Identification of operating system processes as self or non-self processes

Aayushi Anand Zutshi

203/CO/15 225/CO/15

**Introduction**

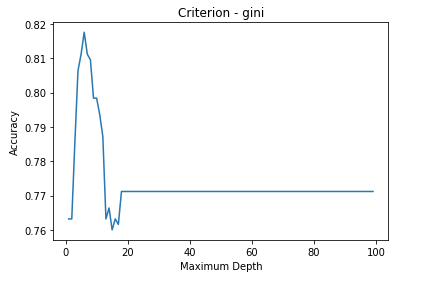
Information security, Cyber security and Computer Security[1] are a vital issue. To provide the highest possible extent of computer security, implementation of an efficient and secure operating system has become a necessity[2]. Some operating system developers design a secure operating system and security tools which work to identify the unauthorized access of the system. With the help of this paper a new methodology is tested and verified to provide maximum security by identification of Self and Non-Self process[3-5] using concepts of Machine Learning Models and Artificial Neural Network . The operating system processes generated by virus, worms etc. can be classified as ‘ Non-Self’. The operating system processes generated for system software or by some reliable application software can be classified as ‘Self’.

**Proposed Methodology**

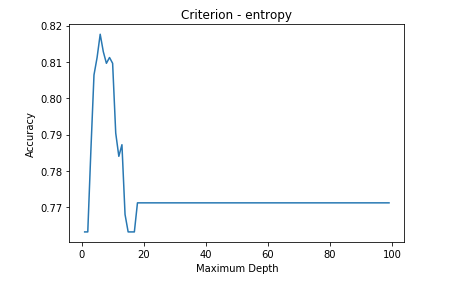
In a computer system all the viruses and attacks create their own processes in the system. Proposed methodology works on the processes and its parameters to identify the process generated by viruses or attacks. These processes will be identified as Non-Self by using a variety of machine learning approaches like Decision tree , KNN, Genetic Algorithm and Artificial Neural Network .

A process has many attributes ,initially five attributes having non-null values which are Process ID, File size, Memory Peak Usage,Page Faults and Page File Peak Usage will be used in the proposed approach. Every process parameter has some values like a number,size of the file in bytes or KB, address in hexadecimal format, character etc. These values have some lower and higher range.The values of parameters are converted into very Low(VL), Low (L), Medium (M), High (H) and very High (VH) to make calculations easy and understanding efficient.

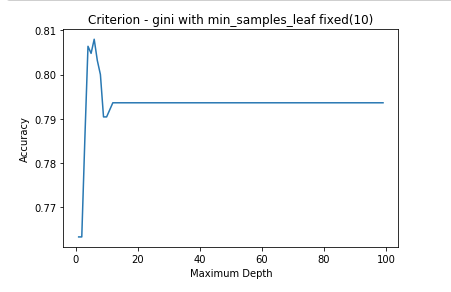
**Experiments, Results and Comparisons**

**Decision Tree Result**s

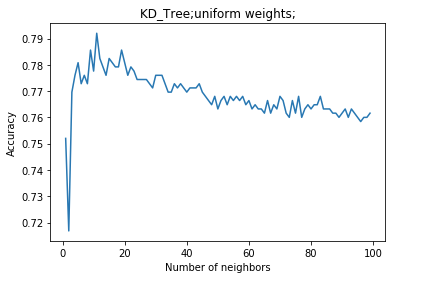
Decision tree algorithms are based on classification using attribute values for taking decisions. Decision tree classifies data from the root node to a leaf node till decisions not made. Decision tree represents data in a fashion which is very easily interpreted by users.

The following results using decision tree show accuracy versus different parameters. The criteria used for the decision tree is entropy and gini in the two comparisons. Moreover, it is seen that highest accuracy of 82% is reached in both the cases.

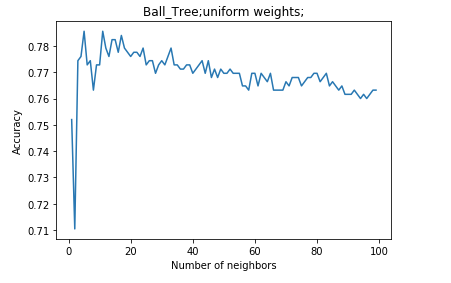
This means that decision tree algorithms can determine the operating system processes with an accuracy of upto 82%.



