4. Results

The most relevant results of the analysis are shown and organized in this chapter. As a first step, the number of states of the model was decided (Analysis I). Then, the number of parameters that have switching effects in the model was determined (Analysis II). A model selection was performed for Analysis I and Analysis II in order to find the most appropriate model for each given dataset. An analysis of residuals was carried out as a means to validate the models. The results are shown in a later section. Next, the results of a non-parametric analysis are presented, and a comparison between the results of Markov switching model analysis and the results of non-parametric analysis are illustrated. The last two sections report the results of a state prediction of the new observations in each dataset, and an evaluation of the predicting function using a simulated data.

4.1. Analysis I: Number of States

To estimate the set of necessary parameters, an *MSwM*[1] package in R was used. More details about the package can be found in Appendix B. A complete linear Markov switching autoregressive model in this thesis framework is defined as

$$\begin{aligned} y_t =& \beta_{intercept,S_t} + \beta_{RrcConnectionSetupComplete,S_t} X_{RrcConnectionSetupComplete,t} \\ &+ \beta_{Paging,S_t} X_{Paging,t} + \beta_{X2HandoverRequest,S_t} X_{X2HandoverRequest,t} \\ &+ \beta_{DuProdName,S_t} X_{DuProdName,t} + \beta_{Fdd/Tdd,S_t} X_{Fdd/Tdd,t} \\ &+ \beta_{NumCells,S_t} X_{NumCells,t} + \phi_{1,S_t} y_{t-1} + \varepsilon_{S_t} \end{aligned} \tag{4.1}$$

The estimation was made under the assumptions of two or three states $S_t \in S$, where $S = 1, 2, .., k$ and $k = 2$ or 3. These two numbers come from a hypothesis that the state of the CPU utilization might have two states (*Normal* and *Bad*, *Normal* and *Good*, *Bad* and *Good*) or three states (*Normal*, *Bad*, and *Good*). During the estimation, a normality assumption was also applied to the distribution of residuals.

BICs from fitting the Markov switching autoregressive model are shown in Table 4.1. For the software release L16A, the BIC suggests that the three-state Markov switching autoregressive model gives a better fit in comparison to the two-state model.

[1] https://cran.r-project.org/web/packages/MSwM/index.html

However, the models with two states for the remaining two software releases, L16B and L17A, had lower BICs.

Table 4.1.: BIC of the model with two and three states. The left column gives the different datasets.

Software release	BIC	
	$k = 2$	$k = 3$
L16A	439.677	417.682
L16B	1,763.507	1,797.259
L17A	1,189.061	1,199.075

4.1.1. Software release L16A

Before performing the Markov switching autoregressive model, a standard linear regression model was fitted to the dataset first. It was found that one coefficient in the dataset of the software release L16A was not defined because of singularity i.e., a perfect correlation between predictor variables. Hence, *DuProdName* variable was dropped from Equation 4.1.

Figure 4.1 presents that the Markov chain remained in State1 for an extensive period of time before it switched to State2. When the chain is in State2, it stays there only a short time and then quickly moves back to State1. There are a few switches between these two states in Figure 4.1. On the other hand, it is visible that there are more switches between states in Figure 4.2. Note that State2 in the two-state model seems to be defined as State1 in the three-state model instead. Moreover, the periods of State1, which has a rather long duration in the two-state model, now contains several switches between states in the three-state model.

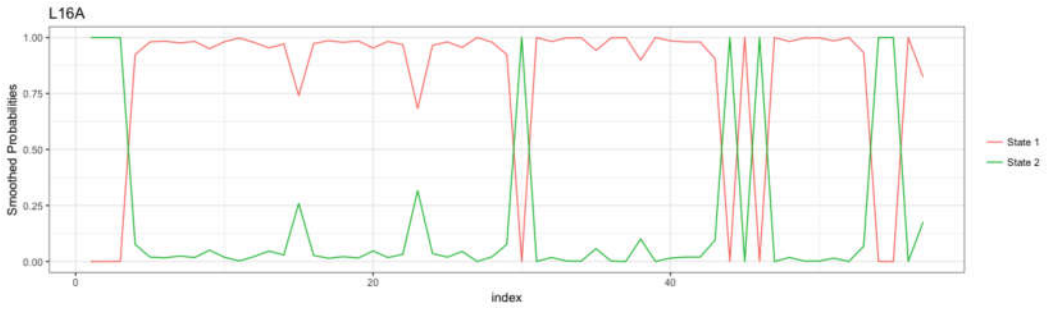

Figure 4.1.: The smoothed probabilities of the software release L16A with two-state model

