Anomaly Detection with Hidden Markov Model

CMPT 318: Cybersecurity Spring 2022

Group 4

Owen Brosseau 301363171

Jennifer Dewan 301361418

Abstract:

The importance of cybersecurity is growing more and more each day as our dependence on technology grows and without the ability to detect anomalies we would be risking the livelihoods of millions of people. With the help of Principal Component Analysis and Hidden Markov Models we learn how to detect anomalies in very large data sets which is essential for automated control processes. After testing with three different sets of data with different anomalies, we conclude that the third data set has the most abnormalities. With such large data sets, manually checking each data point is impossible and a thorough and reliable method to check for anomalies is essential for strong cybersecurity.

Table of Contents

Introduction

Problem Scope

As the technology industry progresses, we continue to become more dependent on mechanization to improve the overall quality and consistency of programs that aid our needs. While this does allow for more connectivity and ease, it creates more space for intrusion or error. Without the proper protection and supervision, our most crucial utilities will become ineffective, leaving many helpless. Through this term project we utilize Principle Component Analysis and Hidden Markov models to detect anomalies and explore the importance of intrusion detection.

Technical Background

Principal Component Analysis (PCA)

Often when dealing with large sets of data there are many different variables. In order to simplify this kind of information we need to use Principal Component Analysis. This method allows us to transform the large data set into a smaller, more compact version of the data set. While we will be losing some data, we are prioritizing simplicity over complete accuracy. This is an important sacrifice to make because it allows us to utilize the data more efficiently while keeping most of the information. The results of the PCA gives us loading scores for each of the variables which we use to determine which variables encapsulate the most information from the original dataset.

Hidden Markov Model (HMM)

The Hidden Markov model is used to represent sequential data that is not influenced by its previous data. It utilizes Markov models to display the probability of the occurrence of different states but also takes into account the hidden states that we are unable to observe. With this model we are able to solve three different problems, we can evaluate the probability of an observation sequence, determine the most likely sequence of states for an observation sequence and finally, optimize the probability of a sequence.

Project

Task 1

This task required us to choose a subset of the response variable from the data given to train the Hidden Markov models. To start this process, we started by importing the data and assigning the proper date and time information. This includes functions like $Date_time so that we are able classify the data by its appropriate day, month, year, hour, minute and second. Once this was complete, we went ahead to Task 2 and determined the weekday and time period we would use. We did this early to help simplify the data and cut out a lot of extra information.

Next, we had to standardize the raw data. Standardization is important because it allows us to keep the standard deviation of the data at 1 and mean at 0. By doing so, we eliminate the chance that variables with very different ranges will impact the result. With standardization we are able to have a more reliable outcome. In the code we used the function scale() that we learnt early in the semester and used in the first assignment.

Finally, we needed to import the library stats to calculate the PCA and used the built in function prcomp() for our scaled values. The graph below expresses the PCA function and allows us to visualize the clusters that form.

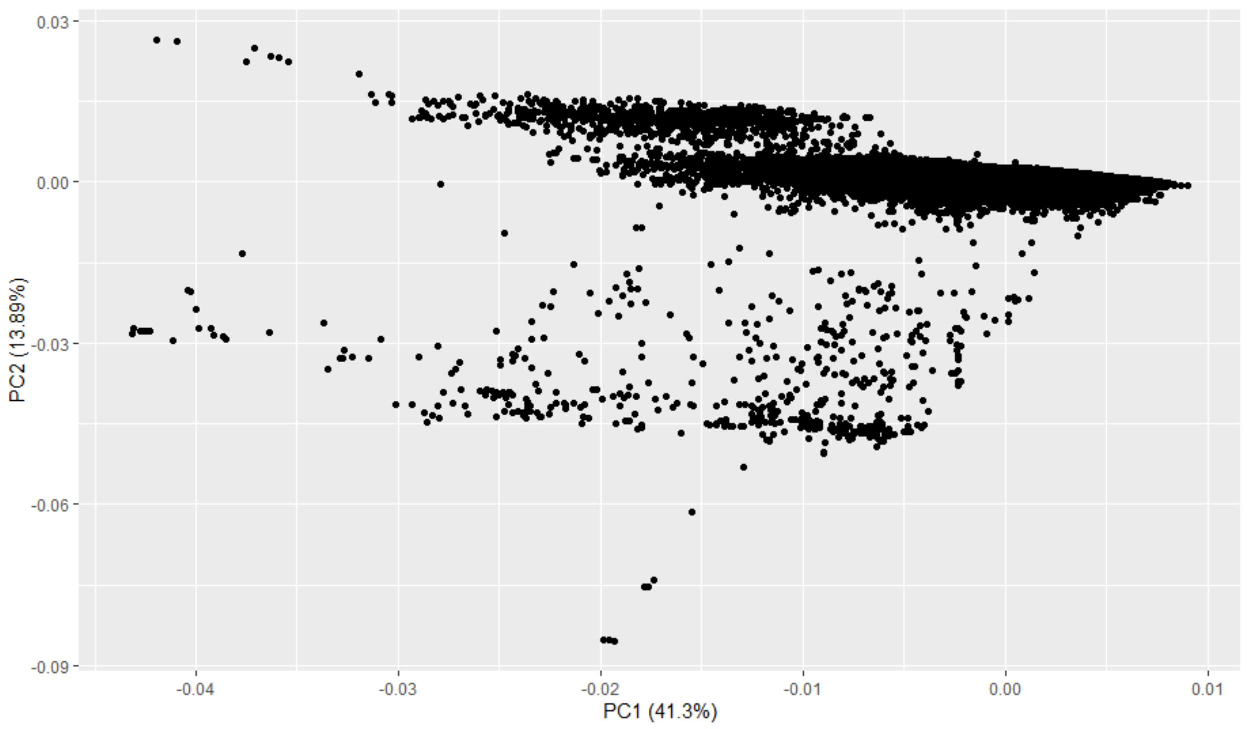
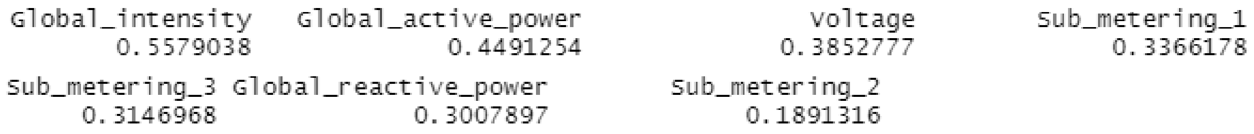


Figure 1: Graph representation of PCA calculated from the scaled values.

The prcomp() function provides plenty of different information regarding the PCA but the key information we took from it was the loading scores. Once sorted, we were able to determine that Global_intensity, Global_active_power and Voltage had the highest absolute values for loading scores. Below we have attached the values for the sorted loading scores to present the values that we arrived at.



```
Global_intensity   Global_active_power                Voltage        Sub_metering_1
       0.5579038             0.4491254              0.3852777             0.3366178
Sub_metering_3 Global_reactive_power         Sub_metering_2
     0.3146968             0.3007897              0.1891316
```

Figure 2: The sorted loading scores (absolute values).

Task 2

For this task we needed to train and test Hidden Markov Models on our data. To start, we partitioned the data, choosing an 8-3 split of training vs test data, since we wanted to keep the 180 minute groupings we chose for the data. To start we established the n state values which range from 4 to 24. With this information, we established how to get log-likeliness and BIC using the lines:

logLiks <- rep(0, times = length(nstateValues))

testLogLiks <- rep(0, times = length(nstateValues))

BICs <- rep(0, times = length(nstateValues)).

After the initial set up, we created a loop to train the HMM, creating a model for every other increment of the n state values (even values from 4 to 24). In this loop we used the depmix package for our HMMs. We extracted the log-likelihood and BIC, as well as testing the model on the partitioned test data. In order to test our models on the test data, we created new models with the same number of states as the model we are testing, but give this model the test data instead of the training data. Instead of fitting this model to the data, we then copy the parameters from the trained model that we are trying to test. This leaves us with a model trained on the training data, but that has been given the test data, which we can extract log-likelihood from using the forwardbackward function, and then accessing logLike from the results. We summarized each of the n state values with its HMM and determined that the ideal model with the best performance is when there are 12 states.

