Anomaly Detection On a Power Grid (Critical Infrastructure Protection)

Anuvrat Sharma - Student # 301321176

Sheil Mehta - Student # 301307105

Abstract:

This project attempts to make us use our findings to generate a Hidden Markov Model that would then be used to detect anomalous behaviour in three other datasets. We were under the assumption that the original dataset was similar to the other three datasets, barring some anomalous injections.

The project report was divided into three main components.

1. Feature engineering, which was essentially deciding what the most important components in the data set were, in order to use them for our hidden markov model.
2. HMM (training and testing), this had to do with the creation of a suitable markov model, that would then be used to detect anomalies in our other datasets. This component consisted of both training a handful of models, as well as testing them in order to see which one of them performed the best
3. Anomaly detection. By using the resulting Hidden markov model, it was applied to three unknown datasets, which supposedly contained anomalous records. The three datasets were ranked and compared, and was discovered if they contained anomalies and to what degree.

Table Of Contents

Problem Scope

The data set used at the beginning of the project contained the measurements for the electric power consumption of a single household with a sampling frequency of one minute from December 16th, 2006 at 5:24 PM to December 1st, 2009 at 2:07 PM. This data set was assumed to be normal and not contain any anomalies. Using the normal dataset, a model was created that described the expected (or normal) behaviour of electrical consumption on a power grid, and how each variable affected the others.

Lastly, the model that was generated was used on the three other data sets that were provided. These three data sets were said to have been injected with anomalies, and using the same criteria that was used before, the data sets were judged whether they contained more anomalies than the others.

All the code was done in R, to create a Hidden markov model and detect anomalies in the given datasets.

Background

Anomaly detection is a process in which patterns or events that do not seem to behave in an expected manner are identified. It is common to see this technique in the real world when it comes to finance, medicine, IT and many other fields. When one talks about anomalies they may also use words such as outliers, deviations or exceptions. Practical applications of anomaly detection include noticing financial fraud (such as stolen banking information), identifying problems with the body (such unusual biological behaviour), and many more such examples.

Data mining is a field that is growing rapidly within the technological sphere and its applications are numerous. The hidden markov model builds on the concept of markov chains, except in this case, the system being modeled assumes unobservable states, where one process depends on another, by a certain probability.

This project allowed for some experience in the fields of cybersecurity, and data mining.

Methodology

For this project, the two main packages used and installed were tidyverse and depmix. Depmix was used to help train the models, and provide us with log-likelihood and BIC.

The dataset was staged in R, by removing noise values, and filtering a section of the data that corresponded to the day and timings that were selected by the team.
The data set “TermProjectData.txt” was split into a training set and a testing set. The model was built to be used on the files “DataWithAnomalies1.txt”, “DataWithAnomalies2.txt” and “DataWithAnomalies3.txt”.

After this, the data set was analysed, to determine the most important components, with which to build the model upon. Once the variables were selected, the creation of a Hidden Markov Model began. The models assumed a different number of states, starting from 4, up to 15. The different models were then assessed based on a specific criteria, and then tested on the training set of the dataset. Finally, the model generated was used to detect anomalies in the anomalous datasets, by determining how similar it behaved to the original.
The full method will be elaborated upon below.

Description of Data Set

The normal data set that is used in this project for training and testing is the consumption of electricity from December 16th, 2006 at 5:24 PM to December 1st, 2009 at 2:07 PM from a single household (Source). The entries in the data set are taken every minute (a sampling frequency of one minute).