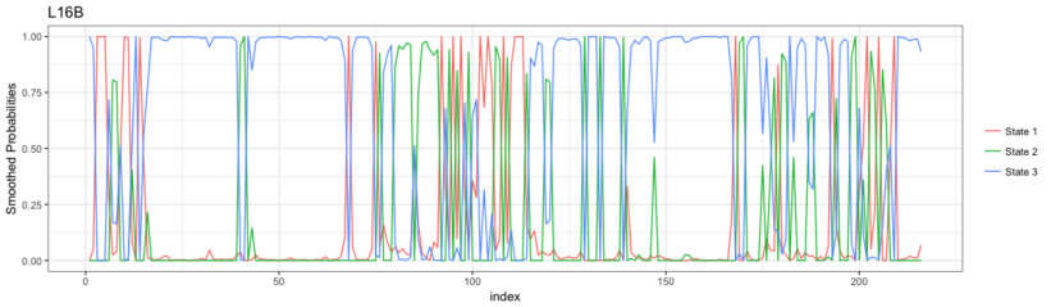

Figure 4.2.: The smoothed probabilities of the software release L16A with three-state model

4.1.2. Software release L16B

In Figure 4.3, the Markov chain has several periods where it switches back and forth between two states of the software release L16B. The durations of the chain being in State2 is longer than the durations of the chain staying in State 1. Although the chain temporarily stays in State1, it remains in this state for a few moments in the middle of the time period (observation 91-99 and 101-114) before returning to State2. Apparently, there are more switches between states in the three-state model, especially in the beginning, middle, and at the end of the period. Figure 4.4 shows that the chain remains in State3 over a considerable period as shown throughout observation 15-39, 42-67, and 140-170.

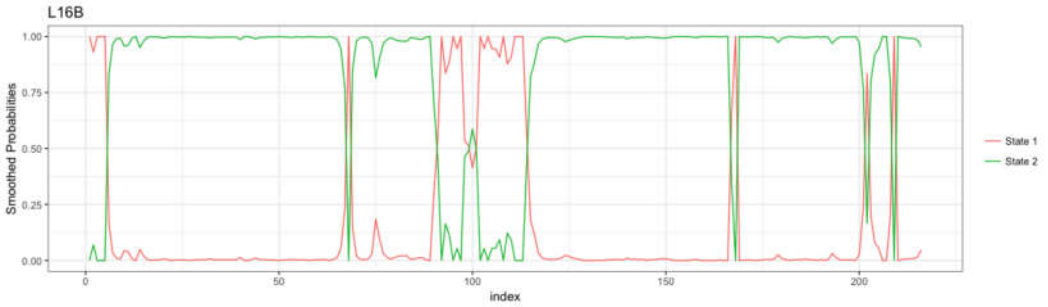

Figure 4.3.: The smoothed probabilities of the software release L16B with two-state model

4.1.3. Software release L17A

There are a number of switches between states in the two-state model of the software release L17A. In Figure 4.5, when the Markov chain is in State1, it continues to stay in its state for a while before leaving to State2. Furthermore, the chain has a

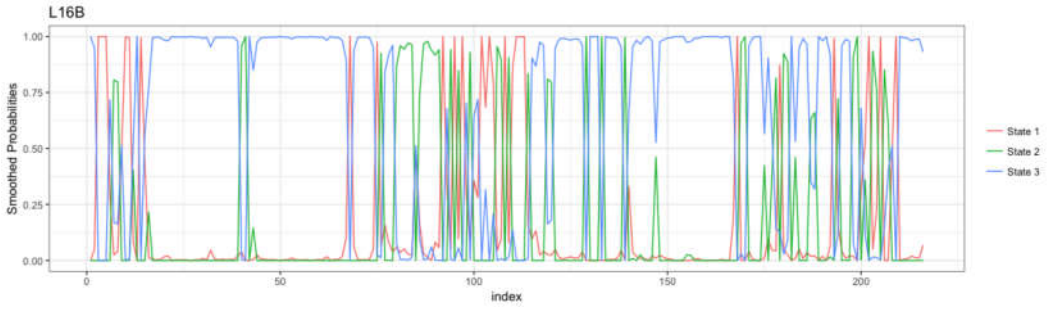

Figure 4.4.: The smoothed probabilities of the software release L16B with three-state model

fairly short duration of staying in State2. After the chain visits State2, it instantly switches back to State1. Figure 4.6 presents the chain which has many switches between State1 and State2 in the first half of the time period. The chain for the three-state model also stays in State2 significantly long from observation 104 to 129, which is the end of the time series.

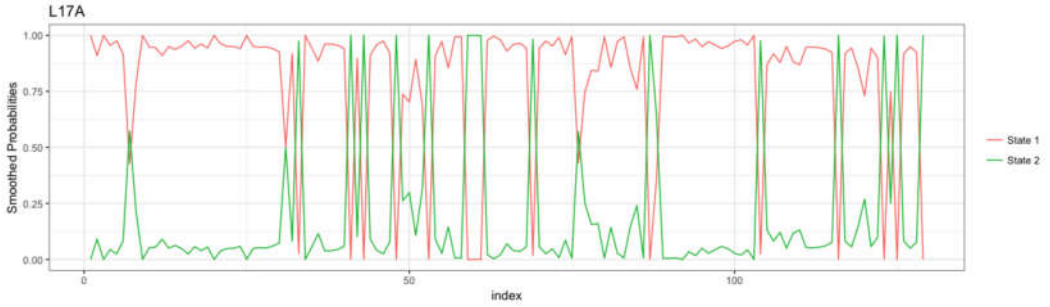

Figure 4.5.: The smoothed probabilities of the software release L17A with three-state model

