Fall 2020
CMPT 318: Term Project

Group 12

Alexander Wang	301 366 260
Alexis Lazcano	301 371 074
Jason Leung	301 374 882

Abstract

This project report will go over the process required to detect anomalies in the context of protecting critical infrastructure. The dataset used in the following report was provided by an instructor. The following methods and topics will be covered: Principal Component Analysis, Hidden Markov Model training, normalized log-likelihood, moving average, and linear/polynomial regression. The results and analysis of the data will be provided, interpreted and explained throughout the report. This report was not written by professionals and only covers the basic foundations and methods for anomaly detection.

1. Table of Contents

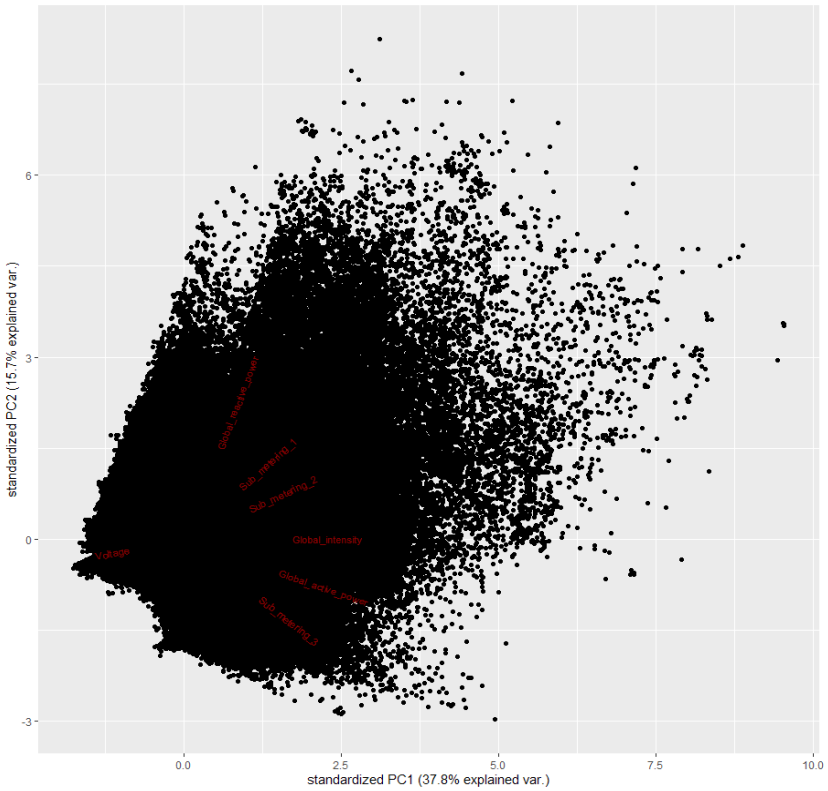
2. Table of Figures

3. Technical Report

3.1 Data Analytics

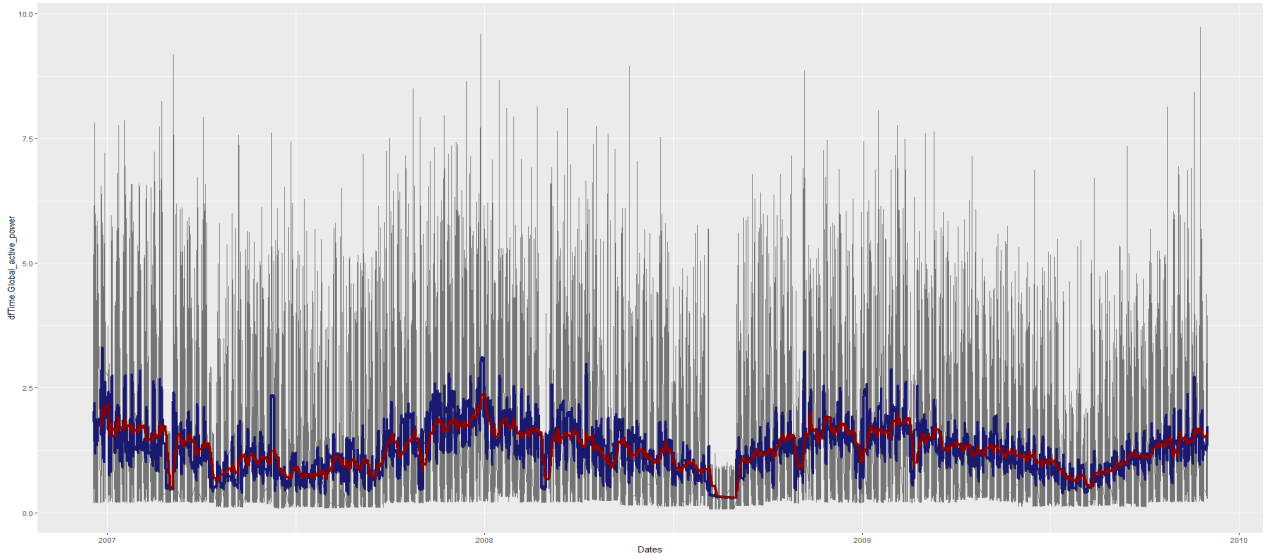
For this project, we were required to detect anomalies amongst the provided dataset, something that we may have to do in the future if we decide to focus on the path of cybersecurity. Some information about how we started going about the project. First, we filtered the provided dataset in order to reduce the size from a total of 1,548,072 objects to 646,436 objects. We chose to make the time period 6:00 AM to 8:00 PM on weekdays, since it is the period where most, if not everyone will have the most fluctuation in their activities. Such as waking up in the morning, going to work, and returning home.

As for how we chose our variables, most of the reason lay with the Principal Component Analysis we applied to our dataset, however there were also other minor decisions that led us to our choice.

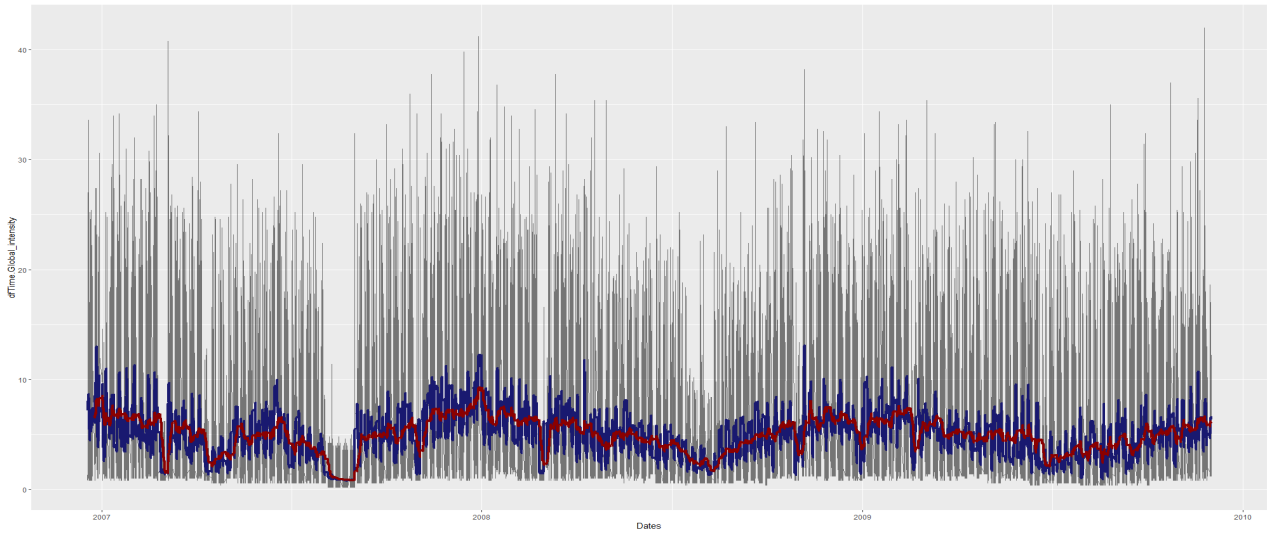


The Principal Component Analysis as shown in the graph above shows which variables contributed the most to the variation in the dataset. As you can see, PC1 is responsible for 37.8% of the dataset, and PC2 is responsible for 15.7% of the dataset. Furthermore, it is shown that Global_intensity contributed the most towards PC1. Meanwhile Sub_metering_2, Sub_metering_1, and Global_reactive_power contributed towards both PC1 and PC2. Ideally, choosing Global_intensity and Global_reactive_power as the variables to use would have been amazing. However, using Global_reactive_power caused a few problems, and using Sub_metering_2 and Sub_metering_3 gave us a smaller dataset than we would have liked to work with. Therefore, we made the decision to use Global_active_power instead as it also contributed majorly towards PC1. We were also leaning towards using Global_reactive_power when it failed to give us the results we wanted. Nevertheless, we decided to follow the Principal Component Analysis results and attempted to work with Sub_metering_2 and Sub_metering_3 first.

