Data Mining Report

Leonardo Chiani, Fabio Comazzi, Andrea Deretti, Michele Russo, Luca Zerman

November 4, 2022

1 Introduction

The goal of this project is to perform anomaly detection on a data set consisting of approximately 11 million observations of 34 features related to data transfer cells. In particular, we are required to compute an anomaly score in $[0, 1]$ for each observation of the data set (which will be interpreted as the probability of such observation being an anomaly).

2 Data Processing

2.1 Inspection of the Features

We grouped the features on the basis of their type and values as follows:

1. **Feature** `ts`: this feature represents the time associated to the observation. Therefore, we cast it to `datetime`;
2. **Features** `node_id`, `node_name`, `inteface`: `node_id` and `node_name` assume 49 different values each, representing exactly the same information. For this reason we decided to keep only `node_id`. `interface` is also a categorical variable which assumes 2094 different values;
3. **Features** `Bits IN (High Speed)`, `Bits Out (High Speed)`: these are continuous variables with non-negative values of type `float64`;
4. **Features** `KPI_Discard_IN_1`, `KPI_Discard_IN_2`, `KPI_Discard_IN_3`, `KPI_Discard_IN_4`: these four features only assume the values 0 or NaN;
5. **All the other KPI's**: these are continuous variables with non-negative values of type `float64`.

2.2 Features Extraction

Given the previous observations, we started trying to extract meaningful information from the time variable; in particular, we focused on understanding if the `hour` component of the `ts` attribute was significant. In order to do so, we analyzed the scatterplots `Bits IN (High Speed) - Bits OUT (High Speed)` for each node highlighting the `hour`'s influence. As we can see in Figure 1, the `hour` seems to be a relevant factor as the colours are not evenly spread across the plot. Therefore, we decided to add `hour` as a feature.

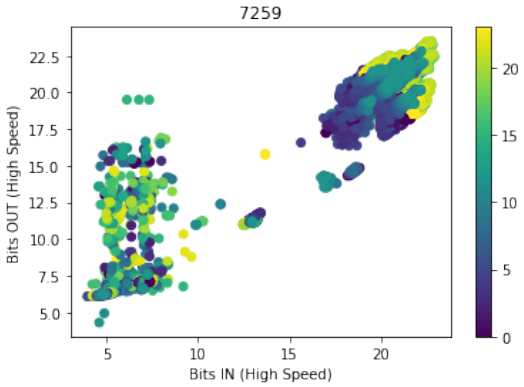

Figure 1: "Bits IN (High Speed)" vs "Bits OUT (High Speed)" in log scale for node with ID 7259, where the colours are associated to different hours of the day (from 0 to 24).

Another remark we can draw from such plot (without considering the different colours) is that different clusters are clearly visible, as if to say that it is possible to further specialise our analysis. In order to do so, we decided to work separately on every `node_id-interface` pair. Indeed, as it is evident from Figure 2, in general there is only one dominant cluster associated to such pairs.

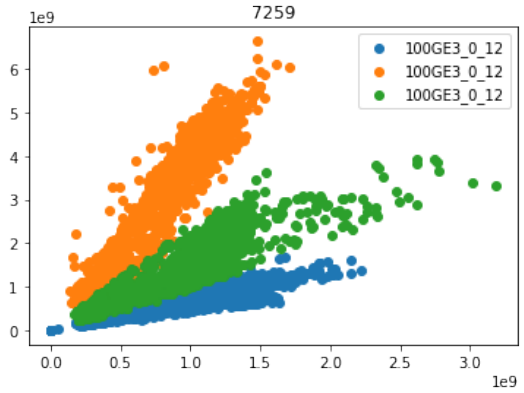

Figure 2: "Bits IN (High Speed)" vs "Bits OUT (High Speed)" for node with ID 7259 and and three different interfaces.

2.3 Infinite and Missing Values

While exploring the data, we found around 70 **infinite values** distributed on 45 observations in the data set. Given such anomalous values, a possibility would be to classify such observations as outliers. Since, however, we lack sufficient information to understand the meaning of the value `inf` in this context, we decided to simply treat them as "high values". In particular, we replaced them with 5 times the maximum value of the respective feature. In this way we are able to treat the corresponding observations at the same level of the others, which allows us to compute their anomaly score with the methods we are going to describe in the following.

Let us now focus on the **missing values** of the data set. At first, we observed that there are 59805 rows in which all the numerical features are equal to NaN: since they are extremely non-informative and also too many to be considered as anomalies, we decided to flat assign them an anomaly score equal

to 0. Moreover, we also detected 2147697 rows in which all the KPI's are equal to NaN: with the same reasoning, we assigned them an anomaly score equal to 0. Hence, these rows were not taken into account any longer for our analysis.