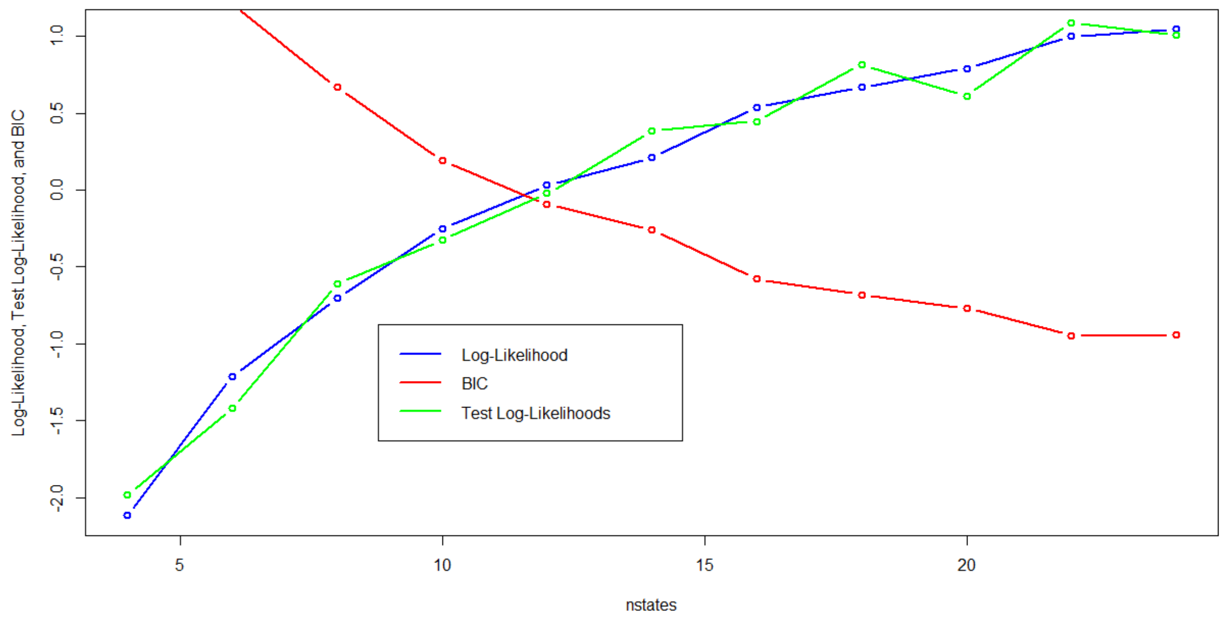
