Data Challenge: Data Augmentation for Rare Events in Multivariate Time Series*

Chitta Ranjan[1], