

  Activation: relu, sigmoid, tanh
  Layers and neuron counts
  Optimizer: adam, sgd, rmsprop, and others
  Kernel number and kernel size (for CNN only)

Experimental Results and Analysis:

Experiments and results on Neural Network:

Activation	**Optimizer**	**Hidden Layer 1**	**Hidden Layer 2**	**Hidden Layer 3**	**Hidden Layer 4**	**Precision**	**recall**	**f1-score**	**Accuracy**
relu	adam	20	20			1	1	1	99.89
relu	adam	100	150	60	30	1	1	1	99.90
tanh	sgd	100	150	60	30	1	1	1	99.86
relu	sgd	75	25			1	1	1	99.91
relu	adam	75	25			1	1	0.99	99.94

All the combination was giving approximately 99% so nothing can be said deterministic. But best result found in combination of relu and adam with values on hidden layers 75 and 25 giving accuracy of 99.93

Experiments and result on CNN:
Activation and Optimizer used: relu and adam

Kernel Size Layer1	Kernel Size Layer 2	Kernel Size Layer 3	precision	recall	f1-score	accuracy
30	30		0.99	0.99	0.99	0.999
50	25		0.99	0.99	0.99	0.990
5	5		1	1	1	0.996
5	10	20	0.99	0.99	0.99	0.998
5	5	5	0.99	0.99	0.99	0.992

All the combination gives approximately 99% so nothing can be said deterministic. But best result found with the kernel size (layers 1, 2, 3) with values of 5, 10 and 20 with accuracy of 99.8
So moving forward used the same kernel size for different combination of activation and optimizer.

Experimenting on CNN by changing activation and optimizer
kernel size on layers 1: 5
kernel size on layers 2: 10
kernel size on layers 3: 20

Activation	Optimizer	precision	recall	f1-score	accuracy
relu	adam	0.99	0.99	0.99	0.994
relu	sgd	0.68	0.82	0.74	0.822
relu	rmsprop	1	1	1	0.996
sigmoid	adam	1	1	1	0.998
sigmoid	sgd	0.65	0.80	0.72	0.8041

sigmoid	rmsprop	1	1	1	0.9959
tanh	adam	1	1	1	0.9986
tanh	sgd	1	1	1	0.9964
tanh	rmsprop	1	1	1	0.9989

Comparision of precision, recall , f1-score and accuracy between different models

Model	**Precision**	**Recall**	**F1-Score**	**Accuracy**
Logistic Regression	0.9972	0.9974	0.9978	0.9972
Nearest Neighbor	0.9995	0.9996	0.9997	0.9995
SVM	0.9980	0.9978	0.9985	0.9980
Gaussian Naive Bayes	0.9492	0.9372	0.9674	
Fully Connected Neural Network	0.9992	0.9994	0.9995	0.9992
CNN	0.9942	0.9963	0.9955	0.994

Additional Features:

I have understood the concepts of feature selection from:
https://machinelearningmastery.com/feature-selection-machine-learning-python/

I have used feature selection for the given dataset that contribute most to the prediction variable or output in which I was interested.

Some important points that I learned from the link above:

Having irrelevant features in data can decrease the accuracy of many models.

Three benefits of performing feature selection before modeling data are:

- **Reduces Overfitting**: Less redundant data means less opportunity to make decisions based on noise.
- **Improves Accuracy**: Less misleading data means modeling accuracy improves.
- **Reduces Training Time**: Less data means that algorithms train faster.

In this project I used 3 feature selection methods:

1. Univariate Selection

Statistical tests can be used to select those features that have the strongest relationship with the output variable.

2. Recursive Feature Elimination

The Recursive Feature Elimination (or RFE) works by recursively removing attributes and building a model on those attributes that remain.

It uses the model accuracy to identify which attributes (and combination of attributes) contribute the most to predicting the target attribute.

