# FINAL PROJECT REPORT

## CMPT 318 Fall 2024

### Group 19

### Author:

Tushar Singh (tsa156) - 301560473

David Mulej (dma122) – 301539538

Victor Nguyen (avn) – 301458739

Arshdeep Kaur (aka232) – 301540172

## Abstract:

To understand the underlying behaviors of electricity energy consumption, our team aims to enhance our ability to recognize patterns and anomalies through various standard dataset processes. With methods like Data Processing, Principal Component Analysis, Hidden Markov Models, and Anomaly detection through thresholding, our project leans towards the process of working together with autonomous systems to recognize and strengthen their robust capabilities to mitigate the threat that exploits the various vulnerabilities of real-world scenarios.

# Table Of Contents

# 1. Introduction

## 1.1 Problem Scope

The increasing reliance on automation in critical infrastructure such as electric power grids, public water utilities and smart transportation networks has enhanced cost efficiency, quality of service delivery and safety. However, this increasing reliance also expanded the attack surface for adversarial threats, amplifying the risk associated with cascading failure. Supervisory control systems, integral to these infrastructures, must remain resilient against cyber intrusions to ensure smooth and continuous operation. This report will provide in-depth analysis of the use of statistical computing and machine learning techniques to develop cybersecurity models for anomaly detection systems. We aim to explain the methods employed in using statistical computing for anomaly detection and provide an interpretation of the outcomes from our experimental models in identifying anomalies in electric energy consumption data.

## 1.2 Technical Background

Our dataset of US electrical energy consumption is a time series data. Usually, the typical consumption pattern for these datasets does not deviate too much from the best fit line but they have their own challenges as well. Firstly, times series data often have imperfections such as missing or corrupted values which do not conform to typical normal behavior. These values are considered anomalies. The nature of anomalies varies depending on the operational context, which makes adaptive detection methods necessary. These anomalies can be simple errors in value recording or cyber-attack evidence. Hidden Markov Models (HMMs), a robust probabilistic framework for modeling temporal data, are employed to address these challenges. For investigative purposes, anomaly-based detection systems should be used to consistently monitor and identify unusual data

patterns, flagging them as irregular. This triggers human involvement, allowing for informed decision-making. This project is using Hidden Markov Models (HMMs), a robust probabilistic framework for modeling temporal data to address these challenges. The main question we need to address is what factors we must consider when designing a cybersecurity system capable of detecting such anomalous data.

## 1.3 Project Objective

For our project, we used various techniques to develop an anomaly-based detection system. We focus on achieving the following objectives:

1. Data Preprocessing: Clean the data, implement feature scaling methods, comparing normalization and standardization, to ensure optimal preprocessing for HMM training.
2. Feature Engineering: Apply Principal Component Analysis (PCA) to select relevant response variables that capture significant variance in the dataset.
3. HMM Training and Testing: Develop and evaluate multivariate HMMs with varying numbers of states, using log-likelihood and Bayesian Information Criterion (BIC) to identify the best-performing models.
4. Anomaly Detection: Establish a threshold to distinguish normal from anomalous observations by analyzing log-likelihood distributions of validation data and injected anomalies.

This systematic approach allowed us to develop an anomaly-based detection system capable of identifying irregularities in critical energy consumption data streams with high precision.

# 2. Methodology and Design Choices

## 2.1 Data Preprocessing and Cleaning

The dataset available for energy consumption is for years 2006, 2007, 2008 and 2009 with features {Date, Time, Global_active_power, Global_reactive_power, Voltage, Global_intensity, Sub_metering_1, Sub_metering_2, Sub_metering_3}. While analysing the data we observed that if Global_reactive_power is NA for any row then other features except Global_active_power, Date, Time are also marked NA. Initially, the dataset consisted of 1,556,444 rows, but after completing data preprocessing and cleaning, we retained 1,512,000 rows.