Every entry (minute) in the data set contains the following 9 attributes:

- Date: The date in the form of a character string and the format of DD/MM/YYYY.
- Time: The time in the form of a character string and the format of HH:MM:SS.
- Global Active Power: Average active power (measured in kilowatt) for that minute.
- Global Reactive Power: Average global reactive power (measured in kilowatt) for that minute.
- Voltage: Average voltage (measured in volts) for that minute.
- Global Intensity: Average intensity (Measured in ampere) for that minute.
- Sub Metering: The active energy consumption per minute in specific locations of the household. Each of the three sub metering variables represents a different section of the household with different appliances.
    - Sub metering 1
    - Sub metering 2
    - Sub metering 3

The data set contained 1556444 instances. Due to the fact that some of the entries in the set were missing values for certain attributes (Global Active Power and Global Intensity) that contained important characteristics they were removed (infinite

values).

This was done using the na.omit() function on the database, leaving us with 1,548,072 instances.

After those desired values were removed in order to create the model the set was divided into training and testing data. With around 80% being training, while 20% was test data.

Analysis and Results of Experiments

Part 1: Principal Component Analysis

The principal component analysis, or PCA is the process of taking a data set, and transforming it into just one or two of the main variables that affect the data. This is more useful than a correlation matrix since it does not contain so many dimensions that are difficult to visualise. Principal component analysis was used to simplify the dataset and reduce the number of dimensions in the dataset. In the case of the data set provided, one of the components accounted for around ~40% of the variance of the entire dataset. The variance can be found in the second row of the table below.

Importance of components:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
Standard deviation	1.6701	1.2352	0.9956	0.8813	0.77178	0.49381	0.27845
Proportion of Variance	0.3985	0.2180	0.1416	0.1110	0.08509	0.03484	0.01108
Cumulative Proportion	0.3985	0.6164	0.7580	0.8690	0.95409	0.98892	1.00000

After discovering the weightage of PC1, the loading scores and variance were calculated for each of the variables.

The data revealed the top 2 variables in this case were ‘Global_intensity’ and ‘Global_active_power’, with the pair accounting for loading scores of ~0.56 and ~0.48.

This was a strong indicator that ‘Global_intensity’ and ‘Global_active_power’ were the two most important variables in the dataset, to create a Hidden markov model with.

Given below are the results of the R program, showing the importance of each variable in descending order.

Loading scores for each component.						
Global_intensity	Global_active_power	Sub_metering_3	Voltage	Sub_metering_1	Sub_metering_2	Global_reactive_power
0.559597	0.4687199	0.3874711	0.3305256	0.2988388	0.2837926	0.1947535

The above is the formatted output in descending order, while below is the raw output from R studio.

```
Standard deviations (1, .., p=7):
[1] 1.6911213 0.9991566 0.9697654 0.9133066 0.8777405 0.6860557 0.3551337

Rotation (n x k) = (7 x 7):
                              PC1          PC2          PC3          PC4          PC5
Global_active_power    -0.4687199  0.13475701 -0.087364024 -0.06854240  0.26144877
Global_reactive_power -0.1947535 -0.74422839  0.166001666  0.60786960  0.06446593
Voltage                 0.3305256 -0.13245740 -0.035064907 -0.13266015  0.91900003
Global_intensity       -0.5595970  0.01912909  0.001155733 -0.06409154  0.13818872
Sub_metering_1         -0.2988388 -0.12874880  0.728446337 -0.47839198  0.04294132
Sub_metering_2         -0.2837926 -0.41294122 -0.651209714 -0.42742059 -0.05496931
Sub_metering_3         -0.3874711  0.47218329 -0.094190943  0.43879699  0.24282899
                              PC6          PC7
Global_active_power    -0.76918472 -0.29968496
Global_reactive_power -0.03290397 -0.07677664
Voltage                 0.08172722  0.05602833
Global_intensity        0.08117594  0.81036445
Sub_metering_1          0.24064565 -0.27362731
Sub_metering_2          0.28686667 -0.23846336
Sub_metering_3          0.50378618 -0.33574973
>
```

PCA is a great way to reduce computing time and also preserve data integrity, by focusing on only the ‘main’ variables that decide the behaviour of a dataset.