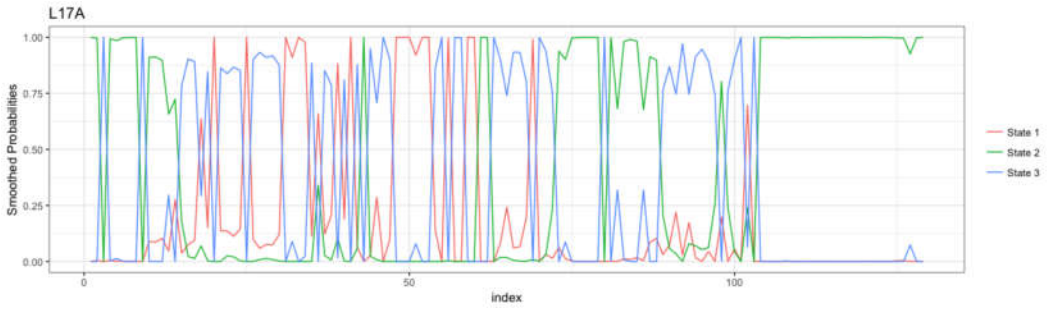

Figure 4.6.: The smoothed probabilities of the software release L17A with three-state model

After examining the outputs from the models along with the plots, the three-state models for each software release were further analyzed in the thesis. More details are provided in chapter 5.

4.2. Analysis II: Number of Switching coefficients

The fitted Markov switching autoregressive models in sec. 4.1 were performed by assuming that every parameter in the model had switching effects i.e., coefficients can have different values in different periods. However, in practice, each coefficient can have either a switching or non-switching effect. Therefore, Markov switching autoregressive models were applied to each dataset again but with a hypothesis that the variables considered as a test environment are possible to have non-switching effects. In this section, the structure of all the models from all three datasets are reported in the table. The best model is selected for each dataset and its state specification is presented in the plots. Further discussion and details about these chosen models are provided in sec. 5.1. It should be noted that these three chosen models will later be used throughout this thesis and the model outputs are shown in Appendix C.

4.2.1. Software release L16A

For the dataset of the software release L16A, *DuProdName* was not included in the model fitting as explained previously. Only two variables of the test environment were left to try whether they could have non-switching effects or not. The result is shown in Table 4.2. The second model has the highest BIC and even higher than the model which have all switching coefficients. The first model, where both *Fdd/Tdd* and *NumCells* have switching effects, was selected to be used with this dataset.

Table 4.2.: List of the model structure of the software release L16A along with its BIC. The last line is the result taken from the three-state model in the Analysis I. The line in bold indicates the selected model.

Model	Switching effect		BIC
	Fdd/Tdd	NumCells	
1	**N**	**N**	**413.408**
2	N	Y	438.371
3	Y	N	401.232
	Y	Y	417.682

Figure 4.7 indicates the CPU utilization of the software release L16A and also shows the periods of the derived state from the model. From the plot, State2 clearly has

the longest duration to remain in its own state. When the chain moves to either State1 or State3, it immediately switches to the other states. However, the duration that the chain stays in State1 is longer in the beginning and almost at the end of the period. Another characteristic that could be observed is that when State2 happens to have more chance to switch to State3 rather than switch to State1.

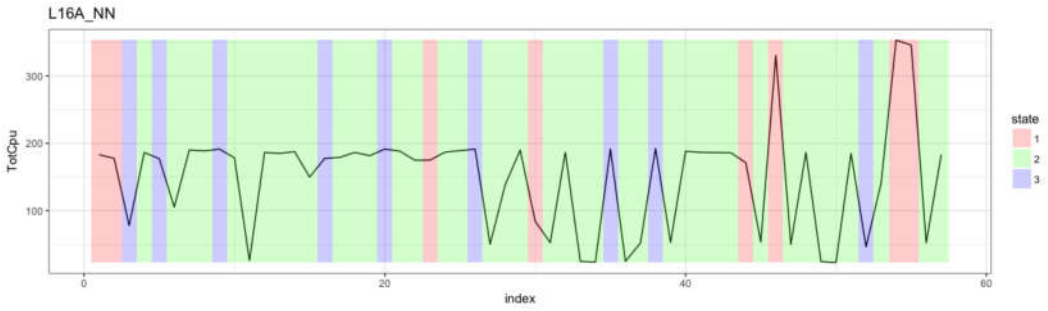

Figure 4.7.: The CPU utilization of the software release L16A showing the periods where the observation is in the specific state.
Model 1: Fdd/Tdd and Numcells are non-switching coefficients.

4.2.2. Software release L16B

For the software release L16B, Table 4.3 presents the results of fitting the model with different combinations of switching coefficients. Models 5 and 7 have higher BICs than the model which have switching effects in all coefficients. The second model, where *DuProdName* and *Fdd/Tdd* are non-switching coefficients, has the smallest BIC. The chosen model for this dataset is the model which has only *DuProdName* as a non-switching coefficient or model 4.

Many switches between states can easily be seen in Figure 4.8. However, the state which has the longest duration remaining in its own state is State3. There are three durations where the chain stays in State3 for a long time. Another noticeable behavior from this switching mechanism is that there are several switches between State1 and State2 in the beginning, middle, and at the end of the time period.

Table 4.3.: List of the model structure of the software release L16B along with its BIC. The last line is the result taken from the three-state model in the Analysis I. The line in bold indicates the selected model.

Model	Switching effect			BIC
	DuProdName	Fdd/Tdd	NumCells	
1	N	N	N	1,787.528
2	N	N	Y	1,704.393
3	N	Y	N	1,784.384
4	**N**	**Y**	**Y**	**1,776.102**
5	Y	N	N	1,806.385
6	Y	N	Y	1,725.865
7	Y	Y	N	1,804.487
	Y	Y	Y	1,797.259

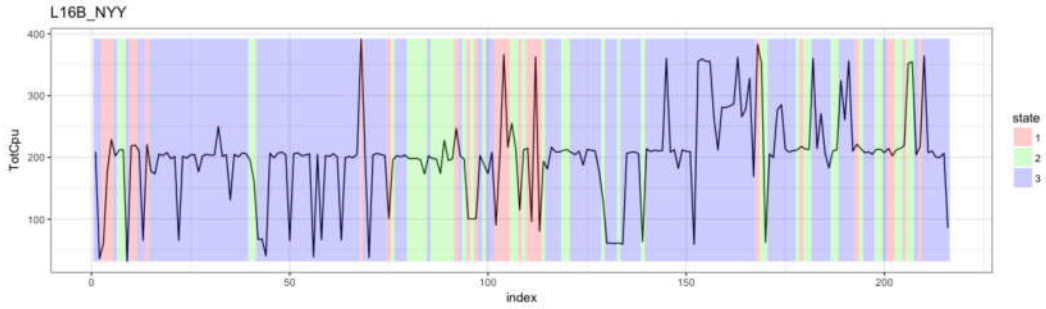

Figure 4.8.: The CPU utilization of the software release L16B showing the periods where the observation is in the specific state.
Model 4: DuProdName is non-switching coefficient.

4.2.3. Software release L17A

Table 4.4 presents model structure of the software release L17A. There is model 2 which has higher BIC than the model which have all switching coefficients. The least BIC is from the first model that all three variables in the test environment have non-switching effects. This model was also chosen to be further used for this dataset.

Several switches between three states occur in the beginning of the time series as shown in Figure 4.9. Around the end of the time series period, State3 appears to have a longer duration and fewer switches to State1. State2 seems to be the only state which has a fairly short duration for the chain to stay in the state. Furthermore, State2 tends to switch to State1 more often than to switch to State3.

Table 4.4.: List of the model structure of the software release L17A along with its BIC. The last line is the result taken from the three-state model in the Analysis I. The line in bold indicates the selected model.

Model	Switching effect			BIC
	DuProdName	Fdd/Tdd	NumCells	
1	**N**	**N**	**N**	**1,140.474**
2	N	N	Y	1,204.280
3	N	Y	N	1,152.740
4	N	Y	Y	1,184.643
5	Y	N	N	1,146.000
6	Y	N	Y	1,189.236
7	Y	Y	N	1,157.311
	Y	Y	Y	1,199.075

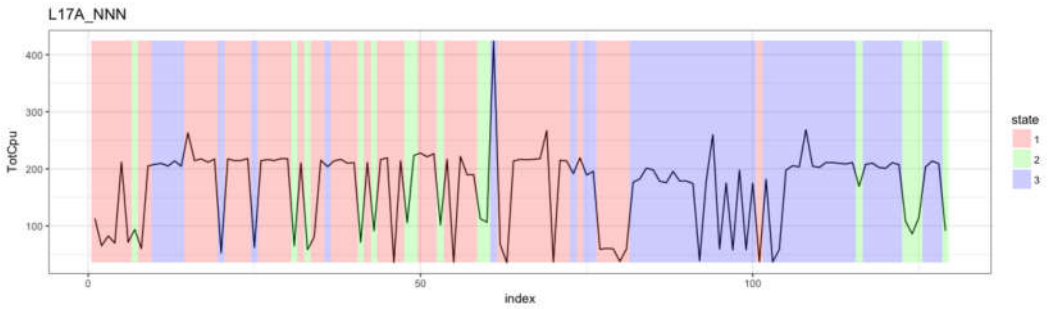

Figure 4.9.: The CPU utilization of the software release L17A showing the periods where the observation is in the specific state.
Model 1: DuProdName, Fdd/Tdd, and NumCells are non-switching coefficients.

4.3. Residual analysis

Pooled residuals of the selected Markov switching autoregressive model from sec. 4.2 were analyzed to see how well the model fitted an assumption of a normal distribution. A Quantile-Quantile (Q-Q) plot is an effective tool for assessing normality. Moreover, an Autocorrelation function (ACF) and a Partial Autocorrelation Function (PACF) of residuals are a useful technique to check on the independence of noise terms in the model. The Q-Q plot and the ACF/PACF plot play a significant role in the residual diagnostics. These plots of each dataset are shown below.

In Figure 4.10, the pooled residuals appears to fall in a straight line with some deviations in its tails. There is an evidence of autocorrelation in the residuals of this model, which can be seen in both ACF and PACF plot, at lag 8.

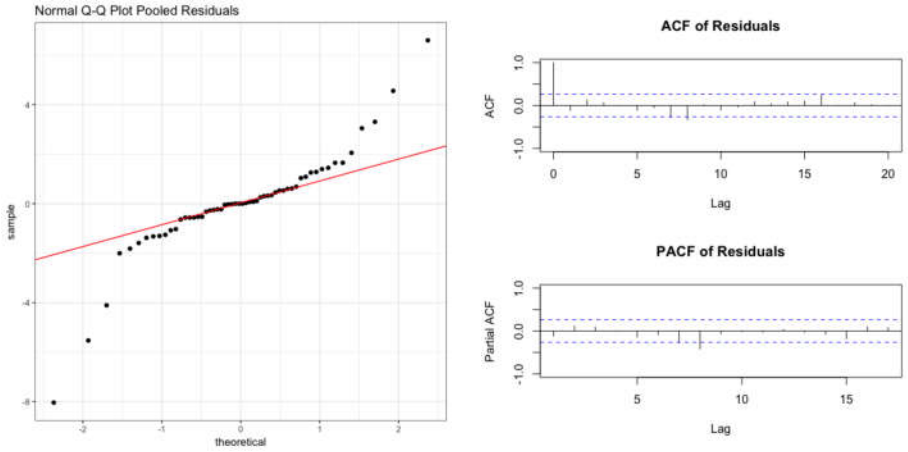

Figure 4.10.: The normal Q-Q plot and the ACF/PACF of pooled residuals of the software release L16A

Figure 4.11 presents points that form a straight line in the middle of the plot, but curve off at both ends. This is a characteristic of a heavy-tailed distribution. The data has more extreme values than it should be if the data truly comes from a normal distribution. In addition, both the ACF and PACF plot show that there is a small amount of autocorrelation remaining in the residuals. The statistically significant correlation of this model are at lags 6 and 10. The significant at lag 4 both in the ACF and PACF plot is slightly higher than two standard errors.

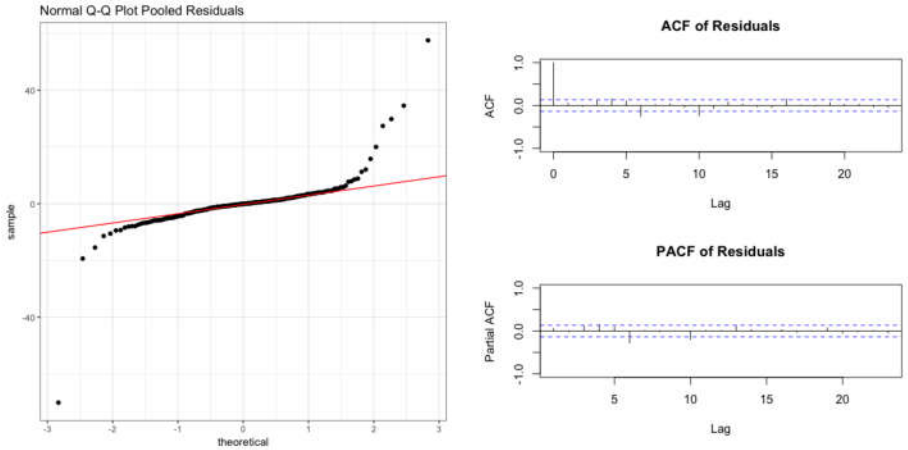

Figure 4.11.: The normal Q-Q plot and the ACF/PACF of pooled residuals of the software release L16B

A Q-Q plot in Figure 4.12 suggests that a distribution of the pooled residuals may have a tail thicker than that of a normal distribution. It is visible that there are many extreme positive and negative residuals in the plot. Furthermore, the ACF

plot of pooled residuals are significant for the first two lags, whereas the PACF plot is significant only at lag 2.

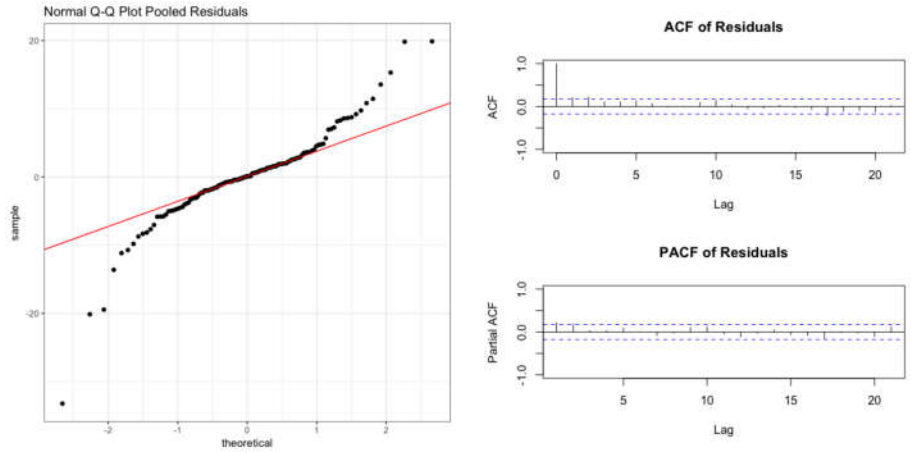

Figure 4.12.: The normal Q-Q plot and the ACF/PACF of pooled residuals of the software release L17A