To prepare the data, we first ensured it contained only complete weeks by removing any extra days that did not form full weeks. We then identified and excluded the top five weeks with the highest number of missing (NA) values, as these could significantly affect the training process. To standardize the dataset, we applied z-score scaling, ensuring that all features contributed equally regardless of their original scales. After scaling, we handled the remaining missing values by interpolating them. Next, we identified outliers using the 3-standard-deviation method, flagging and removing data points that were more than three standard deviations away from the mean. These outliers were replaced with NA values, which we then addressed using linear interpolation to maintain the integrity of the dataset.

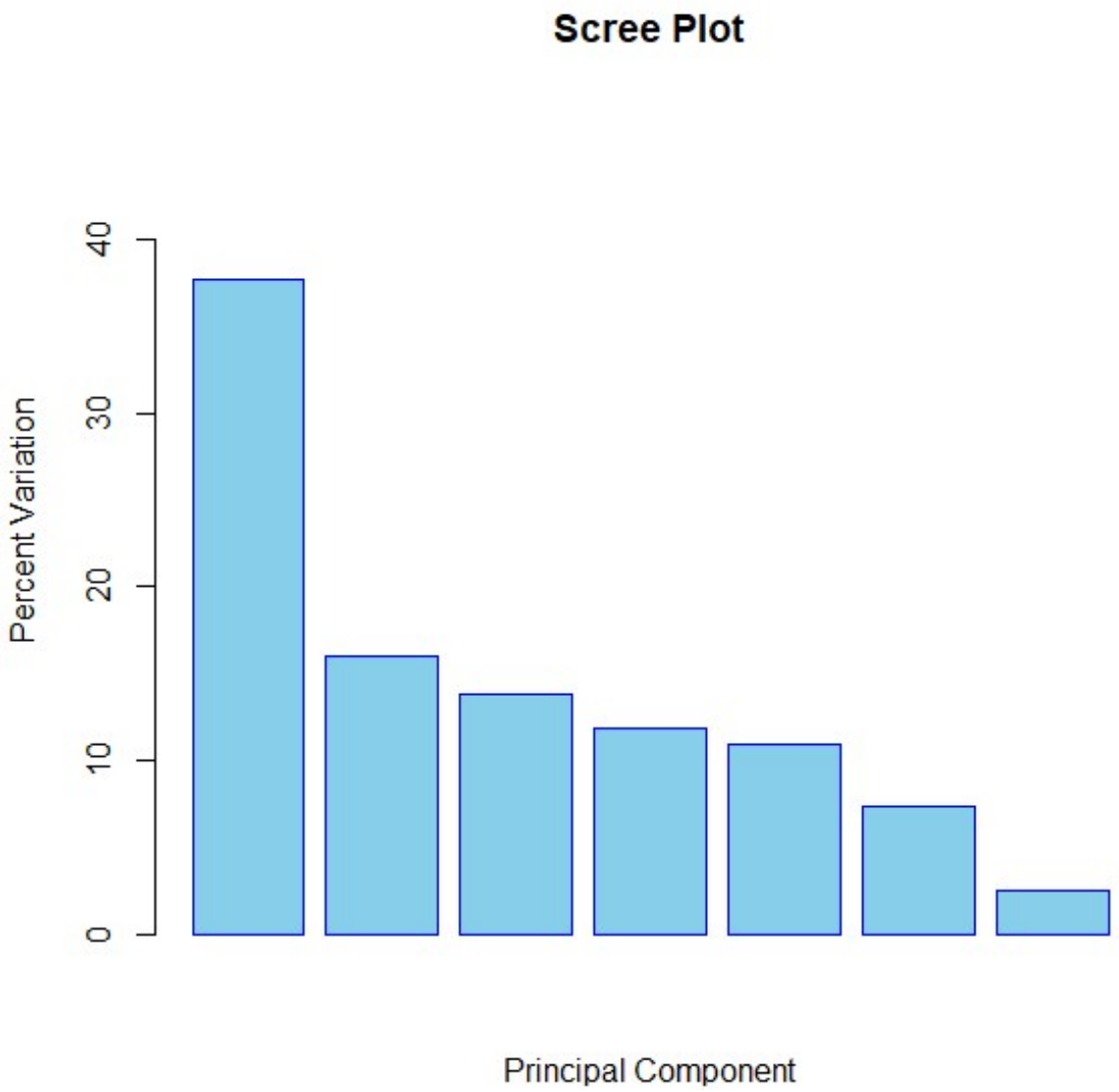
## 2.2 Principal Component Analysis (PCA) and Feature Selection

