# Machine Learning Assignment – Anomaly Detection

After applying basic EDA, I figured out that the data is univariate time series and the anomaly is Collective anomaly.

For time series anomaly detection and the type of data I have, several of the methods are as follows:

1. LSTM(RNN)
2. Auto-encoder (LSTM or Convolution)
3. Arima
4. One-Class SVM
5. Real time detection with HTM and NAB
6. Isolation Forest
7. Bayesian Network and Naïve Bayes

I decided to implement **LSTM**.

## Why LSTM?

As, per the EDA, it suggests that the data is anomalous only with certain pattern i.e. Collective Anomaly and there is dependence of data on previous data steps. The sequential dependence in data is well analysed using LSTM because of its ability to learn long-range pattern in the data.

After training the LSTM model it can perform well in Unsupervised fashion on real time data too which is plus point.

The data provided was CPU utilisation of AWS EC2. For that purpose, I thought of using LSTM to get a stable and better model for long term implementation. We could have used normal neural network but as the dependencies have pattern and in long run we need to focus more on remembering how collection of data changed. So, for that purpose implementing LSTM beforehand is the safest choice.

## Few Key Notes Regarding the Approach

## (Please Look through Complete Assessment of Approach, Python notebook for analysis)

1. The observation by EDA is also that there is no seasonality in the data. Therefore, dropping the time-stamps from the data won't harm us. It's not a feature of interest for calculating the anomaly.
2. Feature Engineering – Applied several combinations on the LSTM and conclusion is that "moving average" on the data makes it more clear for understanding the data by the model.
3. F1 score and accuracy of several combinations can be seen in the Python notebook attached with the report. Following is the F1 score and Accuracy of the method I chose.
   F1 score =0.19
   Accuracy=0.89
4. As, the data have collective anomaly splitting the data in general test-train split will destroy the pattern which implies that splitting the data and validation process is not useful on the data given.

## Conclusion

- If we get more data to train the model – “LSTM on moving average” and run the model for larger epochs we can improve the F1-score by large amount and we can keep the accuracy of the model stable.

## Note-

I wanted to try out synthetic data generation to achieve to repeat the pattern on random fashion using for loop or randomised generation of pattern using pattern in the data. But, dint got opportunity to do so. Else increasing the data and reproducing the pattern again and again using a function would help to tune the model more and train it more effectively as LSTM requires good amount of data for stability