4.4. Non-parametric analysis

An E-divisive method was applied to all three datasets. The method reported one cluster for the dataset of the software release L16A. There were five clusters found in both datasets of the software release L16B and L17A. Table 4.5 shows places in the time series data where the E-divisive algorithm is able to detect the significant changes.

Table 4.5.: The locations of the statistically significant change points from applying the E-divisive algorithm in each dataset

Software release	Change-point location
L16A	-
L16B	130, 135, 153, 170
L17A	9, 77, 82, 105

The CPU utilization of the software release L16A, L16B and L17A along with its estimated change points in the time series are plotted and shown in Figure 4.13, Figure 4.14 and Figure 4.15, respectively. It can be seen that the E-divisive method could not identify any changes in the dataset of the software release L16A.

Four change points were identified from the method for the software release L16B and L17A. In Figure 4.14, the estimated change points for the dataset of the software

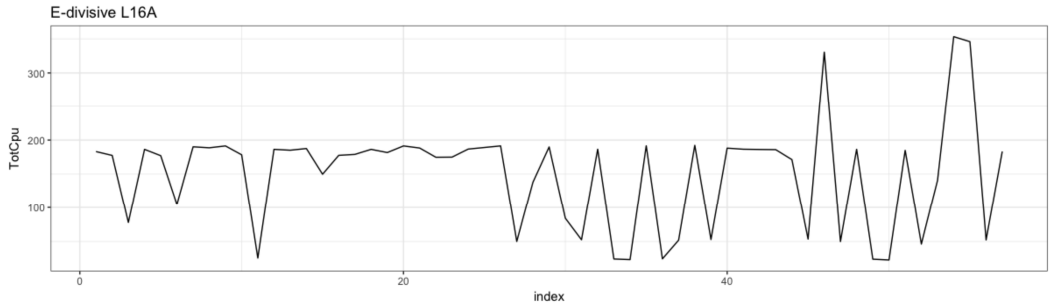

Figure 4.13.: The CPU utilization of the software release L16A

release L16B are likely to occur around the same period of time, which are almost at the end of the time series data. The estimated points from the method are approximately at peaks and negative peaks.

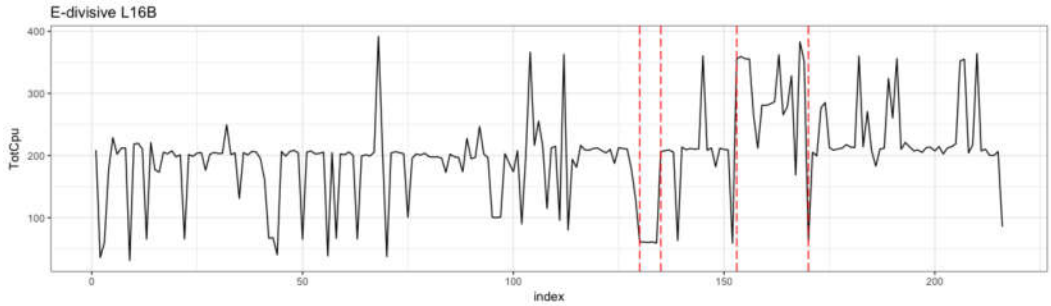

Figure 4.14.: The CPU utilization of the software release L16B. The red dashed vertical lines indicate the locations of estimated change points.

On the contrary, the result of the dataset of the software release L17A, which is shown in Figure 4.15, have the estimated change points rather spread out. The E-divisive method discovers changes when the CPU utilization was about to drop or increase its value.

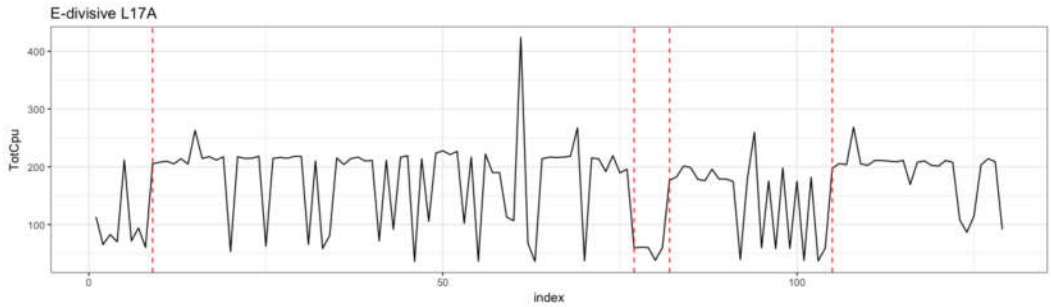

Figure 4.15.: The CPU utilization of the software release L17A. The red dashed vertical lines indicate the locations of estimated change points.