Principal Component Analysis (PCA) is a method used to find a subset of variables by reducing the number of Principal Components (PC) and thereby reducing the number of features needed to train the model. Usually, the number of PCs you aim to reduce to are two or three if you just want a general

visualization of the data, however in certain cases such as doing a precise analysis you may use more PCs so you can get a better representation of the total variance. This is just a general rule of thumb, however since what matters is not the number of PCs used, but rather the amount of variance these PCs cover. If you're looking to just visualize the data, you don't need as much of the total variance (only around 60-80%), but if you want to have a precise analysis then you're going to want to have more of the total variance (>90%). %). To determine the number of PCs you can use a scree plot to determine if a PC is relevant. A scree plot looks like a bar graph with the PCs on the X axis and the percentage variation or eigenvalues on the Y axis. Using this visual representation, you can analyze the "elbow point" which is the point where the curve of the bar graph starts to level off. All the PCs up to the elbow point are kept and only the additional PCs beyond this point are discarded since they contribute minimal variance. For not so ideal scree plots you can either use the Kaiser rule which is to pick PC's with at least eigenvalue of 1 or choose PC's until you reach a variance of 80% [3]. In our case we went with the Kaiser rule since the second method would have included too many PCs, while the first method would simplify our dataset significantly while still allowing us to retain enough variance.

For our project we used the results from the PCA performed along with the scree plot to determine which PCs to use. As seen below the first two PCs make up 53.71% of the total variance which isn't exactly a large sum, but it's enough to have a visual representation of the data. The scree plot was really what helped us to be sure of the numbers of PCs to use due to the clear elbow bend in the plot.

```
Importance of components:
                          PC1    PC2    PC3    PC4    PC5     PC6     PC7
Standard deviation     1.6248 1.0581 0.9830 0.9084 0.8719 0.71499 0.42152
Proportion of Variance 0.3771 0.1599 0.1380 0.1179 0.1086 0.07303 0.02538
Cumulative Proportion  0.3771 0.5371 0.6751 0.7930 0.9016 0.97462 1.00000
```




After we determined the number of PCs to use, we next had to look at the variables that contributed the most to PC1 and PC2. (Values of the variables contributing to PC1 and PC2 respectively)

```
print(loading_scores1)
Global_active_power Global_reactive_power              voltage     Global_intensity       Sub_metering_1       Sub_metering_2
         -0.4876731           -0.1936196            0.3124681           -0.5710638           -0.1968584           -0.2093571
     Sub_metering_3
         -0.4672821
print(loading_scores2)
Global_active_power Global_reactive_power              voltage     Global_intensity       Sub_metering_1       Sub_metering_2
         0.12731053           -0.68278090          -0.12537284           0.06734613           -0.10214420          -0.62635375
     Sub_metering_3
         0.30756455
```

For this we decided to use the top variables for PC1 and PC2 which were Global_intensity, Global_active_power, Global_reactive_power, and Sub_metering_3 since they were the top two

contributors to PC1 and PC2 respectively. This means that these variables introduce the most variance in the dataset hence making them the top choice.

## 2.3 Observation Time Window Selection

The observation window selection for the training data is Wednesday from 12:00 pm to 2:00pm because it represents a consistent mid-week time with regular activity. The narrow window chosen reduces the chance of variability and noise, allowing for a more focused analysis of system behavior.

## 2.4 Hidden Markov Model (HMM) Analysis

Hidden Markov Model is a statistical model that represents a stochastic system that transitions between a finite set of hidden states over time. These hidden states are indirectly inferred through observable data. Each hidden state is associated with a specific probability distribution that governs the generation of observable data. The goal of this analysis is to develop and evaluate a Hidden Markov Model to analyse time series data related to power consumption.

We divided the dataset into training and testing based on the years the data is from. For the training data we extracted all the PCA variables for the years 2006,2007 and 2008 and further filtered them to include only the values from Wednesday 12:00pm to 2:00pm. For the testing data we used thee same filtering criteria on the year 2009.

For the PCA variables namely Global_active_power, Global_intensity, Global_reactive_power and Sub_metering_3 they were all discretized using rounding to lessen the level of detail and make HMM training easier. We did try the top 3 attributes that contributed to PCA1 and PCA2 which were Global_intensity, Global_reactive_power and Sub_metering_3 but those took too long, and training data was underfitting so we used 4 variables.

A likelihood function evaluates how effectively a statistical model accounts for the observed data by determining the probability of obtaining that data for various parameter values of the model [1]. A less negative value mean it better fit the data than a high negative value. On the other hand, BIC (Bayesian Information Criterion) is a statistical criterion used for model selection among a set of models. It balances the goodness of fit of the model with its complexity to avoid overfitting [2]. A lower BIC value indicates a better model, as it suggests a better fit to the data while penalizing overfitting due to a large number of parameters.

The HMM was constructed with 4,6 and 8 hidden states and each variable was modelled independently using the multinomial distribution. For each of the models created the log-likelihood and BIC scores were calculated to assess the model fit and complexity. We tried to have 10 and 12 hidden states too but the training for those states took too long, so we decided to just stick with states 4,6 and 8 due to time and technological limitations.

| States | Log-Likelihood | BIC |
|---|---|---|
| 4 | -50641.10 | 102593.03 |
| 6 | -46772.83 | 95629.80 |
| 8 | -45417.89 | 93768.68 |

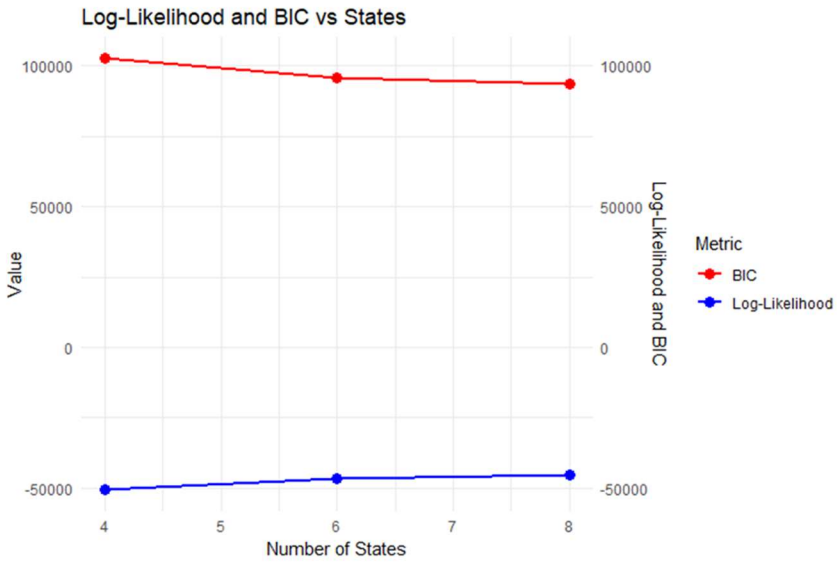



Fig: BIC values and Log-likelihood values

From the observed values for the Log-Likelihood as the number of states increase the Log-likelihood improves suggesting that the models that have more states fit the data better. The decrease in BIC values for the same increase in the number of states suggests a better balance between model complexity and fit for models with more states. The model with 8 states has the lowest BIC for us making it the ideal model choice for our test model.