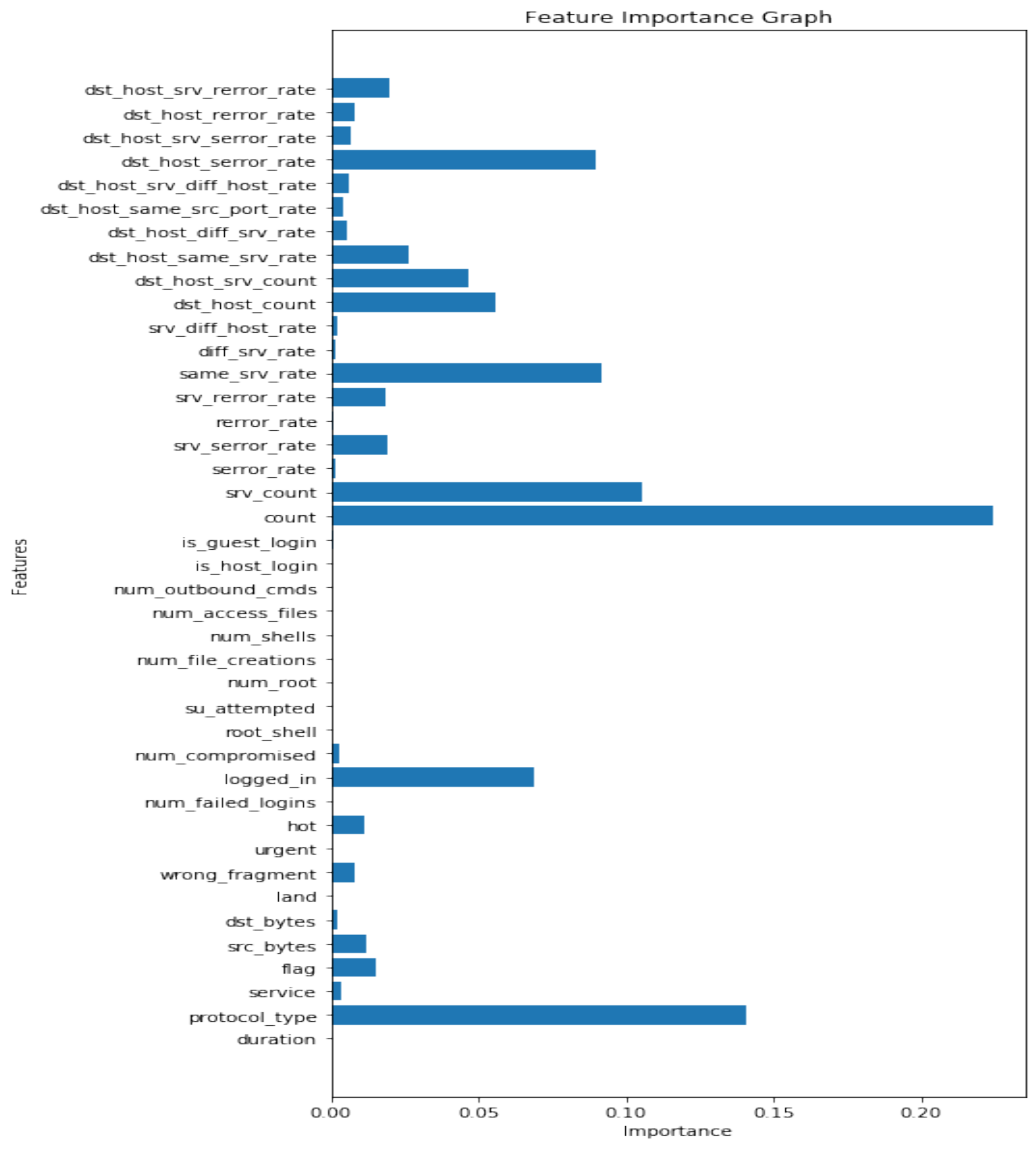
3. Feature Importance

Bagged decision trees like Random Forest and Extra Trees can be used to estimate the importance of features. In my project I have used ExtraTreesClassifier class to find the importance of features.

Following are the top 5 important features according to feature importance:

1. count
2. protocol_type
3. srv_count

4. same_srv_count
5. dst_host_serror_rate



After doing Feature Selection I removed all the columns with importance score of 0 and again find accuracy by applying Logistic regression:
Before removing accuracy score: 0.9973
After removing columns got the accuracy score : 0.9947