Global_active_power



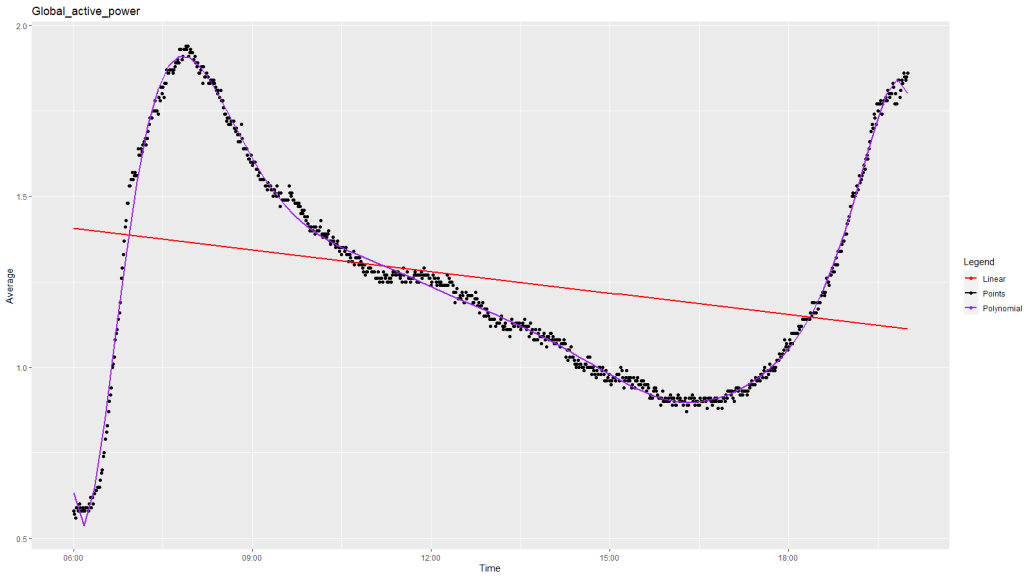
Global_intensity



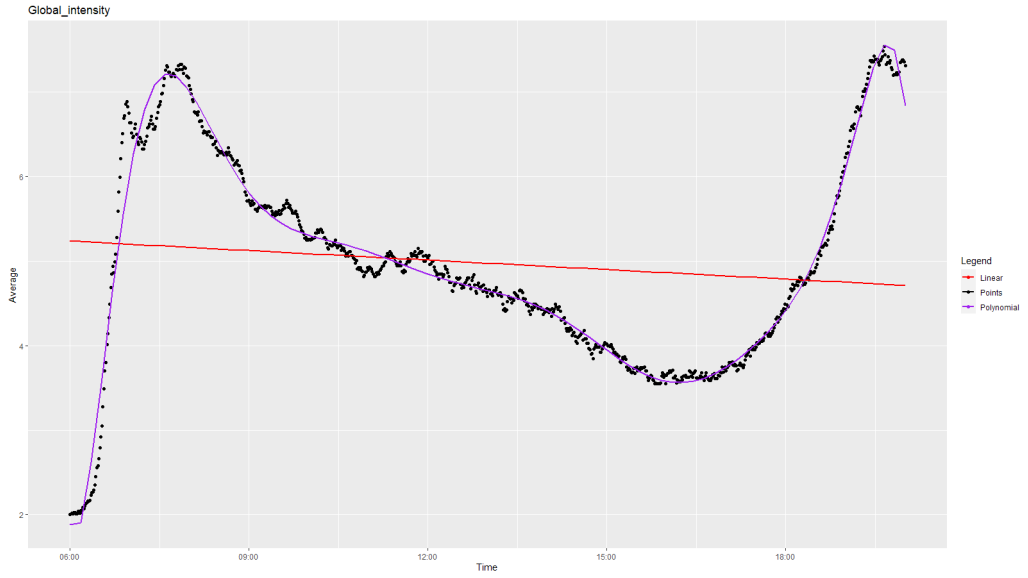
The results of our moving average are shown above for both Global_active_power and Global_intensity. As you can see, the location of the anomalies are not very clear due to the large time period the moving average was calculated with. However we decided to keep this large time period as we believed it would present more information from an overall view for us to use, as compared to a smaller time period where it will have taken more time, and may not let us see the general picture. Due to plotting a large amount of data we were working with onto a small graph, we decided using moving average to locate the anomalies was not an efficient method.

Afterwards, we decided to attempt to use the linear and polynomial regression to plot our data to see if we could gain any helpful information from it. The end outcome is that polynomial regression ended up being very useful towards our conclusion and final results.

Global_active_power



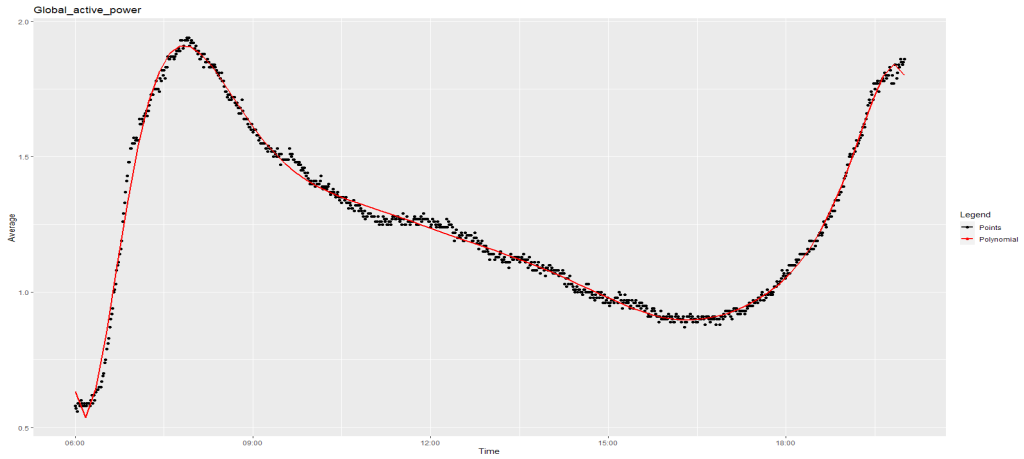
Global_intensity



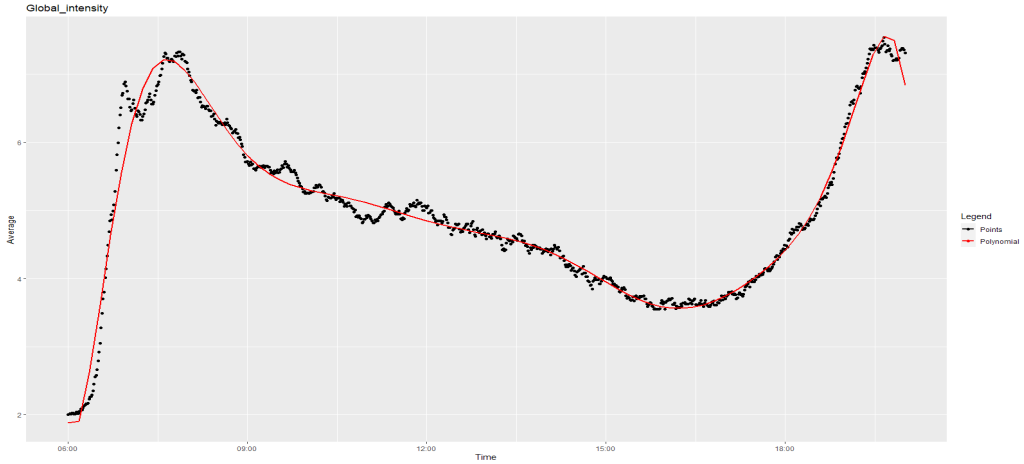
The two graphs shown above are the linear and polynomial regressions for Global_active_power and Global_intensity. It is clear that linear regression would not be very helpful in this case due to the changing pattern amongst the data. However, the polynomial regression line with a degree of 11 fit the data very well.

