Anomaly Detection for Cybersecurity and Critical Infrastructure Protection

CMPT318 - Special Topics in Computing Science - Cybersecurity

D100, Fall 2020

Christofer Calvin Kurniawan 301335375

Seno Adhi Muhammad 301354133

Francis Wan 301351323

Abstract

Supervisory control systems need to be implemented to safeguard the critical services from persistent threats. Anomaly detection is part of the supervisory control systems and it is a measure to detect suspicious and unusual instances in a group of data. This project will explore anomaly based intrusion systems using univariate and multivariate Hidden Markov Models.

Table Of Contents

Table of Figures

1. Introduction

1.1. Preface

In 2003, a widespread power outage occurred throughout the parts of Northeastern and Midwestern United States and the Canadian province of Ontario causing 50 million people to lose power for up to two days cementing this incident as the biggest blackout in North American history. This incident known as the Northeast blackout of 2003 contributed to at least 11 deaths and an estimated $6 billion in damages. The blackout's predicted cause is a software bug in the alarm bug failing to inform system operators a fault has occurred. This incident caused a cascade of failures and ultimately a blackout.

In the modern day, the advancement of technologies has provided more efficient and effective automated control to national critical infrastructures. Automation is essential for monitoring and operating critical infrastructure to have continuous operation. Protecting critical infrastructures is paramount to the protection of our nation. Critical infrastructure uses SCADA control systems containing computers, networked data communications, and other peripheral devices to maintain continuous operation and real time control. In the modern day, cyber criminals threaten vulnerabilities of automation using cyberattacks on existing infrastructures to gain, destroy, or manipulate information and disrupt services for personal gains. In defence, critical infrastructures have implemented a supervisory control system allowing for continuous analysis of current activity in a method known as anomaly detection. Anomaly detection analyzes expected normal behaviour and non expected behaviour as anomalies. Using anomalies to detect harmful activities systems can help mitigate or prevent cyber attacks.

1.2. Objective

In this project we will explore and understand anomaly based intrusion detection systems using univariate and multivariate HMMS on normal electricity consumption data.

1.3. Data

The dataset used for this project is a sample dataset extracted from supervisory control data describing electricity consumption for households. Each variable and its description can be found in the following:

Date: date in format dd/mm/yyyy

Time: time in format hh:mm:ss

Global_active_power: household global minute-averaged active power (in kilowatt)

Global_reactive_power: household global minute-averaged reactive power (in kilowatt)

Voltage: minute-averaged voltage (in volt)

Global_intensity: household global minute-averaged current intensity (in ampere)

Sub_metering_1: energy sub-metering No. 1 (in watt-hour of active energy). It corresponds to the kitchen, containing mainly a dishwasher, an oven and a microwave (hot plates are not electric but gas powered).

Sub_metering_2: energy sub-metering No. 2 (in watt-hour of active energy). It corresponds to the laundry room, containing a washing-machine, a tumble-drier, a refrigerator and a light.

Sub_metering_3: energy sub-metering No. 3 (in watt-hour of active energy). It corresponds to an electric water-heater and an air-conditioner.

`

2. Methodology

2.1. Data Preprocessing

PCA forms the basis of multivariate data analysis. It is used in this project to analyze many variables in the data frame. It is very useful in simplifying the dataset by extracting the important information from the data and reducing the number of variables which are called principal components. In result, observing trends, jumps, clusters, and outliers is more simplified and faster.

After data preprocessing, the process is called mean-centering. In the previous assignment, a basic moving average method is chosen where the average of n points in the graph is calculated. Moving average on this project worked well because there is a lot of data in the datasets as the method does not work well in small datasets. It also works on non-linear dataset. However, depending on the number of states being used, some values on the start and the end are not available. In PCA, the average of all variables in each week is also calculated. By calculating the weekly average, the DataFrame in “TermProjectData.txt” becomes only 155 rows from 183722 rows in the given specific day and time. With this method, the dataset can be simplified by reducing the rows into weekly averages.

After mean-centering, the prcomp() library is used to find the PCA of the data. The function takes a numeric matrix or, in our case, is the dataframe of all variables, with the scale is True. Since it is a library, it does the calculation on itself.

For the result, check the summary of the PCA by using summary(pca). It presents the importance of components such as standard deviation, proportion of variance, and cumulative proportion for the 7 Principal Components.

```
> summary(pca)
Importance of components:
                          PC1    PC2    PC3     PC4     PC5     PC6     PC7
Standard deviation     1.8519 1.1222 0.9760 0.83193 0.62481 0.49043 0.1890
Proportion of Variance 0.4899 0.1799 0.1361 0.09887 0.05577 0.03436 0.0051
Cumulative Proportion  0.4899 0.6698 0.8059 0.90477 0.96054 0.99490 1.0000
```

Fig 2.1.1 PCA summary for the 7 principal components

The summary in Fig 2.1.1 only shows a bunch of numeric values of standard deviation, proportion of variance, and cumulative proportion. To create a better visualization, the data above is converted to a PCA biplot.

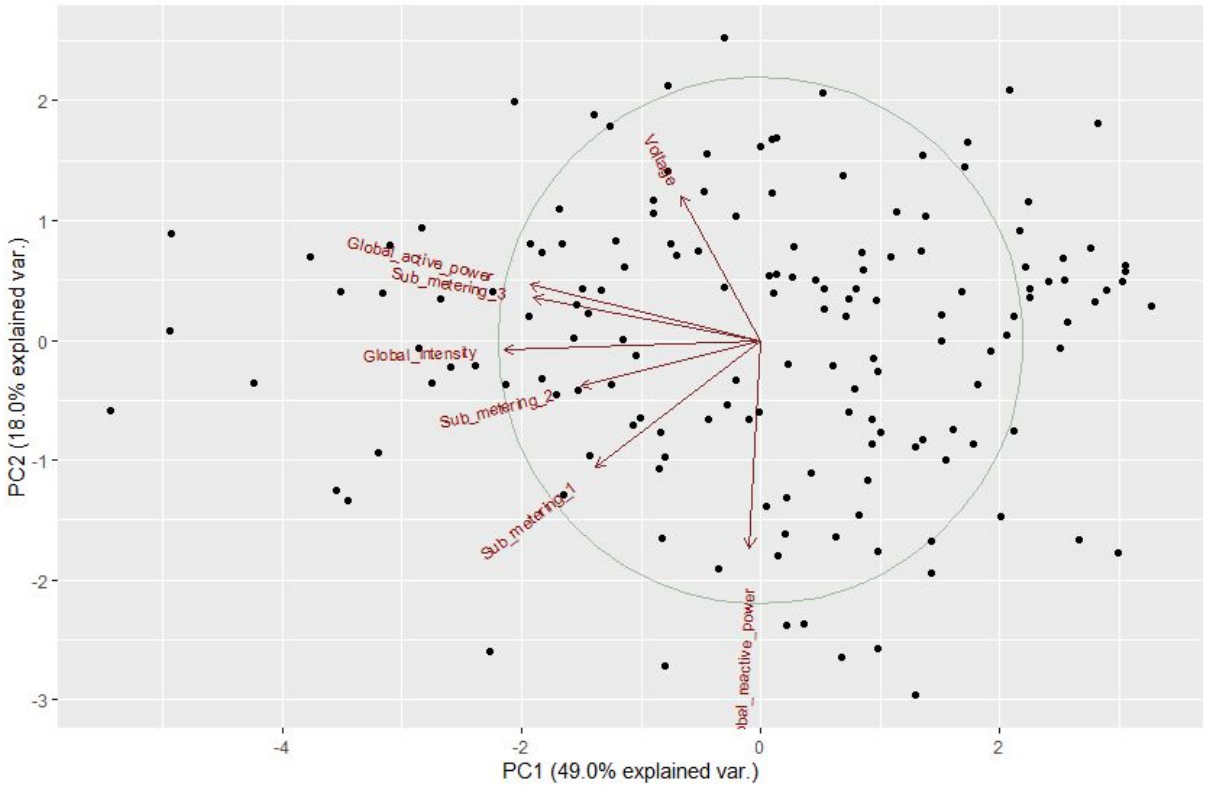

Fig 2.1.2 PCA diagram

The biplot (Fig 2.1.2) is the combination of the score plot and the loading plot. The score plot is the one shown with the black dots, while the loading plot is the vector lines. The biplot shows the relationship of the attributes to the PCs. The vector line direction shows the amount of influence it has on the PCs. For example, Global_intensity shows it has a strong influence on PC1, while Global_reactive_power has more influence on PC2. On the other hand, the angles between the vector line represents the correlation between them. For instance, Global_active_power and Sub_metering_3 have a very strong correlation since they face the same direction with a very small angle. Meanwhile, Voltage and Global_reactive_power have a strong negative correlation as they both diverge and form a large angle

	PC1	PC2
Global_active_power	-0.9038028	0.18484451
Global_reactive_power	0.0783643	-0.80363895
Voltage	-0.4057970	0.42699063
Global_intensity	-0.9759954	-0.06871028
Sub_metering_1	-0.6210855	-0.52999516
Sub_metering_2	-0.7302760	-0.10910557
Sub_metering_3	-0.8919698	0.08139208

Fig 2.1.3 Components Correlation Score to PC1 and PC2

Since PC1 and PC2 have percentage variance of 49% and 18% respectively, both principal components are chosen to represent our dataset. In the biplot above, the Global_active_power, Global_intensity, and Sub_metering_3 have the highest value as can be seen in Fig 2.1.3, with values of -0.904, -0.976, and -0.892. This means that the three variables have a strong negative influence on PC1. Therefore, the three variables are chosen for the hidden markov model in the next part.

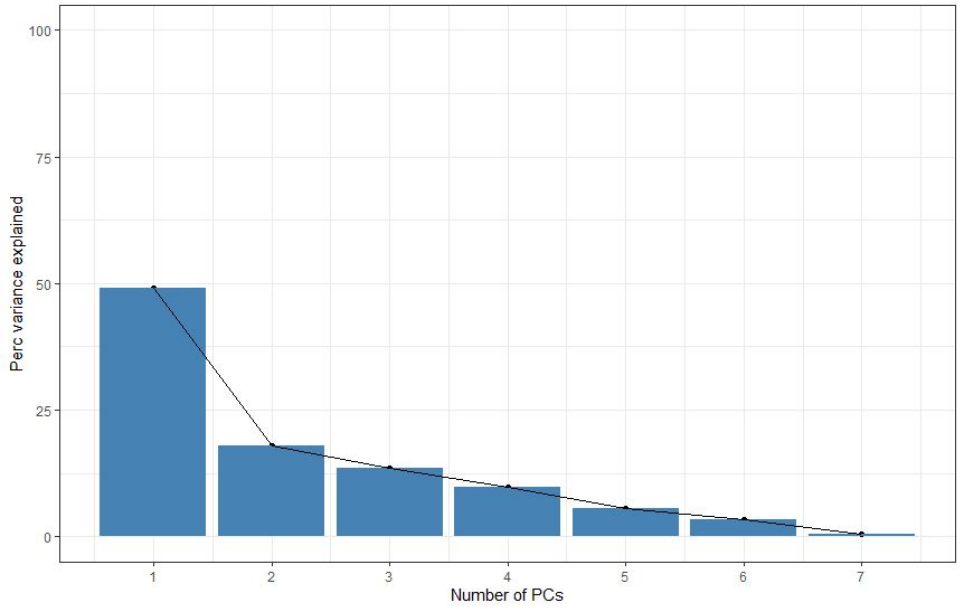

Fig 2.1.4 Scree plot

The scree plot is a line plot of the eigenvalues of the PCA. The main focus of Scree plot is to display how much variation each Principal Component captures from the data. The curve of the scree plot in this dataset is called “elbow” which means that the first couple Principal Components are already sufficient to describe the essence of the data. In this Scree plot, PC1 has the highest percentage of variance explained (~49%) which means it has the highest eigenvalue. The second PC2 has about 18% variance. Both PC1 and PC2 combined resulted in around 70%, which means that PC1 and PC2 explains 70% of the data.

2.2. Variable Selection

In this project, the chosen variables detect anomalies are global active power, global intensity, and sub metering 3.

Global active power is the real power consumption, it is produced by counting the power consumption of electrical appliances used by humans. Things like TV, computer and refrigerator contribute to global active power.

Global intensity is the ratio between energy supply to GDP. Global Intensity is used to measure the efficiency of the economy of a country and it can be said that high intensity means high industrial output.

Submetering is used to measure the utility usage of a place (e.g. house, condominium, apartments), in this case air-conditioner and water heater.

From the definition of these terms, these three variables are somewhat correlated. A biplot was generated to determine the most suited variables. The arrows in the diagram indicate Global active power, Global intensity and Sub metering 3 are the most important variables in the dataset since they are closely located with each other.

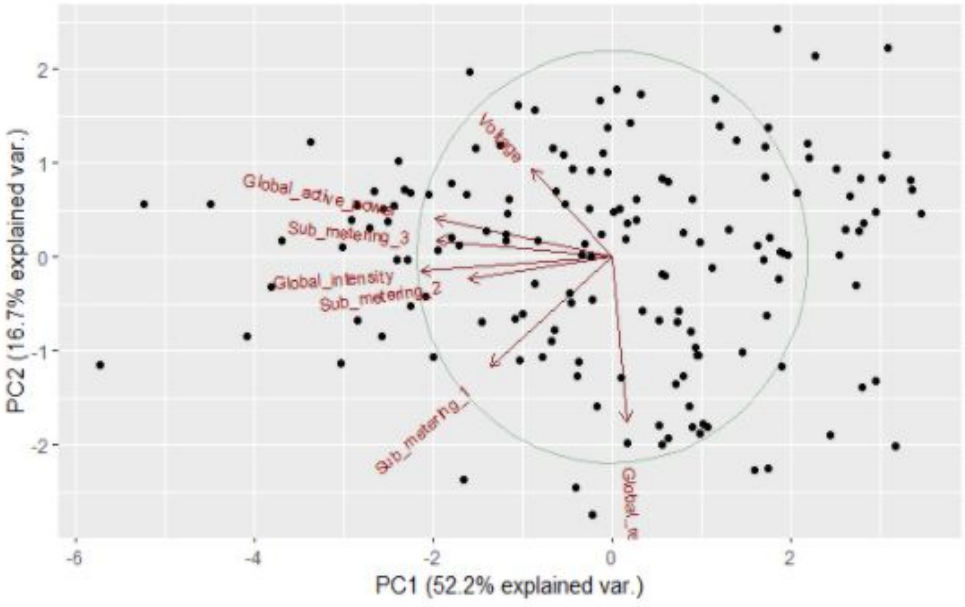

Fig 2.2.1

PCA diagram

2.3. Time Frame Selection

To ease the anomaly detection process, a pattern of the data points in a given time frame must be recognizable. For instance, if there is a decreasing trend in a particular time window, then the data points that go against the trend might be considered as anomalies. To get the idea about the pattern of the data, three variables from the data frame were plotted into a diagram and results in this figure 2.3.1.

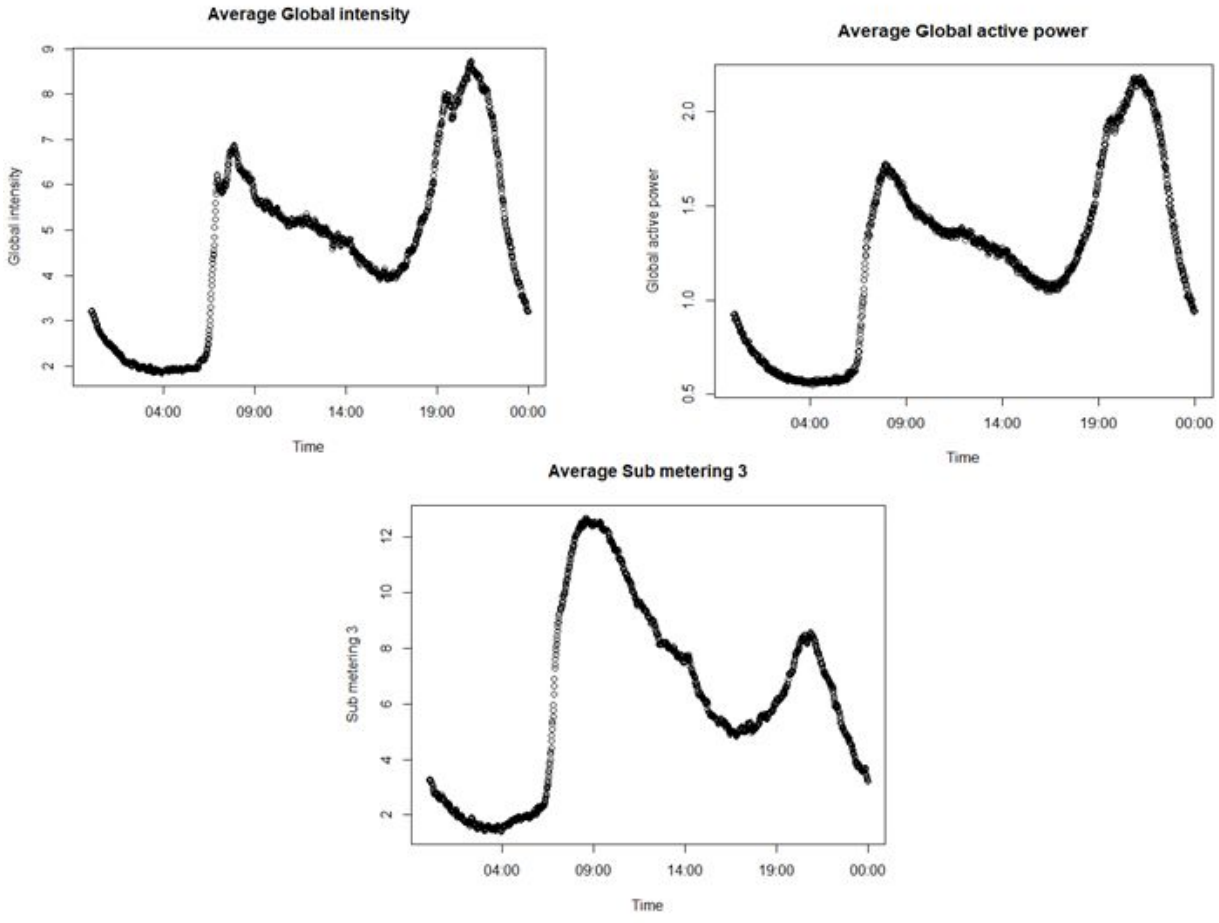

Fig 2.3.1 Average of values in a day

From the figures above, global active power, intensity and sub metering decrease in the afternoon, in particular from 12 pm to 4 pm. The decreasing electricity usage in these times are caused by human activity in the afternoon, where the majority of people are done doing their activities in the afternoon. This particular time frame is interesting as the pattern is recognizable and is making the anomaly detection process easier.

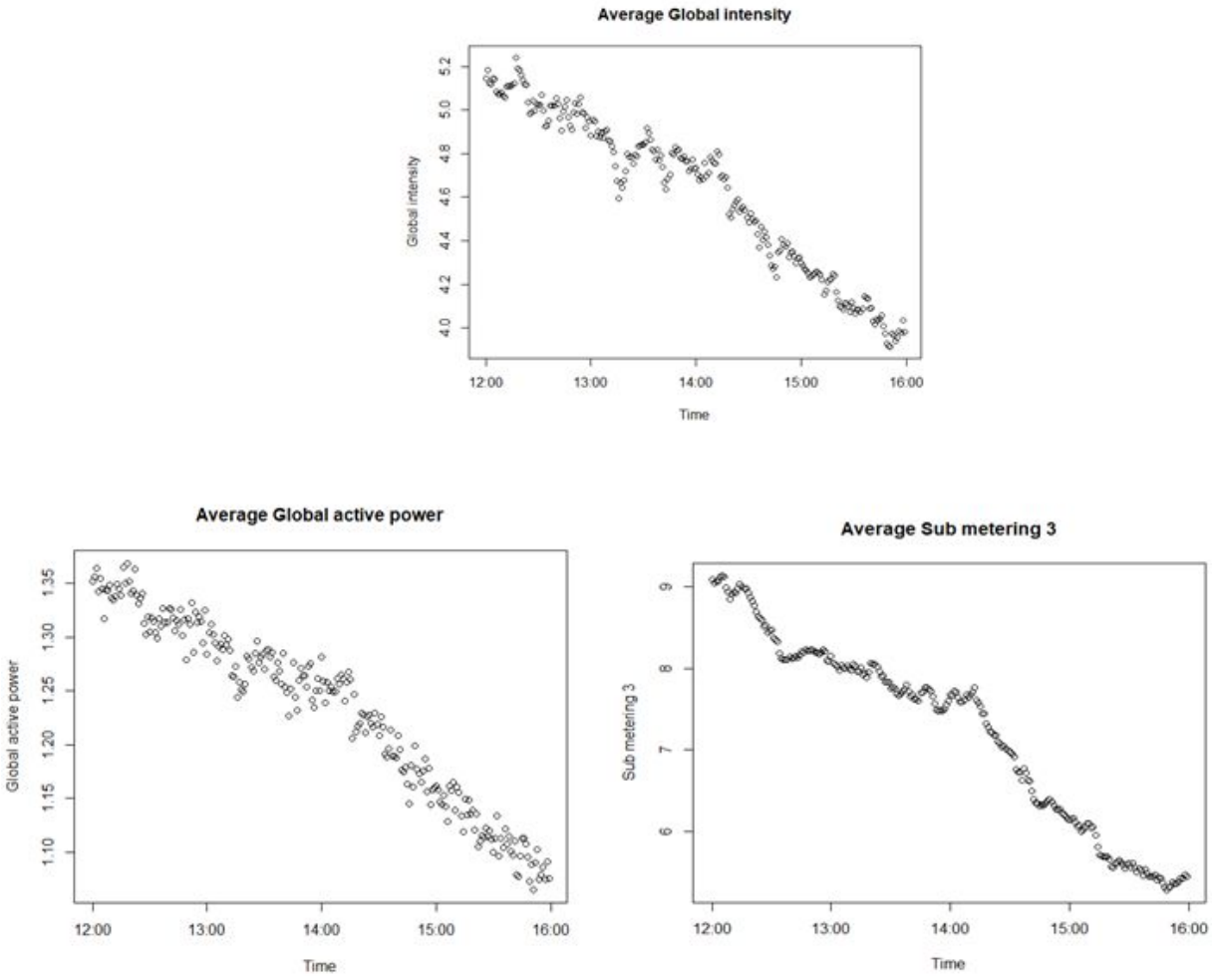

Fig 2.3.2 Variables in the respective time window

2.4. Training and Testing HMMs

The given dataset was split between training and testing data, where training data consists of around 70% of the original dataset and the rest goes to the testing dataset. To be more specific, the data recorded after Jan 1st 2009 was assigned to the testing dataset.

The univariate HMM was trained on different numbers of states to get a variety of results so that they can be compared with each other. For instance, the model that was trained with 5 states has a log-likelihood of -8.0610 and BIC of 174.9981. Whereas the univariate model with 10 states has 22.2035 and 511.6597 for the log-likelihood and BIC respectively.

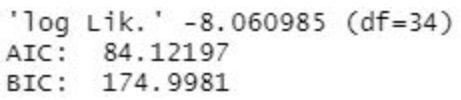

Fig 2.4.1 Univariate 5 states

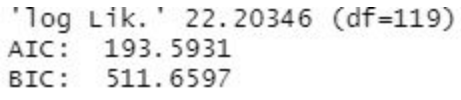

Fig 2.4.2 Univariate 10 states

The preferred model is the one with low BIC or complexity and high log-likelihood. In this case, the model with 5 states is ideal because the complexity is lower than the 10-state model and the difference between both model's log-likelihood is not too big. Choosing a model with lower complexity helps to avoid overfitting in the model.

The multivariate models were also trained on the same number of states, 5 and 10. At state = 5, the model has a log-likelihood of -319.2538 and 890.8404 complexity. A much higher complexity was gained from the model training with 10 states, where 1185.85 is the value of BIC and -221.4353 is the value for its log-likelihood.

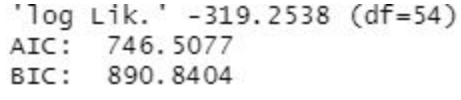

Fig 2.4.3 Multivariate 5 states

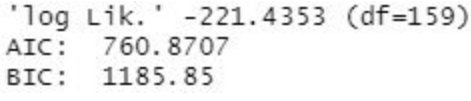

Fig 2.4.4 Multivariate 10 states

From these results, state 5 is better because the BIC is lower and the log-likelihood is not much of a difference as it is still reasonable, considering the huge complexity difference.Thus, the final choice of the number of states for both univariate and multivariate training models is 5 states.

The univariate model was tested using the test dataset and it resulted in 6.3711 log-likelihood, whereas the multivariate model got a log-likelihood score of -130.3845.

Since the size of the training and testing data differs, another measure has to be done to compare log-likelihoods and the process is called normalization. The process takes the log-likelihood value of the model and the size of the dataframe and divides them (log-likelihood/data size). In this case the log-likelihood of the training model is divided by the size of the training data and the same thing happens with the test dataset. For univariate, the calculation results in two normalized log-likelihoods, -0.07533631 for the training data and 0.1327316 for the test data. As for the multivariate models, -3.012684 and -3.311128 are the normalized log-likelihood of training and testing data respectively. These two corresponding values are close to each other, thus it can be said that the training data is not overfitting.

```
> logLik(fm_uni_n5) / nrow(train_data)
'log Lik.' -0.07533631 (df=34)
> logLik(fm_multi_n5) / nrow(train_data)
'log Lik.' -3.012684 (df=54)
```

Fig 2.4.5 Multivariate training log-likelihood

2.5. Anomaly Detection

Three additional datasets with injected anomalies are also going to be tested to see how likely the original data generate data with anomalies. The same procedure is done with the new datasets, all null values are eliminated and a time window is set (12 pm - 4 pm). In this part, the normalized log-likelihood values of the anomalous data and the training data will be compared.

However only multivariate models will be tested here, thus only the normalized log-likelihood of the trained data will be looked at.

The first anomaly dataset gets tested with a log-likelihood of -516.867. The normalized log-likelihood is -2.457005. Compared to the original dataset value, there is a slight difference.. This means the new dataset can somehow be considered as anomalous but it still has some similarities with the original dataset.

```
> logLik(fm_multi_n5) / nrow(train_data)
'log Lik.' -3.114586 (df=54)
> logLik(fm_multi_test) / nrow(df_weekly_avg)
'log Lik.' -2.457005 (df=54)
```

Fig 2.5.1 Data Anom 1 Log-likelihood

The second anomaly seems to be the most anomalous one and by looking at the raw data, one can tell that the values differ significantly from the original data frame. The log-likelihood of this dataset is -269.1011 and when it gets normalized, it becomes -5.294919. Notice that there is a huge gap between the two values shown in figure 2.5.1. The difference between these two values indicates that the both data are different from each other and it shows that the test dataset has anomalies.

```
> logLik(fm_multi_n5) / nrow(train_data)
'log Lik.' -3.114586 (df=54)
> logLik(fm_multi_test) / nrow(df_weekly_avg)
'log Lik.' -5.294919 (df=54)
```

Fig 2.5.2 Data Anom 2 Log-likelihood

The third dataset is also tested and compared with the original dataframe. In this testing process, the calculated log-likelihood value looks similar to the one in the first anomalous

dataset as -2.3249 is not too far from -3.1146. The implication of this value is the anomalous dataset is similar to the original dataset and there is a fair chance that the original dataset would follow the trend of the testing dataset.

```
> logLik(fm_multi_n5) / nrow(train_data)
'log Lik.' -3.114586 (df=54)
> logLik(fm_multi_test) / nrow(df_weekly_avg)
'log Lik.' -2.324857 (df=54)
```

Fig 2.5.3 Data Anom 3 Log-likelihood

3. Problems Encountered

Unavoidable obstacles were encountered in this project, such as imperfections in the data, lack of ground truth and finding a good balance between model complexity and log-likelihood score.

The original data had some missing values and data exploration and cleaning had to be done first. These measures were actually not enough, as errors were still encountered in the model training and testing process. The depmix method in the program does not work perfectly fine, it sometimes gives a Nan/Infinite value error even though there were neither Nan nor infinite values in the data frame. Thus, most of the time spent in this project was mainly for the training and testing model process.

The lack of the ground truth in the original dataset has an impact on the result of this project because the accuracy of the provided dataset is not known. The models may not represent real-world conditions in the present, thus the comparison and calculation done in this project cannot be a solid reference to the current condition.

The balance of the model complexity or BIC with the log-likelihood score was difficult to find. It is easy to find a more complex model with better log-likelihood. However, a very complex model is not preferred as it will result in an overfitted model. Hence, choosing the number of states for a model is tricky and it involves many trials and errors. Though, choosing 5 as the number of states is more reasonable than choosing 10, considering the trade-off between the model complexity and the log-likelihood.

4. Lessons Learned

Our experience completing this project has taught us multiple new lessons. Through our research we gained better understanding about the applications of anomaly detection in cybersecurity as a method for intrusion detection and its comparison to signature based anomaly detection.For example, anomaly based intruder detection systems typically have a higher false alarm rate compared to signature based. However, by combining anomaly based detection with deep learning methodologies, one can create more effective and efficient systems. Furthermore, we learned more about reinforcement learning and its success in the artificial intelligence industry through DeepMind, and OpenAI.

Our task in anomaly detection has helped us improve our skills in data analysis with regards to principal component analysis, variable selection to select response variables and time frames. Although we are only mirroring an anomaly based intrusion detection, we learned of the various challenges associated with it. To specify, we learned that real-world data can be imperfect as a result of missing values, unavailability of labels, and inconsistency associated with many factors. For a better understanding of the challenges we face, refer to the Problems Encountered section of our report.

5. Conclusion

In conclusion, the three datasets are anomalous to some degree compared to the original data. Log-likelihood values of each model were calculated and it can be used to interpret the degree of anomalies present in the dataset. It can be said that the second dataset is the one with the most anomalies, as it has the biggest difference with the original dataset. The third dataset comes in the second place with a value of -2.3249 and finally the first dataset is the one with the least anomalies. Therefore, there is a better chance for the original data to follow the trend of the first or the third dataset than the second dataset.

6. Technical Essay: Reinforcement Learning in Cybersecurity

Society has dramatically changed with the evolution of the Internet and Internet of Things (IoT) devices affecting every aspect of life by providing humans with convenience and efficiency to communication, entertainment and information. Since its introduction, the Internet and IoT devices has quickly become a vital role to many industries due to its networking capabilities. As modern day society becomes more digital, online business and official government websites are rapiding transitioning to digital storage, storing information into computer systems and online databases. The sensitive information stored in these systems can cause life changing damages to humans and businesses which heavily rely on technology to be connected at all times to conduct work effectively. As everyday life becomes filled with more technology every year, the need for cybersecurity has quickly become a necessity to prevent such damage from ever happening.

Cyberattacks are a growing threat to modern society, seeking to gain, destroy, or manipulate data; or disrupt services to extort money from individuals and businesses. Access to sensitive data and information exposing personal identities, intellectual property, or financial assets can cause major damages to a business and a person's life. The field of cybersecurity is the practice of protecting systems and networks from cyberattacks through the protection of information, hardware, and software from theft or damage; and protection of disruption of service. Although the goal of cybersecurity is to prevent cyberattacks, it is impossible to stop all of them. Just as professionals are researching new ways to prevent attacks, attackers are researching new vulnerabilities and methods to break existing security systems.To prevent cyberattacks, organizations and professionals research an existing system for vulnerabilities to determine cyber risk. Cyber risk is the probability of a cyber attack occurring multiplied by the potential damages on assets that would result from it. Understanding cyber risk allows organizations to have a clear understanding of a system's vulnarbilities, potential risk, and assets that must be protected. Although cyberattacks cannot be fully prevented, businesses can create security systems to minimize damages or delay the attack long enough to find a solution. In this paper, we will learn about reinforcement learning as a machine learning paradigms and it's applicability to cybersecurity to create an intrusion detection security system through anomaly detection.

Reinforcement learning is one of three basic machine learning paradigms, focused on interaction and feedback, an agent learns by interacting with a complex environment through trial and error to maximize the notion of cumulative reward by forming a sequence of decisions. To begin, every reinforcement learning problem consists of the same key components: An agent, an environment,a policy, multiple states, reward(s) and punishment(s). In reinforcement learning, an agent is the entity performing an action in a scenario known as the environment. The environment leads to different states which refers to the current situation of the environment. The agent then uses trial and error to interact with the environment. In the event that the agent does the appropriate action, the agent will be rewarded with a reward, and

inversely a punishment if the agent does not do the desired action. In every state the agent experiences a punishment or reward, the agent will update the policy. Lastly, the policy is a sequence of actions an agent takes depending on the current state to maximize reward.

To help illustrate reinforcement learning better, one real life approach of reinforcement learning is the process of teaching a pet new tricks. For this example, imagine trying to teach a dog a new trick at home. As a human we don't understand how to speak the dog's language and vice versa. Thus, in order to teach the dog new tricks, we must teach it through reinforcement learning. When teaching a dog new tricks, we create trigger words such as 'sit', 'jump','paw', and etc. in order to trigger an action in the dog. Each trigger seeks to transition the dog from its current state to the desired state. In the event the dog enters the desired state, we would reward him with a treat. At the same time, the dog is learning that every other action results in a negative experience as it is not getting any rewards or some other form of punishment. For example, if we say "sit", every action other than a sitting state will not result in a reward. To summarize our example, the dog is the agent,the home is the environment, and every other trigger word results in a new state and desired action. By the end of reinforcement training, the dog should be able to identify the trigger words and understand the associated actions that result in punishments and rewards.

Almost all reinforcement learning problems can be formed into a 5-tuple mathematical framework known as the Markov Decision Process or MDP consisting of the State (s_t) at the time (t), Actions available from state s (a_t), State Transition Probabilities $P(s_t+1| s_t, a_t)$, Discount factor for future Rewards (y) and Reward function (r_t) (Koduvely, 2018). The goal of the MDP is to form an optimal policy ($\boldsymbol{\pi}^*$) by creating a policy function $\boldsymbol{\pi}$ that determines the optimal action that should be taken at each state so the agents will maximize cumulative reward over a long period of time. A MDP model can be represented by the following diagram:

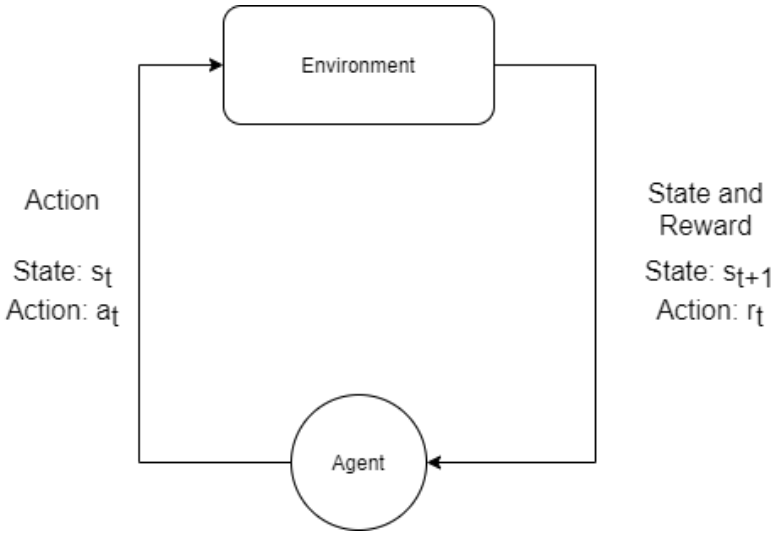

Fig 5.1 Reinforcement learning diagram

There are many established methods for solving MDP problems. One of the most popular methods is known as Q-learning which defines a quality function Q(s,a) estimating the maximum cumulative reward an agent can receive given state s and action a. As of now Q(s,a) only outputs the estimated reward function for a single state at a time. In order to generate an optimal policy $\boldsymbol{\pi}^*$ we must take into account future states s* and their associated Q-values. This can be done by iteratively solving Bellman equation:

Q(s,a) = r + y maxQ(s*,a*)

For this example, consider the game of Go, a standard board size is 19x19. As each tile can possibly be white, black, or unclaimed the total number of possible states is $3^{19\text{^}2} \approx 1.74\text{x}10^{172}$. Using the iterative Bellman equation approach we must compute the quality value for all $1.74\text{x}10^{172}$ states multiplied by the number of actions each state can take. From this example, it is clear that Q-learning by iteratively solving a Bellman equation is not practical for real world solution due to its scalability. To solve the scalability issue, reinforcement learning can be combined with deep neural networks (DNN) in a process known as deep learning. DDN is an artificial neural network (ANN), a machine learning concept heavily inspired by the structure and process of biological brains to classify and make informed decisions based on past learnings. DNN differs from typical ANNs in that it uses multiple layers of black-box functions to transform input into output. Each layer uses a weight to determine the optimal black-box function to be used at that layer. Deep-learning also applies a principle known as “online learning”. Online learning is a method adjusting the weights of DNNs layers as the model trains. Online learning is particularly useful for models which require the agent to dynamically adapt to new patterns in data over time. Deep learning solves the issue of scalability which traditional bellman equation approaches suffer and showcases other benefits such as dynamic modelling, feature extraction, parallelizability, and higher dimension complexity.

In recent years, deep learning has made huge strides in the artificial intelligence industries through the success of AlphaGo (DeepMind Technologies, 2017) and OpenAI Five’s Dota 2 AI (*OpenAI*, 2017). Both achievements were viewed as major milestones for artificial intelligence as machine learning AI’s were able to show strategy, teamwork, and collaboration at the highest level through its victory over professionals in the respective games. Although both achievements were relative to the gaming industry, deep learning can be conceptually applied to any industry.

Anomaly detection is a pattern classification problem for finding patterns in data that do not conform to expected behaviour. By using machine learning, we can teach an agent to recognize patterns of normal behaviour and use it to detect anomalies - a pattern that does not conform to the expected behaviour. Anomaly detection is used extensively in a wide variety of applications including fraud detection for credit cards, insurance and health care, fault detection in safety critical systems, criminal intelligence and many more. In cybersecurity, anomaly detection is used to create intrusion detection systems monitoring the activity level of computer systems and networks. In other words, the intrusion detection system looks to detect deviations

of good or expected traffic which can result in attacks such as Bot attacks orDistributed denial of service attacks (DDoS).

To begin classifying behavior as normal or anomalous , anomaly based intrusion detection systems must be trained to recognize the behaviour of system activity. This training can be accomplished by deep learning in combination with online learning. For example, In a network traffic instruction system, we want to observe parameters such as protocol number, port number, packet size, and transmission rate. Using these parameters we will create a DNN model to classify the relationship between our parameters to determine if it can be considered normal or anomalous traffic. Early intrusion detection is useful as it helps to mitigate potential damages or prevent future attacks. In cybersecurity, the landscape of cyberattacks are constantly evolving and it becomes a necessity that anomaly based intrusion detection systems can evolve as well. By applying online learning, anomaly based intrusion detection systems can continuously learn and adapt to new patterns, shifting the weight of its model to incorporate the adjustment of a new normal to find anomolies. One weakness of anomaly based systems is an inherited problem from online learning known as catastrophic interference. Catastrophic interference is the tendency of an ANN to abruptly forget previously learned information. Attackers can force a catastrophic interference on an existing system by sending fake “good” traffic over a long period of time and altering the expected behaviour. Despite potentially suffering from catastrophic interference and suffering from computational complexity compared to other intrusion detection systems, anomaly based intrusion detection systems are still very powerful systems due to their low false positivity rates and high resilience to novel attacks.

As society grows dependent on the use of the internet and social media, and with the recent success of artificial intelligence technology such as AlphaGo, and OpenFive’s Dota2 AI, the necessity for cybersecurity has become more important than ever. Artificial intelligence technologies have proven machine learning as a powerful tool to some complex problem. As artificial intelligence continues to advance, criminals are unafraid to exploit AI as a method to create cyber attacks for personal gain. Access to sensitive data and information exposing personal identities, intellectual property, or financial assets can cause life changing damages to people and business. Throughout the next few decades, the need for cybersecurity to understand, develop and adapt new defences will be paramount to keeping the everyday lives of business and individuals alike secure.

7. Bibliography

DeepMind. (2016, 06 17). *Deep Reinforcement Learning*. Deep Reinforcement Learning | DeepMind. Retrieved 12 01, 2020, from https://deepmind.com/blog/article/deep-reinforcement-learning

DeepMind Technologies. (2017, 10 18). *AlphaGo | DeepMind*. AlphaGo Zero: Starting from scratch | DeepMind. Retrieved 11 30, 2020, from https://deepmind.com/blog/article/alphago-zero-starting-scratch

Koduvely, H. (2018, 01 19). *Anomaly Detection through Reinforcement Learning*. Anomaly Detection through Reinforcement Learning | Zighra. Retrieved 12 01, 2020, from https://zighra.com/blogs/anomaly-detection-through-reinforcement-learning/

Moni, R., & SmartLab AI. (2019, 02 18). *Reinforcement Learning algorithms — an intuitive overview*. Reinforcement Learning algorithms — an intuitive overview | by SmartLab AI | Medium. Retrieved 12 01, 2020, from https://medium.com/@SmartLabAI/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc

OpenAI. (2017, 08 11). Dota 2. Retrieved 11 2020, 30, from https://openai.com/blog/dota-2/

Osiński, B., & Budek, K. (2018, 07 05). *What is reinforcement learning? The complete guide*. What is reinforcement learning? The complete guide - deepsense.ai. Retrieved 12 01, 2020, from https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide/

Rouse, M., Lutkevich, B., & Rosencrance, L. (2020, 02 30). *intrusion detection system (IDS)*. What is an Intrusion Detection System (IDS) and How Does it Work? Retrieved 12 01, 2020, from https://searchsecurity.techtarget.com/definition/intrusion-detection-system#:~:text=An%20intrusion%20detection%20system%20(IDS)%20is%20a%20system%20that%20monitors,when%20such%20activity%20is%20discovered.