Introduction

In Ericsson, many test runs have been executed for testing the software packages in the simulation environment. Before launching product to customer, they need to test how well do the software package perform. The performance of these software packages are evaluated by considering on the CPU utilization, a percent of the CPU's cycle that spent on each process, and some other metrics (e.g., memory usage, latency). CPU utilization is one of the main focus in the test run that need to be optimized for the released of upgrade software package.

Nowadays, they have to do visual inspection by themselves whenever they want to analyze the performance of update software package. With the rise of data generation and a large number of test runs, this work becomes more difficult and perhaps inefficient to do it manually. This is when Machine learning, the algorithm that has an ability to learn from data, comes into focus. The algorithm will help indicate whether the performance of software package is degradation, improvement or in a steady state. There is also a case when the changes in the test environment affect performance even though there is no change in the software package. The implemented algorithm should be able to detect when the test environment is altered.

Background

Ericsson

Ericsson founded by Lars Magnus Ericsson since 1876 is one of the world's leading in telecommunication industry. It provides services, software product and infrasturcture related to information and communications technology (ICT). Its head quater is located in Stockholm, Sweden. Ericsson continuously expands its service and product beyond telecoms sector such as mobile broadbrand, cloud services, transportation and network design.

Objective

The main objectives for this thesis is to implement machine learning to analyze the performance of the system.

- To detect the degradation, improvement or steady state in CPU Utilization
- To detect whether there is some changes in test environment that impact on CPU Utilization

(Detect abrupt changes lying among time-series data)

R vs Python

Both R and Python are powerful programming language for data analysis. Python has been known as a general purpose language with an easy to understand syntax. It emphasizes on code readability and has a gentle learning curve. R is developed by and for statistician. It provides a huge number of essential packages in statistics, and even starts to expand to different fields. R also has a strong reputation for data visualization. However, in terms of computation, R still cannot compete with Python which is buit specifically for computer programming.

With strengths and weaknesses between R and Python described earlier, R is chosen to be used in this study. It offers more implemented algorithms to solve the problem at hand. Moreover, effective visualizion helps analyst and user to understand more about their data especially for the complex one. Visualizing data is useful in many ways such as uncover new patterns, identify some factors, and get an overall trend. R proves itself a good choice since it provides a great feature in creating an interactive graphic and visualization.

ecp package

ecp is an extension package in R which mainly focus on computing nonparametric test for multiple change point analysis. It is applicable to both univariate and multivariate time series. A fundamental idea for building alogrithms that can identify the change point is based on either divisive or agglomerative method in hierrachical clustering approach.

Parametric and Nonparametric test are the two procedures used for performing change point detection. Parametric analysis benefits from assuming some knowledge of data distribution and integrates it to the detection scheme. On the other hand, nonparametric analysis has lessen a restriction/ is more flexible in which there is no assumption made about the distribution. It can, therefore, apply to a wider range of application and capturing various kinds of changes. Generally, real-world data do not always have a well-defined structure and most of the time they are no known distributional form.

`E-divisive` algorithm recursively partitioned a time series. An estimatation of the change point is computed at each iteration. Permutation test is performed to find the statistical significance of an estimated change point.

`E-agglomerative` alogrithm trys to maximized a goodness of fit test after merging segments in the iteration. The estimated change point is defined by the iteration that maximized a goodness of fit statistic. It allows user to input an initial segmentation for the time series or a prior knowledge of the possible change point in order to reduce the computation time.