Furthermore, if there are any anomalies present in the dataset, it would be very easy to spot it due to how clear the data is presented in the graphs. Which was a problem that we had with the moving average that caused us to switch to a different method.

Global_active_power



Global_intensity



Due to the reasons stated in earlier paragraphs, we have chosen the two graphs above to be our probabilistic model for representing normal system behaviour.

To conclude, there were many reasons leading to our decision of making the above graphs our probabilistic model for normal system behaviour. First, due to using the variables in the data that contribute the most to variation, it is accurate and reliable. Second is that it would be easy to spot anomalies due to the clarity of the data. Third, it is during a time frame where it encapsulates the everyday activities that people go through during their day. This makes the graph a great choice to be the model representing normal system behaviour in this project.\

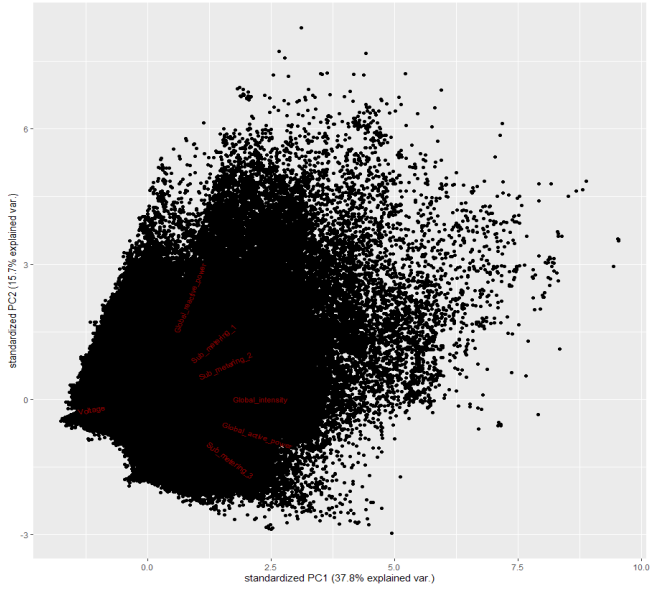
3.2 Project Report

In the following project report, the problem at hand is to find a reliable method to detect anomalies in the context of protecting critical infrastructure. We will be detailing the steps and choices we made as we progressed with the dataset provided, as well as the results we received in the process of solving this problem. The methods we use will include the Principal Component Analysis, Hidden Markov Model training, the comparison of normalized log-likelihoods, the moving average, and the linear and polynomial regression

Principal Component Analysis (PCA)

The first step we took was to apply the Principal Component Analysis on the provided dataset between the chosen timeframe. These were the results:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
Standard Deviation	1.6327	1.0496	0.9794	0.9198	0.9047	0.7017	0.3667
Proportion of Variance	0.3782	0.1574	0.1370	0.1209	0.1169	0.0704	0.0192
Cumulative Proportion	0.3782	0.5356	0.6726	0.7935	0.9104	0.9808	1.0000



As seen in the table and graph provided above, PC1 is responsible for 37.8% of the total variation in the dataset, and PC2 is responsible for 15.7% of the total variation in the dataset. Furthermore, it shows that Global_intensity contributes the most to PC1, while contributing none to PC2. Meanwhile, Global_reactive_power, Sub_metering_3, and Sub_metering_2 contribute the second most to PC1. It is due to this fact that Global_intensity was decided to be one of the variables used. However, for the second variable, there was a debate between choosing Sub_metering_1, Sub_metering_2, and Global_reactive_power, as they contribute to both PC1 as well as slightly to PC2. Using one of these variables as the second pick would have been ideal.

However, in the process of using any one of these variables, many problems occurred when training the Hidden Markov Models, and the decision to use Global_active_power was made.

Hidden Markov Models (Univariate)

The second step taken was to train the univariate Hidden Markov Models for our variables to ensure that the ones we picked were reasonable choices.

The first table shown here are the results of the univariate Hidden Markov Model training for Global_intensity:

States	Log-Likelihood	Normalized Log-Likelihood	BIC
N = 5	-800,775.2	-1.239	1,602,005
N = 10	1,939,021	3.000	-3,876,449
N = 8	55,566.65	0.086	-110,076.3
N = 7	**-686,886.4**	**-1.063**	**1,374,604**

We started off training our univariate HMM with 5 states, and saw that the resulting log-likelihood was -800,775.2. After seeing how high the log-likelihood was, we jumped to 10 states to attempt to get closer to the best number of states. When the resulting log-likelihood came out as positive, we narrowed down the best range to be between 5 and 10 states. The first choice after decreasing the number of possible states was 8. The results came out as positive, and thus we made the decision to test state 7. Once the results came back negative, we determined that 7

states is the ideal number for training the univariate Hidden Markov Model with Global_intensity.

The second table shown here are the results of our univariate Hidden Markov Model training for Global_active_power:

States	Log-Likelihood	Normalized Log-Likelihood	BIC
N = 5	-163,758.7	-0.253	327,972.3
N = 10	**-21,304**	**-0.033**	**44,200.13**
N = 11	-21,731.71	-0.034	45,363.26
N = 9	-79,597.41	-0.123	160,506

These were our final results after training a univariate Hidden Markov Model with Global_active_power. We started off training with 5 states, and jumped up to 10 states when we saw the high number for the log-likelihood. When state 10 came back as a result close to 0 for the log-likelihood, we decided to train state 11. After training state 11, we found it strange that the log-likelihood for 11 states was getting further from 0. We attempted to look into the reason why, but have not been able to come up with a conclusion. Therefore, we decided that 10 states is the ideal number for the univariate Hidden Markov Model with Global_active_power.

Hidden Markov Models (Multivariate)

The third step was to train our model, and test it against the datasets with anomalies to try and detect anomalies through comparing the normalized log-likelihood.

For our multivariate Hidden Markov Model training using the variables Global_intensity and Global_active_power, this was the result of our trained model:

States	Log-Likelihood	Normalized Log-Likelihood	BIC
N = 15	-241,277.2	-0.373	486,021.6
N = 19	107,721.7	0.167	-210,120.6
N = 17	-228,520.4	-0.354	461,386.9
N = 18	**-213,221.3**	**-0.330**	**431,264.9**

As you can see in our results for our trained model, we started off with training state 15. It was a somewhat high number, so we went to train state 19. The result came back positive, so we went back to train state 17 and state 18. The log-likelihood was still in the negatives for state 18, and so we decided 18 states is the ideal number for our trained model.

Once our model was trained, we tested it against the first anomaly set to compare the normalized log-likelihoods. These were the results of the first test:

States	Log-Likelihood	Normalized Log-Likelihood	BIC
N = 18	**211,732.3**	**0.328**	**-418,621.6**

As seen in the results for the same state 18 in the first test model, it had a log-likelihood of 211,732.3, and a normalized log-likelihood of 0.328.

As for the second anomaly set, these were the results of the second test:

States	Log-Likelihood	Normalized Log-Likelihood	BIC
N = 18	**-920,195.3**	**-1.423**	**1,845,234**

As you can see in the table above for the same state 18 in the second test model, it had a log-likelihood of -920,195.3, and a normalized log-likelihood of -1.423.

For the third anomaly set, the results were closer to the results of the first test:

States	Log-Likelihood	Normalized Log-Likelihood	BIC
N = 18	**388,194.1**	**0.601**	**-771,545.2**

For the same state 18 in the third test model, it had a log-likelihood of 388,194.1, and a normalized log-likelihood of 0.601. A result much closer to the first test model, as compared to the second test model.

Normalized Log-Likelihood

Comparing the normalized log-likelihood of the trained model against all three of the test models yielded a promising result.

	Trained Model	Test Model 1	Test Model 2	Test Model 3
Normalized Log-Likelihood	**-0.33**	**0.328**	**-1.423**	**0.601**

As shown in the table above, the normalized log-likelihood for the trained model is -0.33. While the normalized log-likelihood for the three test models were 0.328, -1.423, and 0.601 respectively. Putting all of these normalized log-likelihoods together produced a striking difference. It is clear that the normalized log-likelihood for all three test models differ quite drastically from the trained model. The first and third normalized log-likelihood were in the positives which is undeniably different from the trained model. Meanwhile the second normalized log-likelihood was much higher than the trained model.