The 8-state model was then evaluated on both training and testing datasets and the normalised log-likelihood was calculated for both the datasets.

| Dataset | Normalised Log-likelihood |
|---|---|
| Training | -3.644218 |
| Testing | -4.304683 |

The normalised log likelihood for the training data indicates that it is a good fit to thee training dataset. While for the testing data the normalised log likelihood is slightly worse than that for the training data

which is to be expected since the testing data was hidden during model training. The result also shows that the model generalises well to a new data.

### 2.4.1 Comparison with other HMM Model we analyzed

We trained a 3-variable HMM and observed that the model was underfitting the training dataset. Ideally, the training normalized log-likelihood should be better (less negative) than the testing value, as the model is optimized on the training data. However, in this case, the results indicate that the model struggled to adequately fit the training data, leading to suboptimal performance when evaluated on unseen testing data. Below were the values we got for 12 states HMM with 3 variables, so we decided to go with 8 state 4 variables HMM. Even though we got better values for states 4, 6, 8, 10, and 12 from 4 variables for normalization and BIC the fact that it underfits we decided to go against it.

| Dataset | Normalised Log-likelihood |
|---|---|
| Training | -2.398809 |
| Testing | -1.969653 |

## 2.5 Anomaly Detection

After obtaining our best Hidden Markov Model, we now attempt to uncover a baseline that can be used to check whether a subset of data is within the bounds of our normal data. To reveal anomalies with HMMs, we utilize our normalized training log-likelihood to calculate the maximum normalized log-likelihood deviation with 10 separate yet consecutive weeks in our test dataset. This result will act as our threshold for detecting anomalies with the first few test weeks modeling “the standard normal week” that our model seeks to follow. From there, any test data we wish to verify can be done

by comparing the normalized log-likelihood of the week with the upper and lower thresholds we obtain from the 10 weeks.

We perform this task by first grouping and dividing the data values based on the date of their occurrence. This was done by finding unique entries in the date column of the dataset and storing their row indices. With the unique row indices, we store the week's worth of data corresponding to entries that happened on the same date in a list. However, before the data is feed to the HMM model, we need to check if there is more than one unique entry on a specific column, not including it if that is the case. This is due to an error with depmix regarding contrasts having less than 2 unique values.

```
Error in `contrasts<-`(`*tmp*`, value = contr.funs[1 + isOF[nn]]) :
  contrasts can be applied only to factors with 2 or more levels
```

We bypass this by ignoring columns with only 1 unique value. Then we insert it into a model and compute the log-likelihood using the previously fitted data of our training model. Then we calculate the deviation between the normalized log likelihood of this test model and the training model. This deviation is calculated using $|normalized\ test\ week \log lik - normalized\ train \log lik|$ instead of mean squared error because we want our bounds to be as tight as possible. Then by finding the maximum of this deviation in our 10 subset weeks, we can obtain an upper and lower bound by adding and subtracting the deviation from the train log-likelihood to highlight outliers far from this specified log-likelihood.

| Test week # | Normalized Log-Likelihood |
|---|---|
| 1 | -5.123964 |
| 2 | -4.276666 |
| 3 | -3.988984 |
| 4 | -3.295837 |
| 5 | -5.817111 |
| 6 | -5.123964 |
| 7 | -2.772589 |
| 8 | -5.075174 |
| 9 | -5.123964 |

| | |
|---|---|
| 10 | -4.787492 |

| | |
|---|---|
| Normalized training Log-Likelihood | -3.62164 |
| Threshold Deviation | 2.195472 |
| Upper Threshold | -1.426168 |
| Lower Threshold | -5.817111 |

The training normalized log-likelihood of -3.62164 indicates a good fit to the training data. Testing log-likelihoods range from −5.817111 (week 5) to -2.772589 (week 7), with most values near the training log-likelihood, suggesting the model generalizes well. The threshold deviation of 2.195472 yields an upper threshold of -1.426168 and a lower threshold of -5.817111. Week 4 and week 7's log-likelihood are much larger than the normalized training log likelihood, indicating potential anomaly, or overfitting/underfitting in this dataset while the other weeks fall within the expected range. Since all data fall within bounds of the thresholds, they are deemed to be normal behavior.

