ANM Project
DANM! Team

Fabien RAVET
studentNo
email@mails.tsinghua.edu.cn

Aleksander
studentNo
email@mails.tsinghua.edu.cn

Aurélien VU NGOC
2020280219
vunaa10@mails.tsinghua.edu.cn

main guideline: start simple and build more advanced method later depending on the previous results and performance

1 Exploratory Data Analysis (EDA)

1.1 Trace

trace: very large data that cannot be imported as a single dataframe.

- Use dask.dataframe package. it's an memory-optimized version of pandas' dataframe[1]
- Divide and conquer approach: divide the data into chunks using the read-csv(..., chunksize=10000) option implemented in pandas. However, one needs to concatenate all the chunks in the end. Unless the processing method can be applied to different chunks separately, this method is not useful
- reduce-mem-usage function used many times in kaggle competition[2]. However, trace dataframe has few numerical data (int or float), so this trick is not very effective because it only reduces the memory usage of numerical data. There can also be the solution to transform object into pd.Categorical, which is less stressful for the memory. This would be a way because we know there is a limited amount of unique values in the columns.
- Since the trace data is only used for root cause after anomaly detection, we can imagine loading only a part of it (around the same timestamp as the detected outlier) to free up memory

1.2 esb

esb has 3 main components:

- start time: start time of the event
- avg time: average time of processing
- succee rate: sucess rate

Findings: unfortunately, succee rate only has 1.0, which means in all the collected data that can be used for training, there is no anomaly that can be detected through success rate, as suggested in the project description. We have the following question: is it because there is no anomaly, or is it because anomalies are meant to be detected using another methode ? In any case, we tried to detect anomalies using avg time.

[1] dask.org
[2] e.g. www.kaggle.com/gemartin/load-data-reduce-memory-usage

2 Anomaly detection

2.1 What input should be considered during the anomaly detection process ?

Since we are given a lot of performance indicators, which one is the best for the anomaly detection process ? Let's start with avg time, which seems a good indicator. We imagine, if the average time of processing rises above a given threshold, then:

- probabilistic approach: "chance of anomaly are high"
- direct approach: "there is an anomaly"

2.2 Statistical methods

2.2.1 Exponential smoothing, ...

Simple exponential smoothing, Holwinter exponential smoothing (triple exponential smoothing), and derived methods cannot be applied here because the data is not periodic. Nevertheless, we tried to implement these methods anyway by assuming different frequencies for the time series. Since the data is collected every 1 minute, and that we are given 24h of data, we can assume an hour periodicity, half-an-hour periodicity, 3h periodicity... Besides, because the anomalies must be 10 minutes apart (according to the project description), maybe a periodicity of 10minutes is to be considered ? [3]

2.2.2 Simple moving average, moving std

We tried to implement a simple moving average and a simple moving std (standard deviation) to tackle the detection problem. [4]

2.2.3 ARIMA

ARIMA = AutoRegressive Integrated Moving Average

- simple ARIMA with all the data
- iterative ARIMA (rolling forecast ARIMA model)

[5]

2.3 QUESTIONS

which one is good ? how do we check if the model correctly detect anomalies ?

2.4 Unsupervised learning methods

Because we have no labels in the data (unless we label it by hand, but we would be left with only few data), it is impossible to use supervised learning in order to achieve anomaly detection. [6]

2.5 Hybrid methods

Why not use both methods and apply them to different Maybe combined with a probabilistic approach with every method giving a probability of anomaly, and then by combining all the results, we can determine if there is actually an anomaly. This will allow us to weight the models (e.g. if ARIMA says that there's an anomaly, it is most likely correct compared to Decision Tree saying there is no)

[3] `https://medium.com/@jetnew/anomaly-detection-of-time-series-data-e0cb6b382e33`

[4] `AutoRegressiveIntegratedMovingAverage`

[5] `https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/`

[6] again: `https://medium.com/@jetnew/anomaly-detection-of-time-series-data-e0cb6b382e33`

3 Root cause

3.1 QUESTIONS

what model should be used ? How can I see a root cause myself ? What does it look like ? (meaning, what if I am the algorithm that is trying to find the root cause) supervised learning methods ? (==> no labels unfortunately...)