Although the normalized log-likelihood alerts us of the existence of anomalies as demonstrated, it does not show nor tell us where the anomalies are located in the dataset. It is for the purpose of finding this information that we will be graphing the linear and polynomial regression, as well as the moving average in order to pinpoint these anomalies.

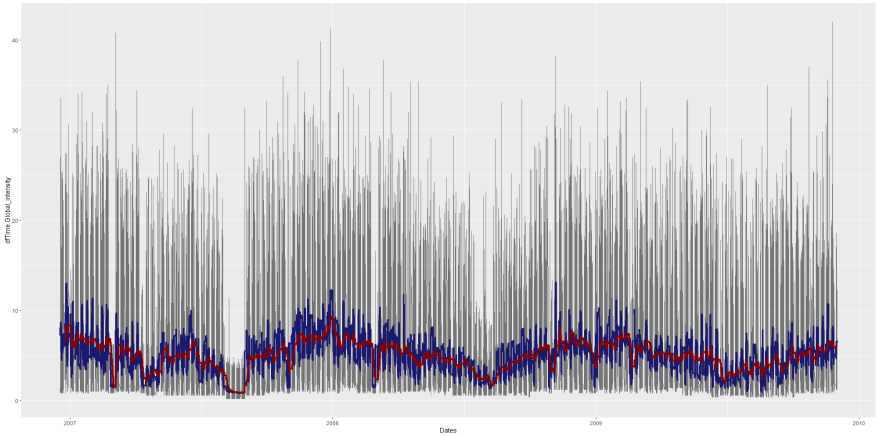
Moving Average

The first method we used to locate the anomalies was moving average. The blue line in the provided graphs is the moving average with a 12 hour moving time frame. The red line is the moving average with a 3 day moving time frame. Although it is difficult to make out the disparity between the test datasets and the training dataset due to the differing amount of data,

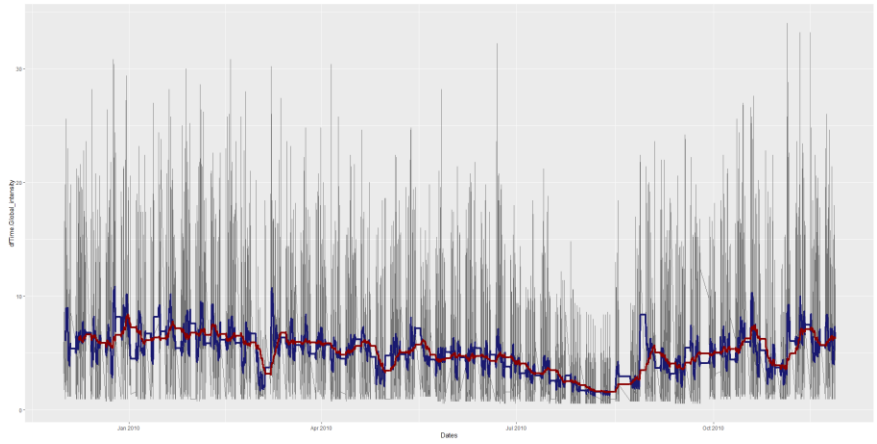
along with the time period, there are still some distinct differences to be found. A few examples of such is the max of each dataset as appeared on the y-value of the graphs.

These are the moving average graph results we received for Global_intensity and Global_active_power:

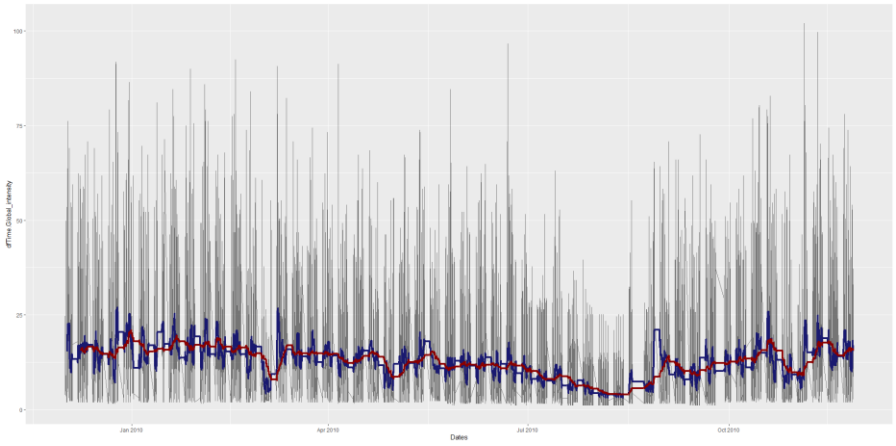
Training Dataset (Global_intensity)



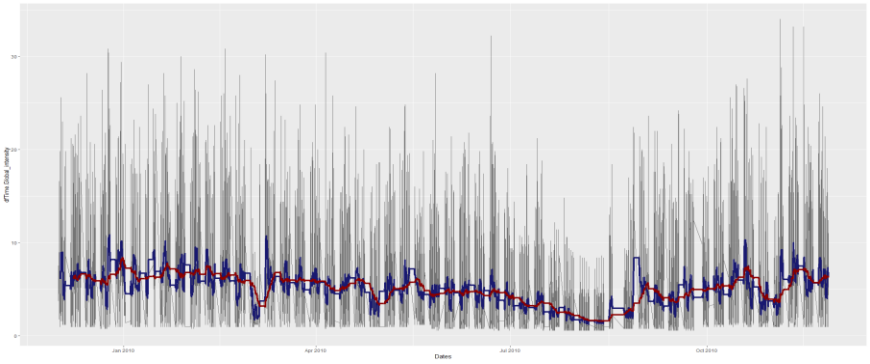
Test Dataset 1 (Global_intensity)



Test Dataset 2 (Global_intensity)



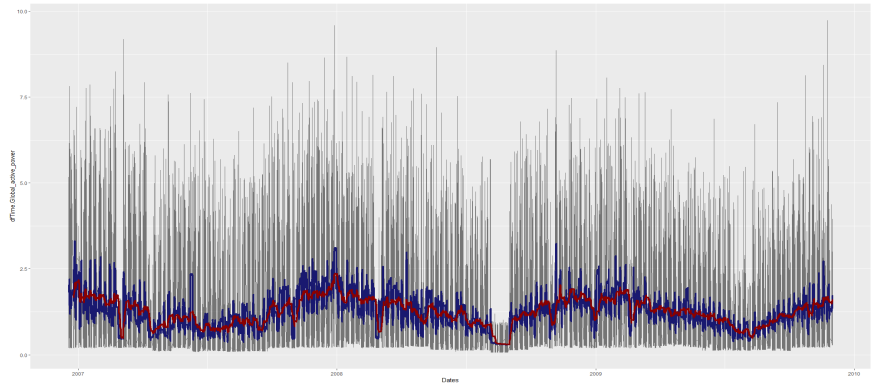
Test Dataset 3 (Global_intensity)



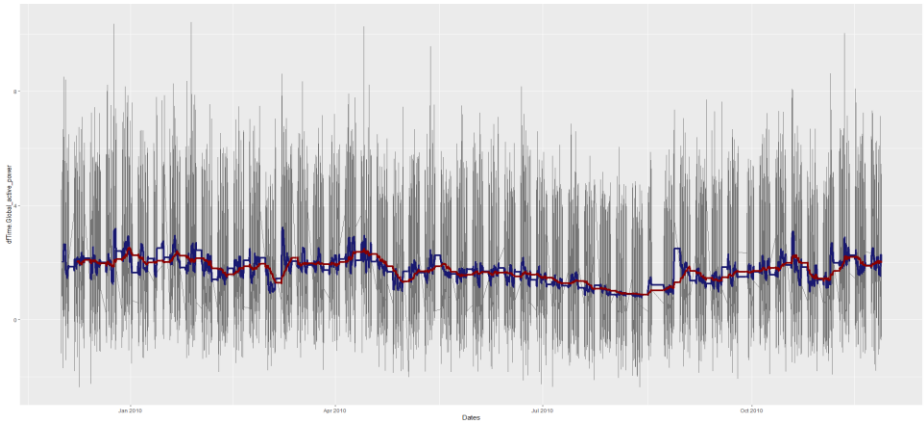
Global_intensity	Training Dataset	Test Dataset 1	Test Dataset 2	Test Dataset 3
Max	~42.5	~34.0	~103.0	~34.0

As shown in the table, the max Global_intensity for the training dataset was around ~42.5, while the test dataset 1 and 3 were ~34.0. With a difference of ~8.5, it is clear that there are some anomalies in the dataset, further supporting the existence of anomalies found through normalized log-likelihood comparison. Needless to say, with a max of ~103.0 for the test dataset 2, and a difference of ~60.5, it is without doubt that there are anomalies in the test dataset 2.