4.4.1. Comparison between the Markov switching autoregressive model and the E-divisive

One noticeable thing is that the E-divisive method was able to identify the changes of the data less than the Markov switching autoregressive model. As mentioned previously, no estimated change points was discovered when applying the E-divisive algorithm to the dataset of the software release L16A. Thus, a comparison between two methods could not be made for this dataset.

Figure 4.16 presents results of switches between states from the Markov switching autoregressive model and change point locations from the E-divisive method for the software release L16B. There is one location where the E-divisive method detects a change but the Markov switching autoregressive model does not show any switches. As can be seen, the E-divisive method appears to detect changes when State2 switches to State3. At observation 130, the E-divisive method discovers a switch in the state at the same time as Markov switching autoregressive model does. However, the E-divisive method identifies switches after and before the Markov switching autoregressive model does at observations 135 and 170, respectively.

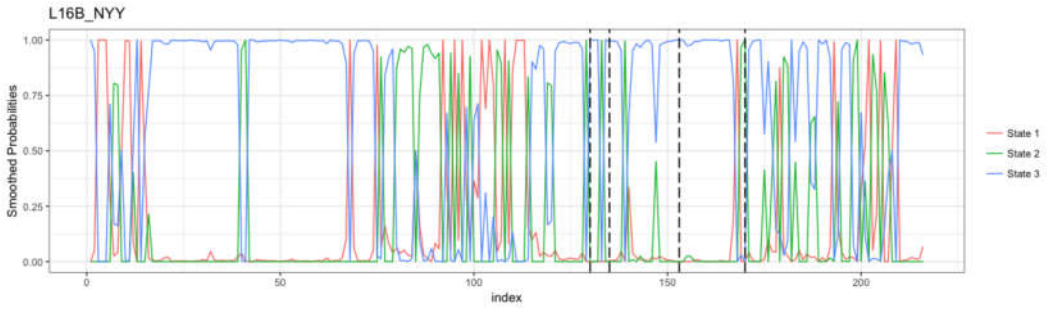

Figure 4.16.: The combined results of the Markov switching autoregressive model and the E-divisive method for the software release 16B

In Figure 4.17, a change is reported from the E-divisive method at observation 105 whereas no switch between states could be found from the result of the Markov switching autoregressive model. It is visible that the E-divisive method is able to detect a switch from State1 to State3, and also a switch from State3 to State1. Both two methods could detect the changes at the same period of time at observations 77 and 82. The E-divisive method detects a change at observation 9 which is before there was a switch in the state from the result of the Markov switching autoregressive model.

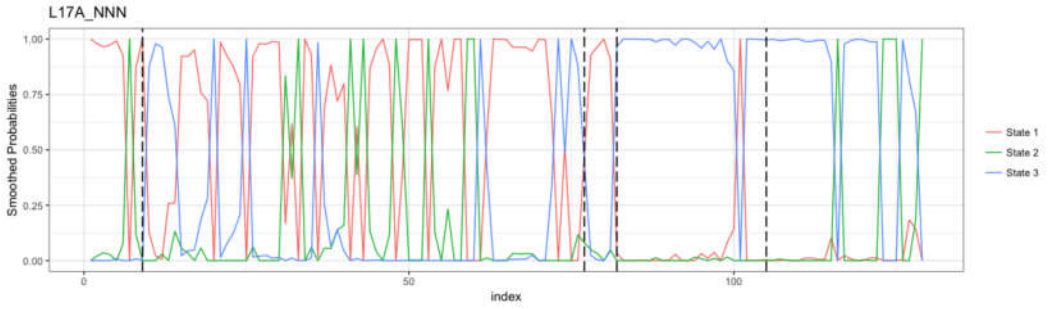

Figure 4.17.: The combined results of the Markov switching autoregressive model and the E-divisive method for the software release 17A

4.5. Predicting the state of the CPU utilization

Now, an implemented of state prediction function was applied to the test set in order to find the most probable state for new observations. For the software release L16A, there are 7 observations in a test set. In Figure 4.18, only two states, which are State1 and State2, are assigned for these observations. The first three observations are in State2. Afterwards, observation tends to switch back and forth between states until the end. It is noticed that the last observation does not belong to any state. After applying a predict function to the test set, the function is unable to predict the most likely state for the last observation of the test set.