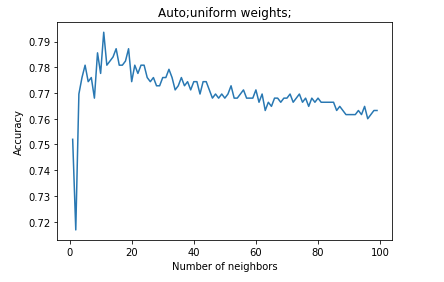
**K Nearest Neighbours Algorithm**



*k*-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

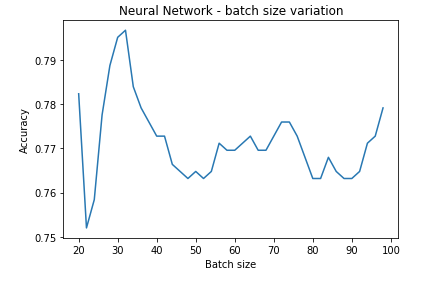
In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small).

In *k-NN regression*, the output is the property value for the object. This value is the average of the values of its *k*nearest neighbors.

The following results using K Nearest neighbors shows the accuracy versus different types of trees used; and also when none is used. Moreover, it is seen that a highest accuracy of 79% is reached in these cases.

This means that the K Nearest neighbors algorithm can determine the nature of operating system processes upto an accuracy of 79%.

**Artificial Neural Network**



**Artificial neural networks** (**ANNs**) or **connectionist systems** are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn" (i.e. progressively improve performance on) tasks by considering examples, generally without task-specific programming. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any a priori knowledge about cats, e.g., that they have fur, tails, whiskers and cat-like faces. Instead, they evolve their own set of relevant characteristics from the learning material that they process.

An ANN is based on a collection of connected units or nodes called artificial neurons (a simplified version of biological neurons in an animal brain). Each connection (a simplified version of a synapse) between artificial neurons can transmit a signal from one to another. The artificial neuron that receives the signal can process it and then signal artificial neurons connected to it.

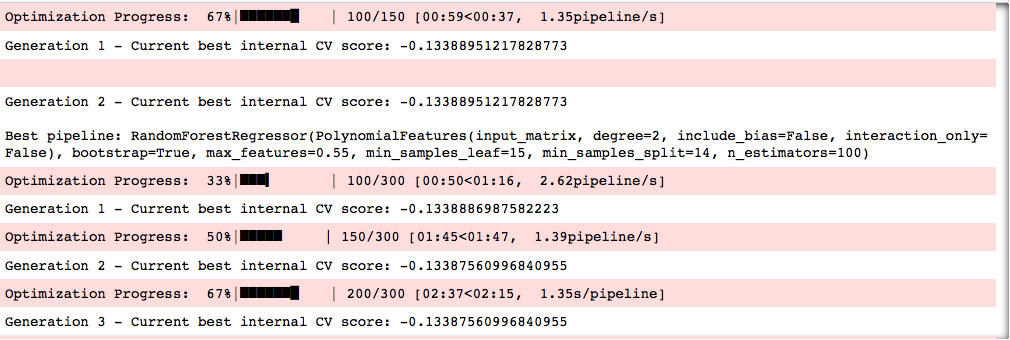
The above figure shows the variation of batch size used to train the neural network versus the accuracy of the testing. It shows an irregular variation of accuracy with the batch size and reaches a maximum of around 80% accuracy.

Thus, neural network can also determine the correct operating system process upto an accuracy of 80%.

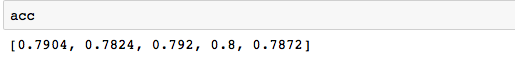
**Genetic Algorithm**

In computer science and operations research, a **genetic algorithm** (**GA**) is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA). Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection.

In a genetic algorithm, a population of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered; traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible.

The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a *generation*. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

When genetic algorithm ran on the dataset above, it produced the following results with the accuracy as showwn below. This shows that genetic algorithm can determine the type of operating system with an accuracy of maximum 80%.



## **References**

1. Solms, R. V., & Niekerk, J. V. (2013). From information security to cyber security. *Elsevier’s Computer & Security*, October, 38, 97-102.

2. Yang, C.Q. (2003). *Operating System Security and Secure Operating Systems*. Global Information Assurance Certification Paper. Retrieved from http://www.giac.org/paper/gsec/2776/operating-system-security-secure-operating-systems/104723

3.Percus, J. K., Percus, O. E., & Perelson, A. S. (1992). Probability of self-nonself discrimination. *Theoretical and Experimental Insights into Immunology*, 66, 63-70.

4. Forrest, S., Hofmeyr, S. A., Somayaji, A. B., &Longstaff, T. A. (1996). *A Sense of Self for UNIX Processes*. Proceedings of IEEE Symposium on Computer Security and Privacy. Retrieved from http://www.cs.unm.edu/~immsec/publications/ieee- sp-96-unix.pdf

5. Forrest, S., & Perelson, A. S. (1994). *Self Non- Self Discrimination in a Computer*. In Proceedings of the IEEE Symposium on Research in Security and Privacy. Retrieved from http://www.cs.unm. edu/~immsec/publications/virus.pdf