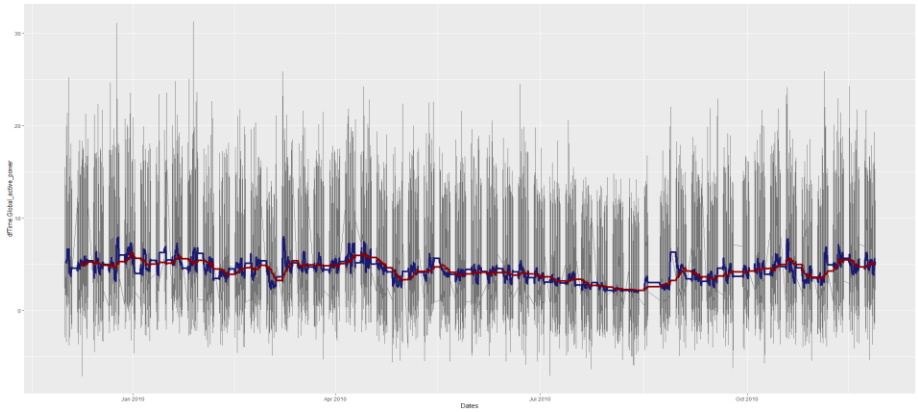
Training Dataset (Global_active_power)



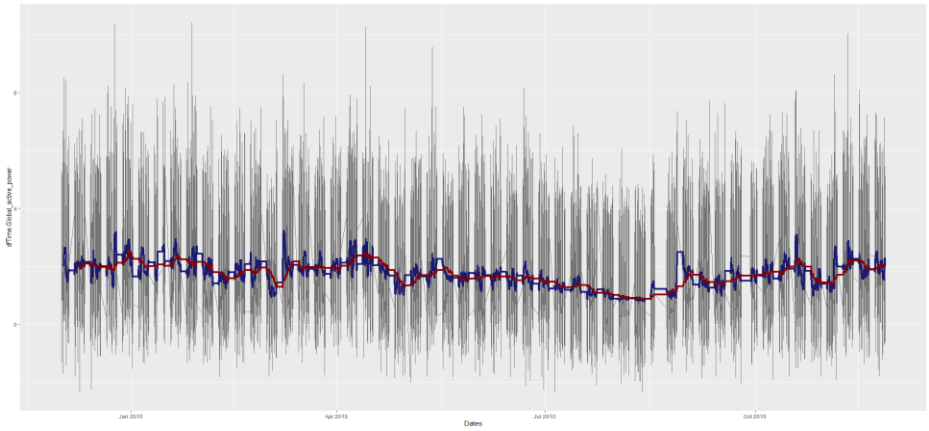
Test Dataset 1 (Global_active_power)



Test Dataset 2 (Global_active_power)



Test Dataset 3 (Global_active_power)



Global_active_power	Training Dataset	Test Dataset 1	Test Dataset 2	Test Dataset 3
Max	~9.7	~10.5	~31.0	~10.5
Min	~0.1	~ -2.3	~ -7.5	~ -2.3

However, for Global_active_power, the max for test dataset 1 and 3 were within a reasonable margin of error compared to the training dataset. The min on the other hand was in the negative range, a clear sign of anomalies. Once again for dataset 2, it is without doubt that there are anomalies due to a difference of ~21.3 in the max, and a negative min as compared to the training dataset.

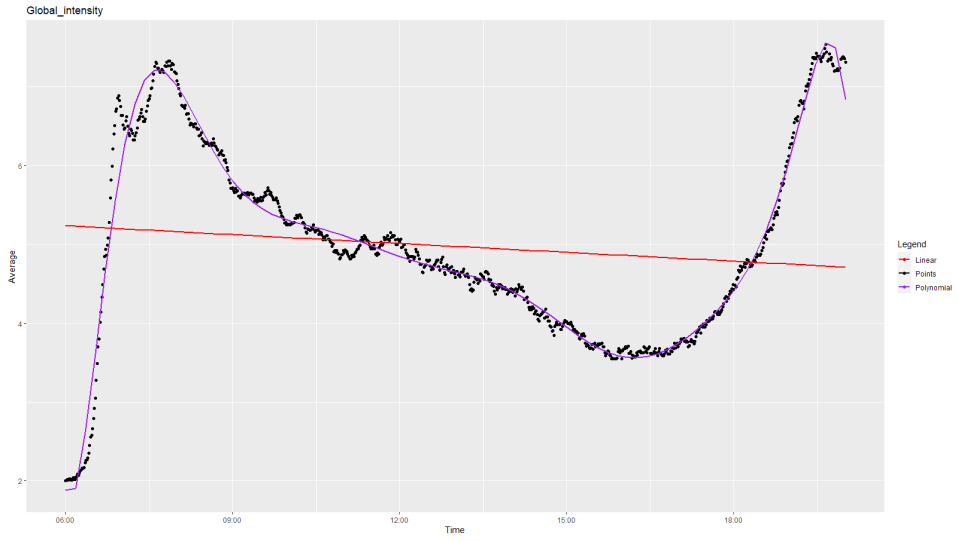
Although the moving average further supports the existence of anomalies, it was not the main goal of this method. Due to the high amount of data provided in the dataset and the wide range of dates, it is hard to make out the locations of the anomalies. Therefore, we cannot rely on the moving average to find the locations of the anomalies.

Linear/Polynomial Regression

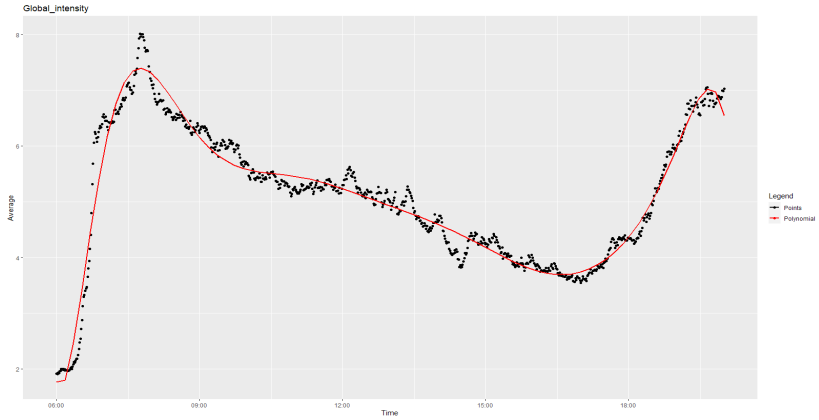
The second method we utilize in locating the anomalies will be with linear and polynomial regression.

As you will notice in the following graphs for both Global_intensity and Global_active_power, the data is not linear and thus linear regression would not be a good fit for either of the data patterns. Therefore, the linear regression line would be a terrible choice to use for finding outliers. However, the polynomial regression line with a degree of 11 fits the data pattern very well, and is a great way for detecting outliers in this scenario. These are the results of our linear and polynomial regression lines for our dataset on weekdays between 6:00 AM to 8:00 PM.