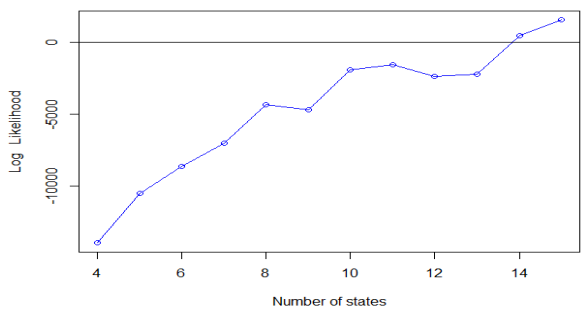
Part 2: Multivariate Hidden Markov Models

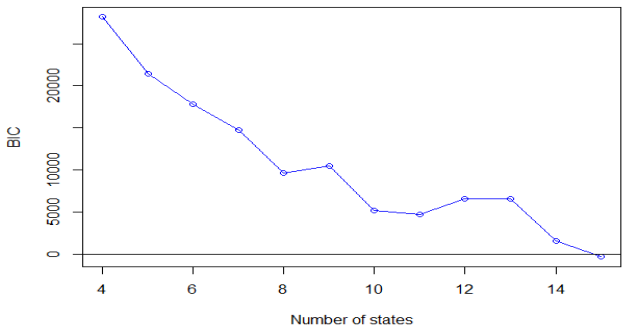
With PCA done, The next step was to train the multivariate HMM based on the results of the principal component analysis. Only two components were chosen, ignoring the rest of them when creating the hidden markov model.

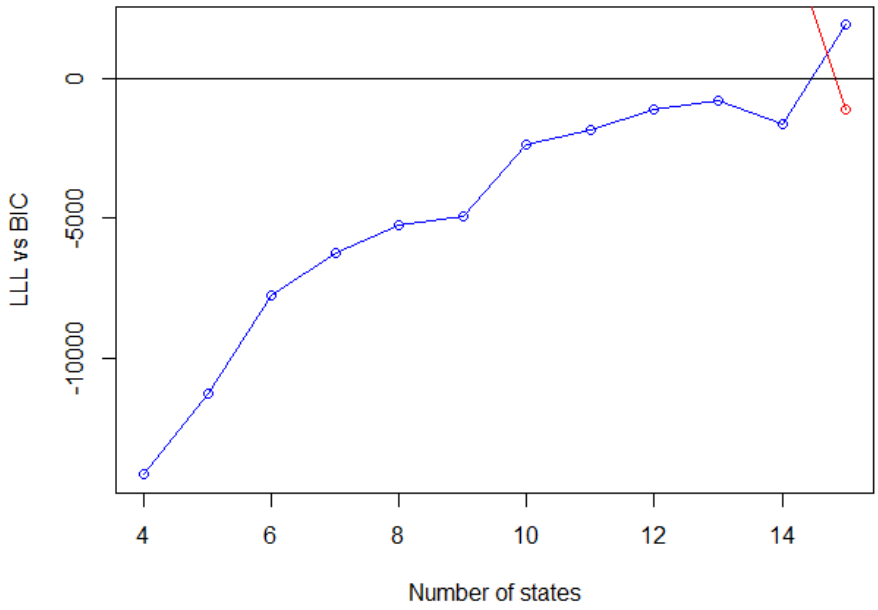
After settling on the two important components, the data set was partitioned into a test and training set, and the data was filtered for only a chosen time frame. The selected time frame was Tuesdays between 9 and 11am. For this time window, various multivariate hidden markov models were trained, each with different states starting from 4, up to 15. The dependent variables based on our PCA analysis were: Global_intensity and Global_active_power. The 'gaussian' distribution was used for Global_intensity and Global_active_since they are both continuous data.

For each of the markov models with differing states, the Bayesian information criterion (BIC) and Log-likelihood were calculated. These two were the criteria used to assess the models, when fitted onto the test data. Models with lower BIC and higher log-likelihoods are preferred. This was obviously a challenge, as there was always going to be a trade-off when deciding which criterion to favour.

