**SOFTWARE-DEFINED NETWORK**

**Software-defined network:**

Software-defined network (SDN) is basically a network model that allows programmatic administration and control, along with network resource optimization, simply by applying open APIs. As SDN decouples network configuration and traffic engineering from their critical hardware infrastructure, this network control is established.

**Random Forest:**

As discussed above, we will be building the model using Random Forest Classifier. Random Forest Classifier is composed of various numbers of decision trees that acts as a compound Machine Learning algorithm in order to get a much better diagnostic performance in contrast to using one decision tree algorithm.

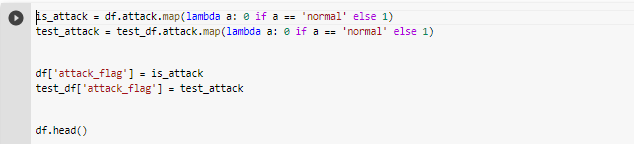
**Application of Random Forest on given model:**

On execution, multiple decision trees are generated,each formed on an individual subtype of the dataset, and the model’s final outcome is acquired by totalizing the results or output of all the trees.

**SDN Methodology:**

* The SDN is implemented using a machine learning algorithm called Random Forest.
* The networking datasets are extracted from [kaggle](https://www.kaggle.com/).com.
* The training dataset undergoes into a RandomForestClassifier() function.
* A testing dataset is then compared with the training ones to predict the network attacks.
* As the model is implemented, a confusion matrix of actual attacks along with predicted ones is generated as output for all tested dataset.

**INPUT SCREENSHOTS:**

****

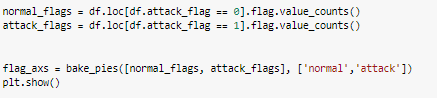
Input

****

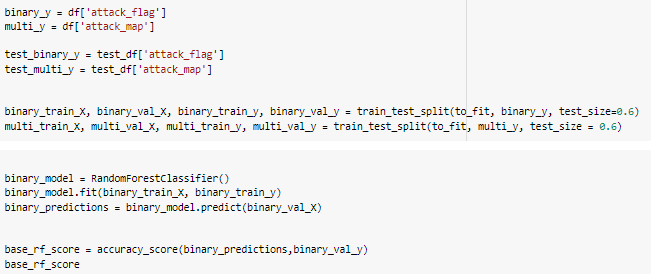
Input

****

Input 3

****

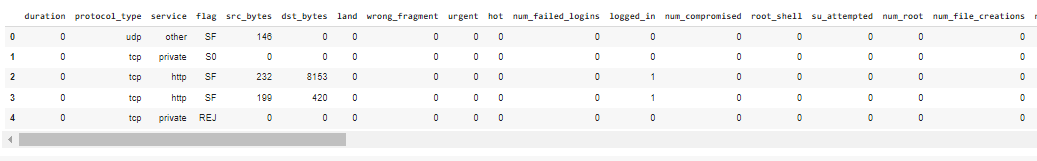
Input 4

****

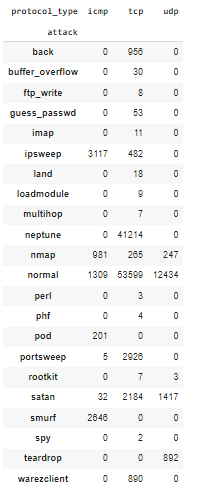
****Input 5

Input 6

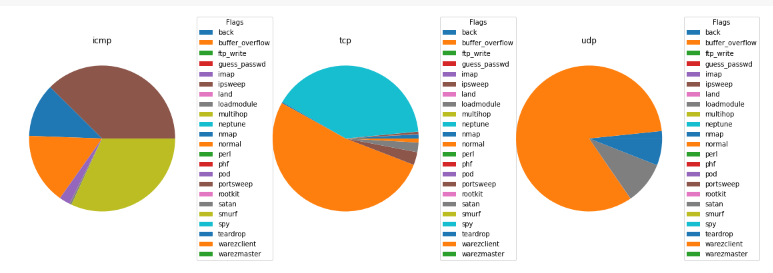
**OUTPUT SCREESHOTS:**

****

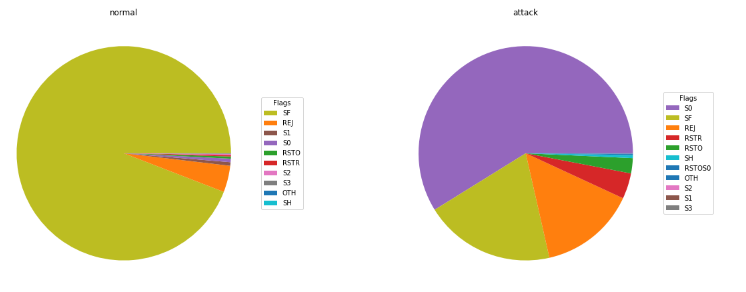
Dataset of KDD test

****

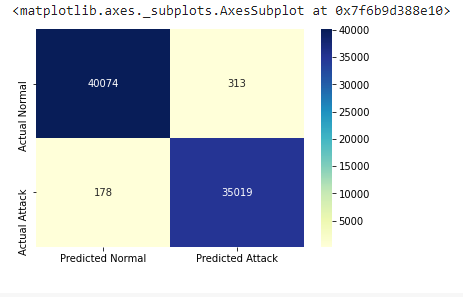
Attack vs protocol

****

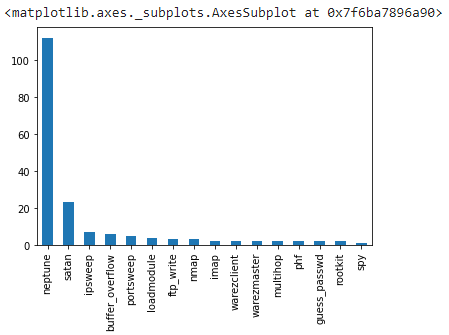
Data profiling

****

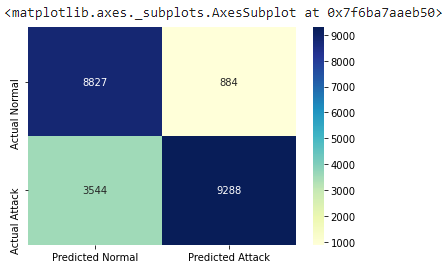
Flag Based Data Profiling



Confusion Matrix

****

False Positive Graph

****

Confusion Matrix on unseen data

**Instructions:**

You must have following library install in your PC:

* numpy
* pandas
* matplotlib
* seaborn
* itertools
* random
* sklearn

Run 'sdn.ipynb' in cells as it is.

**References:**

[1] A. Dixit, F. Hao, S. Mukherjee, T. V. Lakshman, and R. Kompella, “Towards an elastic distributed SDN controller,” ACM SIGCOMM Comput. Commun. Rev., vol. 43, no. 4, pp. 7–12, Oct. 2013.

[2] S. Scott-Hayward, “Design and deployment of secure, robust, and resilient SDN controllers,” in Proc. 1st IEEE Conf. Netw. Softwariza- tion (NetSoft), Apr. 2015.

[3] X.-F. Chen and S.-Z. Yu, “CIPA: A collaborative intrusion prevention architecture for programmable network and SDN,” Comput. Secur., vol. 58, pp. 1–19, May 2016.