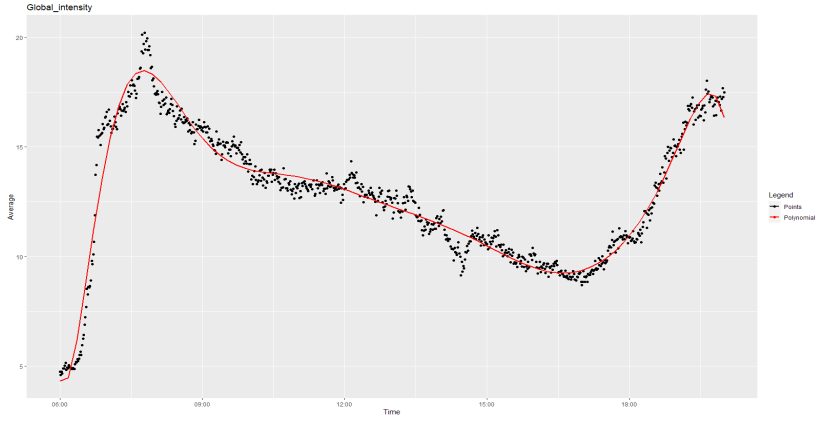
Training Dataset (Global_intensity)



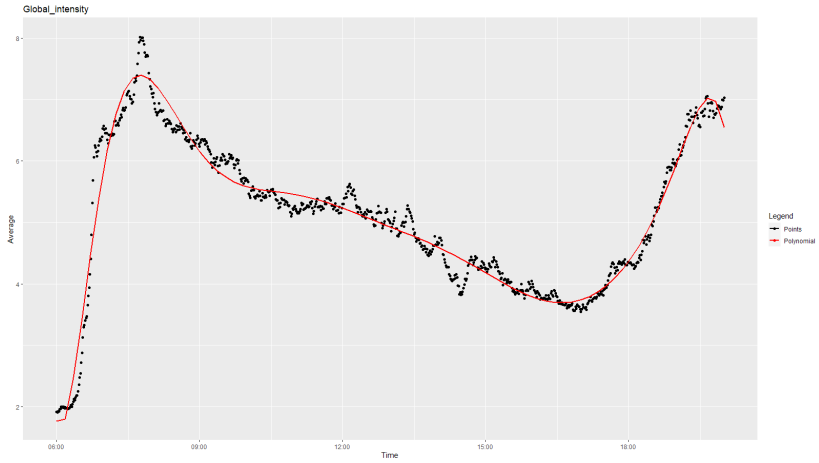
Test Dataset 1 (Global_intensity)



Test Dataset 2 (Global_intensity)



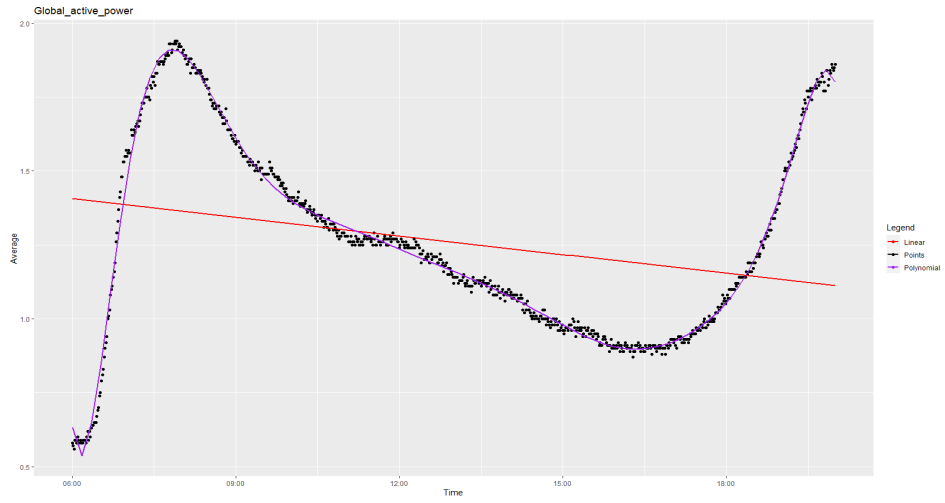
Test Dataset 3 (Global_intensity)



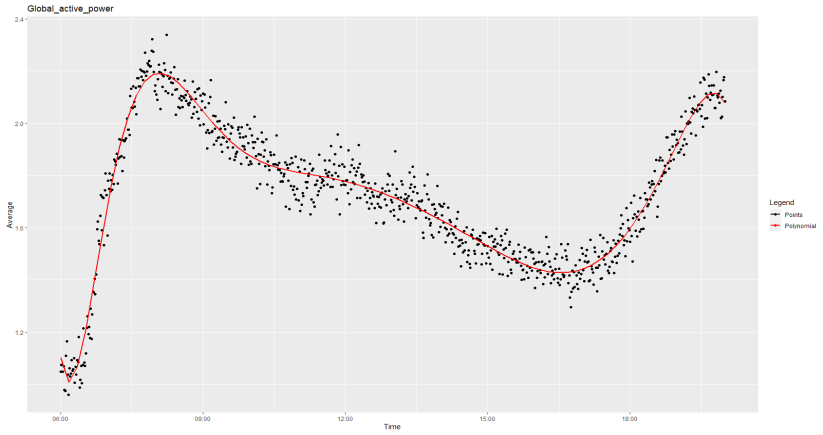
In all of the test datasets, there is a sudden spike in the Global_intensity average around 8:00 AM, an increase from ~7.4 to ~8.0 when compared to the training dataset. A reasonable increase that is clearly noticeable due to the fact that it strays far from the polynomial regression line. Needless to say, there are many more random instances of increase and decrease in the Global_intensity average throughout the graph. These instances are made further apparent through comparing the average Global_intensity value where both the training and test dataset graphs have a bump in common.

Meanwhile, comparison of the max and min Global_intensity average values do not seem to give much evidence in regards to the existence of any anomalies. Other than test dataset 2, test dataset 1 and 3 have very similar min values of ~1.9 as compared to the training dataset of ~2.0. The only noticeable difference is in the max values of ~8.0 compared to the training dataset of ~7.3, but this can be written off as just a coincidence.

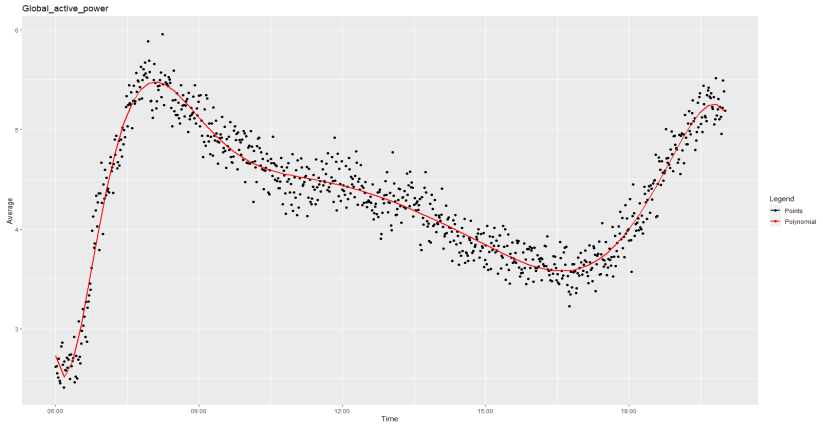
Training Dataset (Global_active_power)



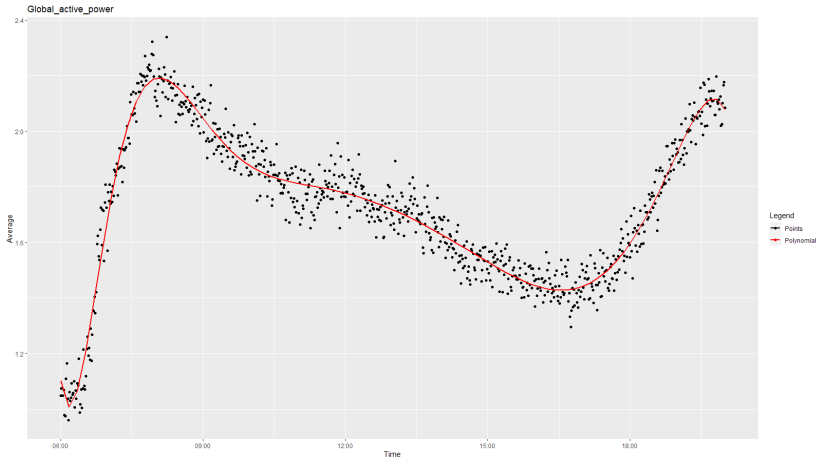
Test Dataset 1 (Global_active_power)



Test Dataset 2 (Global_active_power)



Test Dataset 3 (Global_active_power)



Meanwhile for the graphs above showcasing the linear and polynomial regression for Global_active_power, it is in a similar situation as Global_intensity. The data is not plotted in a linear fashion, thus rendering the linear regression line useless in this scenario. However, just as Global_intensity, the polynomial regression with a degree of 11 fits the training dataset very well, and creates a striking contrast with the test datasets.