The log-likelihood and BIC was calculated and graphed for the models generated on the test data, to uncover what would be a good starting point for deciding an appropriate number of states.







The two metrics were graphed together, to get a better understanding of their behaviour relative to each other. This revealed an intersection of their points at around 14 to 15 states. This could represent an ideal region for when the model performed most optimally. Since there were two points in question, both of them would be tested on the test data, to find which of the two performed better. This was done to make sure that the probabilistic model that would be generated, was a good representation of how the normal behaviour would look like.

Part 2.1: HMM Testing

The models generated assuming 14 and 15 states were tested against each other. This was done by using setpars and getpars to fit the models onto the test split of the data. Once this was done, the log-likelihoods of the test and train data were compared. The one with the lower difference between its two log-likelihoods would be the more accurate model. In this case, it was the one with 15 states.

	States = 14		States = 15	
	train n =14	test n =14	train n =15	test n =15
Log Likelihood	-1419.943	-2531.32	2529.253	-1894.838
Abs Log Likelihood	1419.943	2531.32	2529.253	1894.838
Difference	1111.377		634.415	

Part 3: Anomaly Detection

Now with a trained model now verified, the markov models could be used in anomaly detection, to detect any anomalies or patterns that deviate from the normal behavior in the other data sets. Once again, Log-likelihood was used as the method to assess the similarity of the new data set with the original. The lower the difference between log-likelihood, the closer the data set is to the original.

The anomalous datasets were fed to an instance of the trained model, using scaled variables.

	Train	Test	Anomaly 1	Anomaly 2	Anomaly 3
Log-likelihood	-1419.94347	-2531.319528	-50754.56147	-50754.56177	NaN

Above is the output from running the chosen model on the different datasets and subsets.

Both Anomaly 1 and 2 are skewed far from the original, however interestingly they produced similar outputs. On the other hand, Anomaly 3 didn't give us any results, which could be due to the fact that this data set was very different from our original, and did not fit our model at all.

Conclusions and Reflections

Anomaly detection using machine learning techniques can be a very powerful tool. However there are many compromises and there are several trade-offs to be made.

By doing this project, we learned that right from the beginning, cleaning and processing our dataset is very important. Additionally, using techniques such as PCA are imperative to realistically create a workable model.

We found that there could be room for improvement in the following areas:

- We could have given more thought to choosing our time interval, by diving more into the dataset. We could have visualised the data points using graphs.
- We could have rounded the number of days to the closest full day to 154 instead of using 154.8 days. This would include getting rid of certain days in which there wasn't complete data for the 2 hour time interval
- We could have tried more states, for the Hidden Markov Model, and maybe tested them out as well.
- Our anomaly data seems to be very skewed,and could be investigated more. The NaN value for anomaly 3 could possibly have been due to another error that is not related to the data set.

For the future, we could possibly try different time intervals and different days of the week, to have a more conclusive assessment of such a dataset.

References

https://archive.ics.uci.edu/ml/datasets/individual%2Bhousehold%2Belectric%2Bpower%2Bconsumption.

Contributions

Anuvrat Sharma: Contributed to Part 1 of the code (Feature Engineering) and Part 2 (HMM Training and Testing). Worked alongside Sheil to complete the presentation and the report. Worked in tandem so that both parties were up to speed with what was happening.

Sheil Mehta: Contributed to Part 2 of the code (HMM Training and Testing) and Part 3 (Anomaly detection). Worked alongside Anuvrat to complete the presentation and the report. Worked in tandem so that both parties were up to speed with what was happening.

Both parties also made sure to come to decisions together, about what time interval to choose, how many states to choose, what to graph etc. The report and presentation was also worked on together, on the off chance that the Instructor or the TAs asked any questions about our project, both would be able to respond appropriately.

Constant communication was required throughout the working of this project, and teamwork was essential for its completion.