6. Conclusions

This thesis assesses an ability of detecting any changes in the performance of the Ericsson's software products by applying the Markov switching autoregressive model and the E-divisive method. The simulated datasets with known states were used to make a comparison between both methods. The results from applying the Markov switching model to the real data were presented with interpretations and discussions.

For the Markov switching model, the number of states and the number of switching coefficients in the model were determined and chosen by examining the BIC, along with model output and plot. The findings from the simulated datasets revealed that the Markov switching model were able to discover switches between states rather well, despite some false alarms and missed detections. The E-divisive method is less powerful compare to the Markov switching model. The method could identify fewer change point locations, and failed to detect many changes that were occurred in the simulated datasets. The E-divisive method will perform better and will be more efficient if the data have an obvious pattern of shifting in the time series. Based on the results from the simulated datasets and the real data, the Markov switching model was considered to be the representative method for the analysis. The E-divisive method was rather used as a guideline for any changes that could happen in the data. The method could also be used together with the Markov switching model for a confirmation of the changes in the data when the actual state is unknown. After applying the Markov switching model to both simulated datasets, the accuracy of the test sets implied that an implementation of a state prediction function appears to functionally work well.

Evaluating the obtained results is rather difficult and complicated, mostly due to a lack of annotations or label of the state of the CPU utilization. This is a common situation to an unsupervised learning problem where the ground truths are not often available. To conclude, this work has provided knowledge to understand more about the properties of the state of the CPU utilization which will, in turn, pave the way for further analysis.

6.1. Future work

The Markov switching model gave quite a promising result but several improvements could also be done to increase a robustness of the analysis. For future work, using more observations in the data is recommended. Obtained results will be more reliable as additional information will decrease an uncertainty in the data. As the assumption of the distribution of residuals was not entirely fulfill, significant details that were used to explained the CPU utilization might not all be caught by the model. Another future extension is to consider on the effects of interaction terms or the other predictor variables that might have an effect on the CPU utilization. Furthermore, there are still two more performance metrics in QA Capacity area which have not been taken into account in the thesis (e.g., a memory usage and a latency). The analysis could also be extended to analyze these metrics in a further work. Finally, in the future if some test cases have been labeled by a domain expert, a semi-supervised learning algorithm, a technique that falls between a supervised learning and an unsupervised learning, could also be implemented. Training a model with a large amount of unlabeled data and a small amount of labeled data could considerably improving an accuracy of the model. The semi-supervised learning could be of good practical use especially to an application where labeling all the data is very expensive and time-consuming.