In the training dataset, the averages of Global_active_power throughout the hours are very clustered and close to each other in value, with a differing value of ~0.01 to ~0.03 at most when compared to their neighbouring points. This is not the case when viewing the polynomial regression for all three test datasets. There is an obvious difference in the case that the averages of Global_active_power are scattered with a high differing range of ~0.1 to ~0.2.

Global_active_power	Training Dataset	Test Dataset 1	Test Dataset 2	Test Dataset 3
Max	~1.9	~2.3	~5.9	~2.3
Min	~0.6	~0.9	~2.3	~0.9

Furthermore, taking a look at the max and min values show yet another evident anomaly. The min and max values for all three test datasets are reasonably higher than the min and max values for the training dataset, with a minimum differing value of ~0.3. This is a great difference in the context of Global_active_power.

Through the comparison of the polynomial regression graphs in this section, it is clear to see that the polynomial regression graph precisely shows the location of the anomalies as well as sometimes supporting the existence of any anomalies.

Problems Encountered

When we first started the project, we had chosen our time frame to be between 6:00 AM to 8:00 PM, with nothing in regards to whether it was a weekday or a weekend. However, this caused our dataset to be extremely large, and we were unable to train the Hidden Markov Models due to insufficient memory, and long waiting times for the graph to be produced. This was the reason we decided to reduce our dataset further by reducing our time frame to be weekdays only on top of the original 6:00 AM to 8:00 PM.

The second problem we faced was in regards to choosing the variables to use from our Principal Component Analysis. We originally would have preferred to either use Global_reactive_power, Sub_metering_1, or Sub_metering_2 over Global_active_power, as they contributed to both PC1 and PC2, while Global_active_power only contributed to PC1. When we attempted to train our Hidden Markov Models for Global_reactive_power, Sub_metering_1, and Sub_metering_2, we faced many errors in the training process.

After many tests were performed for Global_reactive_power, we realized the issue with training the Hidden Markov Model was due to the numbers under the column that were of either value 0, or a value extremely close to 0. We fixed this problem by removing all of these values from the columns. It gave us a reasonable sized dataset that we could still work with. However, this is where we came across another issue. Training the univariate Hidden Markov Model for Global_reactive_power with state 5 was already resulting in a positive log-likelihood. All the way down to state 2, it continued to be a positive number, and we chose to not use Global_reactive_power as our second variable.

For Sub_metering_1, and Sub_metering_2, we faced a similar problem to the beginning of Global_reactive_power. There were numbers under the columns that were of either value 0, or a value extremely close to 0 that prevented us from training a Hidden Markov Model. Removing those values also fixed the problem of training the Hidden Markov Models. However, there were many more values under these columns that were removed from the dataset, resulting in a dataset that we believed to be too small to give accurate results.

It was for these reasons that we decided to pick the next best variable according to our PCA results, which was Global_active_power. When training the Hidden Markov Models for Global_active_power, we did not receive any errors that were apparent in Global_reactive_power, Sub_metering_1, or Sub_metering_2, so we decided to finalize Global_active_power as our second variable to use.

Lessons Learned

Throughout the course of this project, we learned many things regarding anomaly detection, including different methods and approaches to the problem, as well as how to interpret the data gathered. We learned how to fix problems, perhaps avoid them if they cannot be fixed, and how to approach the problem with many different perspectives. We learned the importance of cybersecurity and the difficulties cybersecurity engineers face when attempting to detect anomalies. There is much data to check and look out for when aiming to identify anomalies.

Conclusion

This section at the end of the report is to summarize our findings and give a conclusion regarding our results. At the beginning, through the use of Principal Component Analysis, we

determined that the two variables to use are Global_intensity and Global_active_power. After further testing with univariate and multivariate Hidden Markov Model training for these variables, we came to the conclusion that they were usable data, and created a trained model.

The next step we took was to find the existence of anomalies in respect to the trained model. The method we applied was comparison of the normalized log-likelihood which supported the existence of anomalies in the dataset.

With this result in mind, our next step was to locate these anomalies. The first method we used was moving average. However, the results we gathered did not give us any information on the location of the anomalies. It only served to further support the existence of any anomalies. The second method we used was linear and polynomial regression. Linear regression did not end up being a reliable method, however polynomial regression showed promising results. It clearly revealed the locations of the anomalies through the use of the time frames. On top of that, polynomial regression was able to serve as further evidence supporting the existence of any anomalies; but only for Global_active_power.

It was through all of these steps that we detected the existence of anomalies, and located them in the dataset; through the use of the time frame. This concludes our project report on our findings with anomaly detection in the provided dataset.

4. Technical Essay

Reinforcement Learning and Its Use in Intrusion Detection

Abstract

This paper will describe the basic principles of Reinforcement Learning, including the agent, states, and rewards. The paper will also detail the Markov Decision Process for creating a suitable training environment for such an agent. As well as the learning algorithm, this paper will describe the use Reinforcement Learning has in online intrusion detection, specifically anomaly detection. Although this essay cannot provide a comprehensive view at all features of Reinforcement Learning, it should provide a suitable look at its foundational aspects.

1 Introduction

In the modern age, humans are accustomed to the convenience of having access to vast stores of information at our fingertips. This is achieved using various advanced technologies which connect the world, through both the transfer of data and the transportation of materials. However, as society has advanced, it has grown increasingly difficult to improve and protect the world's vital infrastructure through only the use of the human mind. In order to refine and safeguard these systems, researchers are turning to an AI training process known as Reinforcement

Learning. This process is fundamentally the same as the operant conditioning process used on biological organisms, in which the subject is rewarded for correct actions, and is punished for incorrect actions. In the machine world, these actions typically take the form of selecting a correct choice out of a massive dataset (Koduvely, 2019), for which the machine is rewarded. This essay will outline the principles of Reinforcement Learning, the Markov Decision Process, and their use in intrusion detection.

2 Reinforcement Learning

At the most basic level, Reinforcement Learning is simply another method through which researchers try to design a machine that is capable of acting rationally and humanlike (Sewak, 2019). However, this method is considered the most advanced as it creates an agent that is able to manipulate the environment in order to achieve the best possible outcome; essentially what the brain performs in an intelligent life form such as humans. This process takes the form of a loop, in which the agent is presented with a state, it chooses an action, a reward is distributed, and the agent is presented with a subsequent state. The goal of the agent in this scenario is to gain the maximum reward possible through every state. This figure, taken from Sewak's book "Deep Reinforcement Learning", illustrates this.

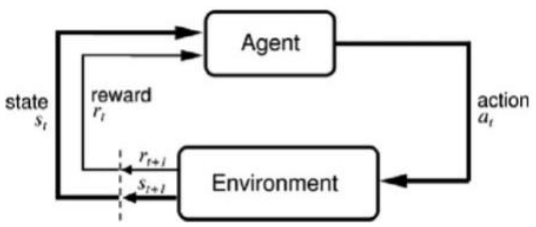


Fig 4.1 Action-State loop

2.1 The Agent

Due to the basic goal of creating a rational thinking machine, the agent is identifiably the most important part of the Reinforcement Learning process. The agent is simply the actor that contains the knowledge and ability required to complete a task with a sequence of actions in the given environment, which could be a video game, dataset, or a simulation of a real-world scenario. In training an agent, the goal is to create an optimal, or near-optimal policy that would yield the maximum cumulated rewards (Zhao, 2015). There are many different functions for creating an agent's policies, such as Q-learning, SARSA, and Temporal Difference Learning, however, these functions all serve the same basic purpose of creating policies that yield the highest rewards.

2.2 The State

In this process, the state of the environment is a scenario that the agent would likely encounter while it is deployed, such as an intersection on a road, for example. During training, this would be a simulated intersection where any action the agent takes would have no impact on the world. Encompassed in the state are all the possible factors that the agent could recognize to be impactful, such as other cars, pedestrians, and traffic lights in our scenario. All other factors, such as colour, would be regarded as noise. Ideally, a simulation would represent all the necessary factors and ignore all possible noise, but this is not always possible due to uncertainties in the real-world. Once the agent has perceived and measured all the relevant aspects of the state, it will then perform the action that it deems "best" under the current state, and would then be either rewarded or penalized by the environment based on both the action taken and the state.

