Telecommunications software: practical exercise 5

Timothy Picard

1 Task 1

For this task, we first begin by creating two classes: one that represents a tree node, and one representing an binary search tree, thus creating instances of the first class. In the second class are implemented traversal algorithms (pre-order, post-order and in-order) as well as a display function, adapted from J. V. (<https://stackoverflow.com/questions/34012886/print-binary-tree-level-by-level-in-python>).

This allows us to create a binary search tree based on the three lists provided. We can now display the tree and print all the different traversals for this specific tree. After that, we delete a node and repeat this exact procedure:

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Une image contenant texte, diagramme, conception

Description générée automatiquement

As we can see from the output above, all traversals are correct, and the tree correctly rearrange itself after deleting the node 13.

2 Task 2

We first look at the composition of the dataset, and see that there are some object features, *i.e.* strings. This format cannot be used for training a machine learning model, because it has not a numerical value.

Une image contenant texte, capture d’écran, menu, Police

Description générée automatiquement

So, we have to remove some categorical columns from the dataset, that should not be important for the classification anyway (value of source and destination IP addresses, and ID). The variance of the flow rate in the forward direction is represented by an object column, so we need to clean it first. In fact, when trying to build the model, we get this error:



So, we can format it using a regular expression to make it convertible to a float. Then, we can run the program.

Une image contenant texte, capture d’écran

Description générée automatiquement

We use the decision tree classifier from scikit-learn: ID3 is built using the entropy criterion, and the CART using the Gini index. The results are as follows:

Une image contenant texte, capture d’écran, menu

Description générée automatiquement

The ID3 and CART algorithms have almost the same performance, which is around 77 %.

3 Task 3

Firstly, after examining all the XML files provided, only the *TestbedSunJun13Flows.xml* file contains sufficient amounts of “Attack” tags. We will use this file for the analysis.

The dataset is in the form of an XML file, so we have to parse it to make it usable by the machine learning model. For this task we will use a random forest classifier.

This dataset contains a lot of categorical features, so in the code, we use an original encoder from scikit-learn to convert these features into numerical ones. Also, date features are converted into time stamps.

We use a random forest classifier, and obtain this output:

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement

The model performs extremely well on this data. It is reliable in distinguishing malicious flows from normal flows.

4 Task 4

For this task, we proceed in the same way as before, but this time using a random forest regressor, and obtain this output:

Une image contenant texte, Police, capture d’écran, Graphique

Description générée automatiquement

Again, the model is extremely performant.