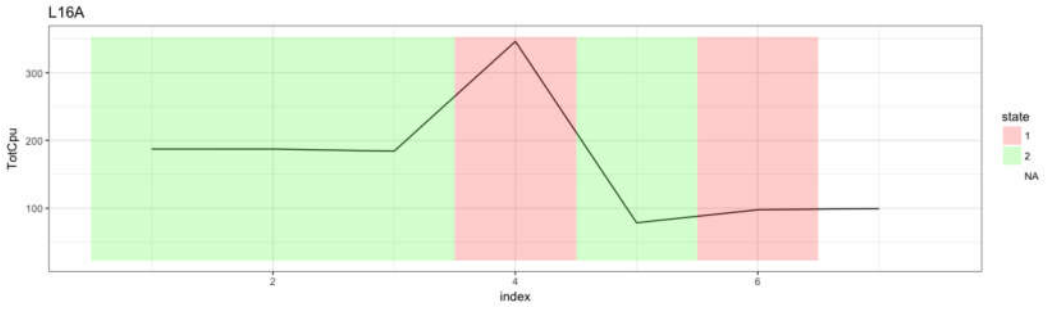

Figure 4.18.: The predicted state of the test set in the software release L16A

In total, there are 25 observations in a test set of the software release L16B. The result after applying the predict function to the test set is shown in Figure 4.19. Observation 15 is the only observation which is in State2. Many switches between State1 and State2 can be seen from the plot. In addition, observation appears to stay in State1 only a short time before moving to State3, except for the first five observations.

Fifteen observations is in a test set of the software release L17A. Figure 4.20 presents a considerably long period for staying in State2, which is from observation 10 to the

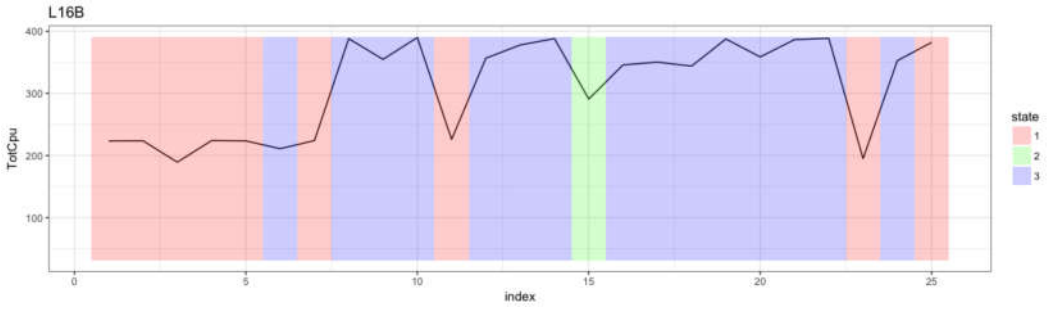

Figure 4.19.: The predicted state of the test set in the software release L16B

end of the time series data. There are several switches between states happening in the plot. As can be seen, observation between 4 and 7 swap between states fairly quick. Observation visits the particular state for one time and then moves to the other states.

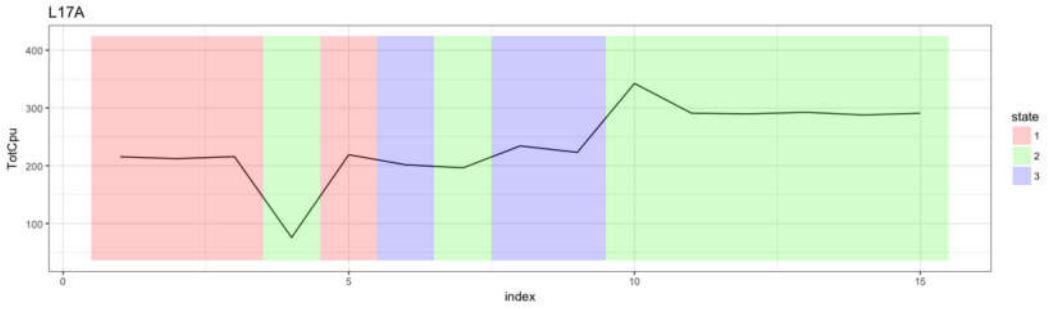

Figure 4.20.: The predicted state of the test set in the software release L17A

4.6. Assessing state prediction using a simulation technique

Markov switching autoregressive model was fitted with eighty percents of the observations from a simulated data, and the remaining was used as a test set to evaluate a performance of the model. The result of the model performance using Dataset 1 is shown in Table 4.6. There were two observations from the *Bad* state which were wrongly classified to the Normal state. Moreover, another two observations from the *Good* state were classified to the *Normal* state. The overall accuracy of the model was 0.96, and the misclassification rate was 0.04. One can see that the model was able to correctly predict the observations which had *Bad* and *Good* state.

Table 4.7 presents a confusion matrix for a test set from a second simulated data. One can see that the model able to correctly predict the observations which had

Table 4.6.: Confusion matrix after applying the Markov switching autoregressive model to fit with the test set from the simulated Dataset 1

		Predicted state		
		Bad	Normal	Good
Actual state	Bad	58	2	0
	Normal	0	30	0
	Good	0	2	8

Bad state. On the contrary, the model did not perform well in predicting observation which had *Good* state. Nine observations were classified to *Bad* state while another five observations were classified to *Normal* state. Moreover, there were six observations from the *Normal* state which were wrongly classified to the *Good* state. The overall accuracy of the model and the misclassification rate was 0.8 and 0.2, respectively.

Table 4.7.: Confusion matrix after applying the Markov switching autoregressive model to fit with the test set from the simulated Datatset 2

		Predicted state		
		Bad	Normal	Good
Actual state	Bad	35	0	0
	Normal	0	29	6
	Good	9	5	16