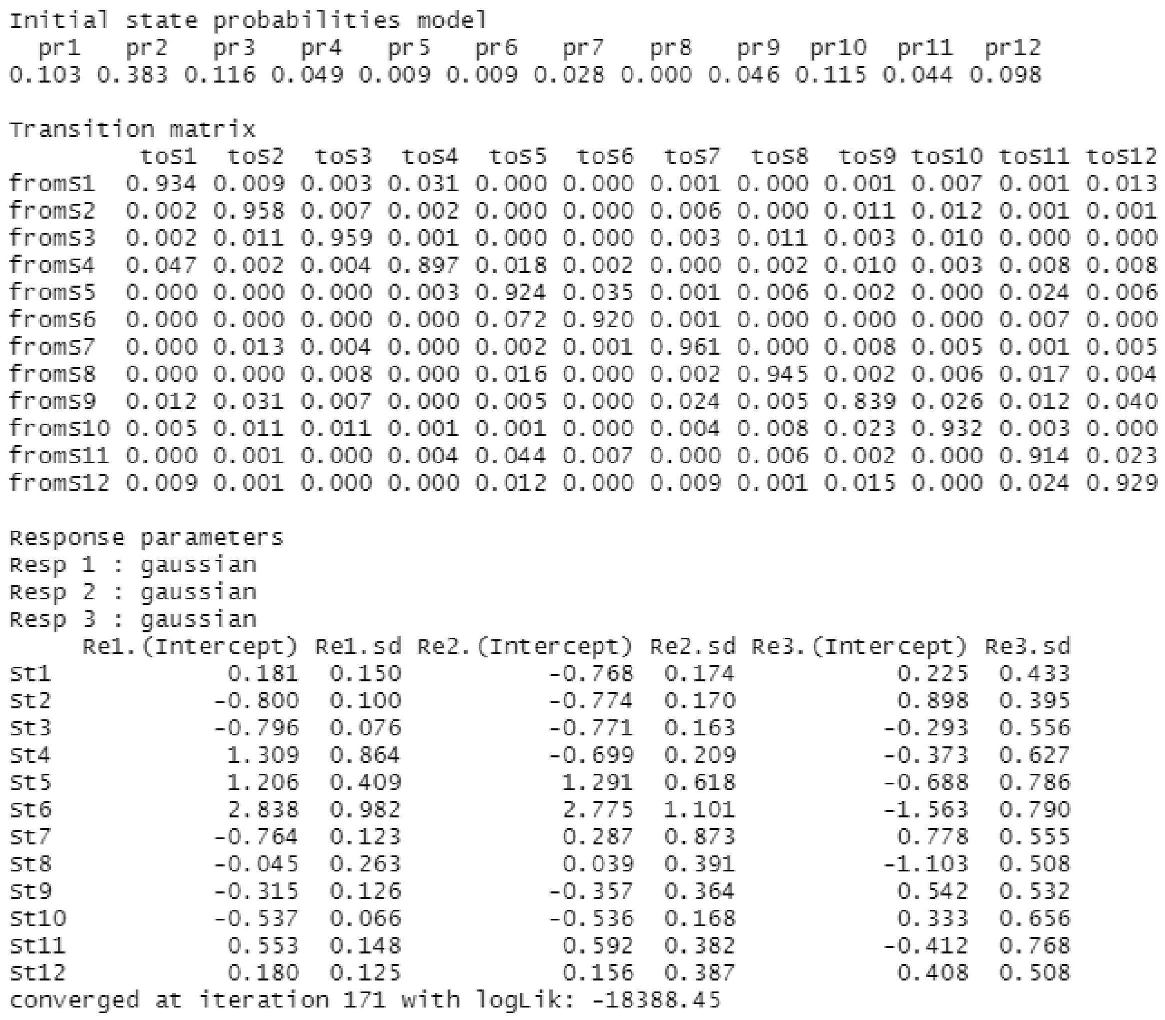