## 3. Key Findings

The analysis revealed that the 8-state Hidden Markov Model (HMM) with 4 variables (Global_active_power, Global_intensity, Global_reactive_power, and Sub_metering_3) provided the best fit, based on the lowest Bayesian Information Criterion (BIC) and log-likelihood values. While the model performed well on the training dataset, its generalization to the testing data was slightly less optimal, which is typical due to the unseen nature of test data. A 3-variable HMM underperformed, leading to underfitting. In the anomaly detection phase, most test weeks fell within the expected log-likelihood range, with a few weeks indicating potential anomalies, suggesting that the model effectively identifies deviations from normal patterns in power consumption.

# 4. Problems Faced

Throughout this project, our team faced several challenges, particularly in the data cleaning process. One of the first hurdles was deciding how to handle extreme values and determining which data points to retain or discard. After experimenting with various methods, we found that the most effective approach was to interpolate the outliers after marking them as missing values (NA). While this approach introduced some unusual behavior in the data, it ultimately proved to be the most successful solution for our analysis.

One of the most critical tasks in building a Hidden Markov Model (HMM) is determining the appropriate number of hidden states. These hidden states decide the underlying mechanisms that generate the observable data, making their selection crucial for both the interpretability and quality of the model. Choosing too few hidden states can result in an underfit model that fails to capture the complexity of the data, while selecting too many states risks overfitting, where the model captures noise rather than meaningful patterns. Additionally, some state configurations either failed to converge or required excessively long training times, leading to their exclusion from consideration.

Another challenge was selecting the appropriate time window and day for the analysis. Initially, we experimented with a 6-hour observation window, but training on these datasets proved computationally intensive. Reducing the window to 4 hours yielded similar issues. Ultimately, we settled on a 2-hour observation window, which provided a feasible balance between computational efficiency and model performance.

The choice of probability distribution also posed difficulties. Our initial approach employed a Gaussian distribution, but this resulted in positive log-likelihood values for higher numbers of states, even after data scaling. To address this, we switched to a multinomial distribution. Although this

method required longer training times, it consistently produced negative log-likelihood values and delivered models with superior fit.

# 4. Conclusion

In this project, the Hidden Markov Model (HMM) played a central role in analyzing the power consumption time series data. The goal was to uncover hidden patterns in the data that could offer insights into typical power consumption behaviors, while also identifying outliers or anomalies. By modeling the data with an HMM, we were able to represent the system as a stochastic process that transitions between a finite set of hidden states. Each state was associated with a distinct probability distribution that governed the generation of observable data, such as global active power, global intensity, global reactive power, and sub-metering values.

The Hidden Markov Model (HMM) was tested on both the training and testing datasets, with normalized log-likelihood values showing how well the model generalized to new, unseen data. Although the model performed slightly worse on the testing data, it still managed to classify most data points within expected ranges, demonstrating good generalization. The anomaly detection feature, which identified unusual data points by comparing their log-likelihood values to the training baseline, worked well for spotting outliers. While the project faced challenges like long computation times and some underfitting issues, the HMM provided valuable insights into power consumption patterns. The model could have been improved if we had more computational power to test with a larger number of states in a reasonable amount of time. However, we made the best use of the available resources and were able to achieve meaningful results within those constraints.

## References

[1] *Likelihood function*, Wikipedia*, The Free Encyclopedia*. [Online].

Available: https://en.wikipedia.org/wiki/Likelihood_function. [Accessed: 24-Nov-2024].

[2] N. Dridi and M. Hadzagic, "Akaike and Bayesian Information Criteria for Hidden Markov Models,"

*Signal Processing Letters*, IEEE, vol. PP, no. 99, pp. 1-1, Dec. 2018, doi:

10.1109/LSP.2018.2886933.

[3] S. Mangale, "Scree Plot," *Medium*. [Online]. Available:

https://sanchitamangale12.medium.com/scree-plot-733ed72c8608. [Accessed: 24-Nov 2024].