From a feature's point of view, we first considered the variables `Bits IN (High Speed)` and `Bits OUT (High Speed)`. In particular, we realized that if such features are equal to NaN then all the others of the corresponding observation are also equal to NaN: thus, we already dealt with such observations in the previous paragraph. For what concerns the `KPI_Discard_IN` features (which only assume the values 0 or NaN), we observed that they cannot be considered to be Missing Completely At Random, as shown in Figure 3.

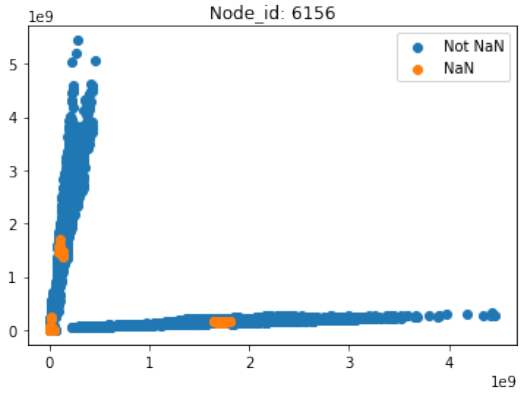

Figure 3: "Bits IN (High Speed)" vs "Bits OUT (High Speed)" for node with ID 6156, where the colour is associated to the presence of NaN in feature "KPI_Discard_IN_2".

Therefore, we first decided to impute their missing values by replacing them with a value equal to 1. Since, however, our final approach consists in performing a Principal Components Analysis on the numeric features of the data set, we later decided to drop these columns as they essentially represent categorical variables. We deem this will not compromise the final results as these features are not particularly informative.

All in all, for what regards the other numerical features, we decided to impute their missing values with a regression model as they are continuous variables. In particular, we resorted to the **XGBoost algorithm** because of its ability to deal with missing values in the input attributes and its convergence speed. We then trained 24 different models, each one having a specific feature as the target variable and the remaining ones as regressors. We trained each model on 500000 observations randomly sampled from the original data set and validated over 500000 other observations, again sampled at random. Since the Mean Percentage Squared Errors were lower than 15% in the worst case and lower than 1% in most cases, we considered these results to be satisfying.

3 Anomaly Detection Pipeline

With all the missing values being imputed, we can finally compute the anomaly score for each observation of the data set.

As mentioned before, we split the data set in subsets pertaining a specific node-interface pair. Our approach consisted in performing a dimensionality reduction via **Principal Component Analysis** on each one of these subsets after they have been standardized. In particular, we decided to retain only the Principal Components necessary to explain at least 90% of the variability of the data. After that, we projected the data points on the space of reduced dimensionality and we reconstructed the projected

points on the original space. All in all, we computed the anomaly score for each observation as the corresponding squared reconstruction error normalized by the largest value obtained.

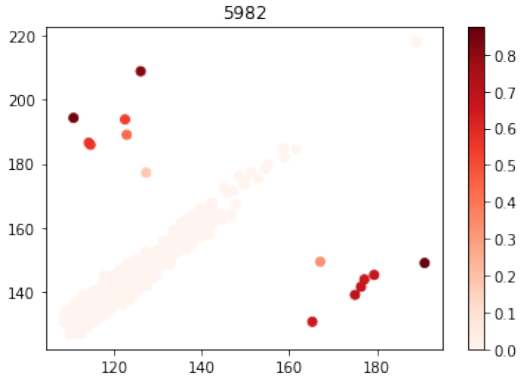

Figure 4: "Bits IN (High Speed)" vs "Bits OUT (High Speed)" for node with ID 5982 and interface "xe_4_2_2_136", where the colour is associated to the reconstruction error of the data point.

The idea underlying our approach is to use Principal Component Analysis to learn an encoding of the data points in a lower dimensional space. This encoding considers the distribution of the majority of the data and inherently ignores the anomalies. In this way the observations with a high reconstruction error will be the ones which do not follow the data distribution and thus can be considered as anomalies.

We point out that Principal Component Analysis just performs a linear encoding (rotation) of the data. Another interesting approach would be to perform the same analysis with some non linear encoder such as a Neural Network.

4 Analytical Results

Since this is an unsupervised task, providing an assessment of the obtained results is quite difficult. It is, however, interesting to consider the distribution of the anomaly scores obtained for the whole dataset. In particular, Figure 5 shows that the vast majority of the observations are ranked with a pretty low score (approximately 0) whereas few data points (less than 100) have a score greater than 0.3.

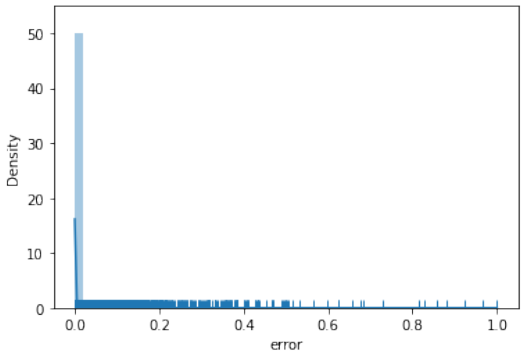

Figure 5: Density plot of the anomaly scores.