Figure 3: Graph representing the Log-likelihood and BIC for n states from 4 - 24



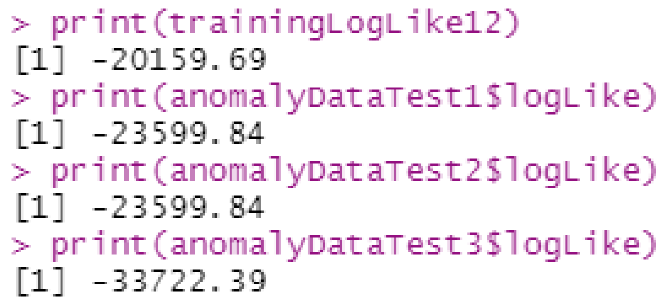
```
Initial state probabilities model
  pr1   pr2   pr3   pr4   pr5   pr6   pr7   pr8   pr9  pr10  pr11  pr12
0.103 0.383 0.116 0.049 0.009 0.009 0.028 0.000 0.046 0.115 0.044 0.098

Transition matrix
        toS1  toS2  toS3  toS4  toS5  toS6  toS7  toS8  toS9 toS10 toS11 toS12
fromS1  0.934 0.009 0.003 0.031 0.000 0.000 0.001 0.000 0.001 0.007 0.001 0.013
fromS2  0.002 0.958 0.007 0.002 0.000 0.000 0.006 0.000 0.011 0.012 0.001 0.001
fromS3  0.002 0.011 0.959 0.001 0.000 0.000 0.003 0.011 0.003 0.010 0.000 0.000
fromS4  0.047 0.002 0.004 0.897 0.018 0.002 0.000 0.002 0.010 0.003 0.008 0.008
fromS5  0.000 0.000 0.000 0.003 0.924 0.035 0.001 0.006 0.002 0.000 0.024 0.006
fromS6  0.000 0.000 0.000 0.000 0.072 0.920 0.001 0.000 0.000 0.000 0.007 0.000
fromS7  0.000 0.013 0.004 0.000 0.002 0.001 0.961 0.000 0.008 0.005 0.001 0.005
fromS8  0.000 0.000 0.008 0.000 0.016 0.000 0.002 0.945 0.002 0.006 0.017 0.004
fromS9  0.012 0.031 0.007 0.000 0.005 0.000 0.024 0.005 0.839 0.026 0.012 0.040
fromS10 0.005 0.011 0.011 0.001 0.001 0.000 0.004 0.008 0.023 0.932 0.003 0.000
fromS11 0.000 0.001 0.000 0.004 0.044 0.007 0.000 0.006 0.002 0.000 0.914 0.023
fromS12 0.009 0.001 0.000 0.000 0.012 0.000 0.009 0.001 0.015 0.000 0.024 0.929

Response parameters
Resp 1 : gaussian
Resp 2 : gaussian
Resp 3 : gaussian
     Re1.(Intercept) Re1.sd Re2.(Intercept) Re2.sd Re3.(Intercept) Re3.sd
St1            0.181  0.150          -0.768  0.174           0.225  0.433
St2           -0.800  0.100          -0.774  0.170           0.898  0.395
St3           -0.796  0.076          -0.771  0.163          -0.293  0.556
St4            1.309  0.864          -0.699  0.209          -0.373  0.627
St5            1.206  0.409           1.291  0.618          -0.688  0.786
St6            2.838  0.982           2.775  1.101          -1.563  0.790
St7           -0.764  0.123           0.287  0.873           0.778  0.555
St8           -0.045  0.263           0.039  0.391          -1.103  0.508
St9           -0.315  0.126          -0.357  0.364           0.542  0.532
St10          -0.537  0.066          -0.536  0.168           0.333  0.656
St11           0.553  0.148           0.592  0.382          -0.412  0.768
St12           0.180  0.125           0.156  0.387           0.408  0.508
converged at iteration 171 with logLik: -18388.45
```

Figure 4: Summary(fitmodel) of the selected model (12 states)

Task 3

To test for anomalies in the 3 datasets given, we took the model selected in task 2, and tested it on the datasets the same way we had tested the model in task 2. We first had to subset the data the same way we had when training the other model, selecting only data from Thursdays between 5pm and 8pm, and also taking only the columns Global_intensity, Global_active_power, and Voltage. The data is also scaled before it is given to the depmix function to make the new models.

We then created a new model for each dataset, and copied the parameters from the model selected in part 2. Testing these models on the datasets with anomalies and extracting the log-likelihood values from each, we see that the first and second datasets have slightly lower log-likelihoods than the original dataset (a difference of about 3500), but the third dataset has a much lower log-likelihood being 13600 or so lower than the original dataset. A lower log-likelihood is attributed to a higher degree of anomalies in the dataset, meaning that of the datasets with anomalies, dataset 3 seems to have a notably higher degree of anomalies than the first two.



```
> print(trainingLogLike12)
[1] -20159.69
> print(anomalyDataTest1$logLike)
[1] -23599.84
> print(anomalyDataTest2$logLike)
[1] -23599.84
> print(anomalyDataTest3$logLike)
[1] -33722.39
```

Figure 5: The log-likelihood for each of the three datasets with anomalies

Conclusion

Every assignment and this term project had data files that contained thousands of data points and while initially we questioned the need, it replicated the situation of real life companies that deal with the information from millions of people. It is easy for anomalies to slip by when working with this much information but a small slip could result in major issues. In this assignment we utilized Principal Component Analysis and Hidden Markov Models to learn how to discover anomalies. Using these methods we discovered that the third data set contained a higher degree of anomalies than the other two datasets. Through proper detection, the response time and strength of the security in the technology field will increase and create reliable safety precautions.

References in MLA Format

Uwe Glässer,Mahsa Keramati and Seyed Amir Yaghoubi Shahir. Simon Fraser University, Spring 2022, Simon Fraser University, Burnaby. Lecture and Tutorial.