2.3 The Reward

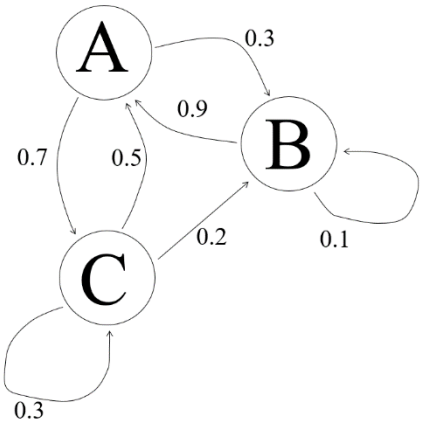
For Reinforcement Learning, the reward generally takes the form of points, and it is possible for the reward to be negative, which can also be thought of as a penalty. Although the concept seems simple, there are many challenges in choosing an ideal reward function for the agent, and this section will detail two of these. The first challenge is that of rewards that cannot be immediately realized by the agent. As an example, consider an agent tasked with choosing between exercise and sleep. At the immediate state, sleep would obviously be the more rewarding action, as it grants the agent rest, whereas exercise will take away energy and possibly cause discomfort. However, over the period of several weeks, months, or years, the reward from sustained exercise could be far greater than from sleep. Another issue that needs to be addressed is uncertain rewards. Taking the same example of exercise and sleep, let us consider an agent with an unknown heart condition. In this situation, the potential penalty of exercise could be infinite, whereas sleep would be rewarding. Therefore, the rewards from the current action could be probabilistic, and it is a problem for the designer to dictate a function that mitigates these issues to the greatest extent possible.

3 Markov Decision Process

In order to train the agent with states and rewards, one must first create a suitable environment for the agent. In Reinforcement Learning, the environment is simply the congregation of all the states and the rewards given for an action at each state. Although the basic idea behind an environment is fairly simple, there are very clear challenges that must be overcome, such as factor representation and reward functions. To construct an objective environment, a Markov

Decision Process (MDP) is used. This process, which is a form of a Markov Chain, is behind nearly every Reinforcement Learning algorithm and forms the foundation for the transition from state to state.

Fig 4.2 Markov Chain



3.1 The Markov Property

For an agent to make objective and rational decisions at each state, it must be presented with the consequences of its actions at that state alone. To achieve this, the Markov Property, the basic principle of the Markov Chain, is employed. The Markov Property is a memoryless property, in that the probability the chain transitions from one state to the next is solely dependent upon the state it is currently in. This property is generally used for modeling random systems, in which a sequence of linked events occurring over time tend to only depend on the present state. This allows for more rational predictions in real-world applications since the conditional probability distribution of each state is unchanging.

3.2 The Training Environment

In training an agent with Reinforcement Learning, the Markov Chain is desirable as it is able to create an agent that does not consider its past actions. As an example, consider a car at an intersection, it would have the same conditional probability distribution regardless of whether it had made an emergency stop or had slowly drifted forward. This allows the agent to not be influenced by past rewards as it tries to choose the best action at the current state. This is necessary as an agent might encounter endless different combinations of states in deployment, and if it does not employ the MDP, it would likely be unable to decide on the best action at each state. To create a wholly rational agent, it cannot be burdened by the rewards or penalties that it had encountered in the past, as it is unlikely that the same action would lead to, or follow from the same event chain. Instead, the agent must decide on the action that it deems the best only under the current situation, which should in theory lead to the maximum possible reward.

4 Intrusion Detection

As cyber attacks have grown in sophistication, signature-based detection is no longer viable as the sole method of protection as zero-day exploits become more common. Instead, behaviour-based detection is used to determine whether a system has been infiltrated and is relaying data to unauthorized parties. However, due to the ever-changing nature of attacks, it is necessary to use an AI agent to discover patterns that a human cannot see. To achieve this goal, an agent needs to be in constant evolution in order to not be outpaced by different methods of data extraction. To this end, a technique known as online learning is employed.

4.1 Online Reinforcement Learning

In online learning, an agent is fed the results of its past predictions, with more significance given to more recent predictions (Koduvely, 2019). It is clear then, why Reinforcement Learning would be the method chosen for online tasks. In these prediction scenarios, each time the agent must make a prediction, it can be seen as a state, and the result would then be the reward for the agent's choice. As an agent is used to make constant predictions of anomalous data, it would learn to choose the action that yields the highest total reward, in this case, the correct choice. Given enough time, an agent can learn to create a near-optimal policy, in such a situation, this would mean near-perfect predictions.

4.2 Anomaly and Intrusion Detection

Although online Reinforcement Learning can be used to predict events such as the weather, purchasing tendencies, and the economy, it is most valuable in the field of cyber defense. In this domain, it is often not possible for humans, or regular online learning algorithms, to detect anomalous behavior. Due to this fact, an agent that has been trained with countless different scenarios is needed. In using such an agent, it is possible to make the most correct, or most rewarding, decision in whether an anomalous behavior exists in a dataset. This agent would also be able to decide what data points are anomalous and which are simply slight differences in normal behaviour. To a human, these subtle differences may appear overly similar, but an AI with optimal or near-optimal policies can decide with relative confidence if defensive measures need to be taken. Once an agent has reached a point in training when it is deemed to have a near-optimal policy, it should in theory be able to determine whether an unauthorized person has

gained access to a secure system. The agent can then flag this as intrusive behaviour and block the actor's activity on the system.

5 Conclusion

This paper has provided a brief overview of Reinforcement Learning and its use in intrusion detection. Despite the fact that it only touches on the surface of Reinforcement Learning, it should provide a general overview of what this method is, how the different parts are interlinked, and what its value is to protecting sensitive data. As a summary, Reinforcement Learning is an algorithm in which an agent is tasked with making actions at different states in an environment. These actions are then provided with a positive or negative reward based on their correctness. To ensure rationality, the environment is constructed with the Markov Property in mind, with each state only being dependent upon itself. In this way, an agent can be trained with vast datasets to predict normal behaviour, and flag anomalous behaviour with relative correctness. Although there are numerous challenges in creating a perfect policy for an agent, given enough time, this technology should be able to drastically reduce the damage of cyber attacks.

6 References

Chandola, Varun, Banerjee, Arindam, & Kumar, Vipin. (2009). Anomaly detection. *ACM Computing Surveys*, *41*(3), 1–58. https://doi.org/10.1145/1541880.1541882

Dongbin Zhao, & Yuanheng Zhu. (2015). MEC-A Near-Optimal Online Reinforcement Learning Algorithm for Continuous Deterministic Systems. *IEEE Transaction on Neural Networks and Learning Systems*, *26*(2), 346–356. https://doi.org/10.1109/TNNLS.2014.2371046

Haviv, A. (2020). Technical Note—Cyclic Variables and Markov Decision Processes. *Operations Research*, *68*(4), 1231–1237. https://doi.org/10.1287/opre.2019.1913

Koduvely, D. (2019, February 07). Anomaly Detection through Reinforcement Learning. Retrieved November 23, 2020, from https://zighra.com/blogs/anomaly-detection-through-reinforcement-learning/

Osiński, B. (2020, July 23). What is reinforcement learning? The complete guide. Retrieved November 24, 2020, from https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide/

Pang, Guansong, Hengel, Anton van den, Shen, Chunhua, & Cao, Longbing. (2020). *Deep Reinforcement Learning for Unknown Anomaly Detection*.

Peng Guan, Raginsky, Maxim, & Willett, Rebecca M. (2014). Online Markov Decision Processes With Kullback-Leibler Control Cost. *IEEE Transactions on Automatic Control*, *59*(6), 1423–1438. https://doi.org/10.1109/TAC.2014.2301558

Sewak, M. (2019). Deep Reinforcement Learning. In *Deep Reinforcement Learning*. Springer Singapore Pte. Limited. https://doi.org/10.1007/978-981-13-8285-7

Shalev-Shwartz, S. (2012). Online Learning and Online Convex Optimization. In *Foundations and trends in machine learning* (Vol. 4, Issue 2, pp. 107–194). Now Publishers. https://doi.org/10.1561/2200000018

Su, Hanguang, Zhang, Huaguang, Zhang, Kun, & Gao, Wenzhong. (2018). Online reinforcement learning for a class of partially unknown continuous-time nonlinear systems via value iteration. *Optimal Control Applications & Methods*, *39*(2), 1011–1028. https://doi.org/10.1002/oca.2391

5. Contributions

Alexander Wang:

- Technical essay
- Project report structure and formatting
- Essay section presentation

Alexis Lazcano:

- Provided data used in the report
- Project report presentation

Jason Leung:

- Writing of project report and data analytics
- Responsible for Q&A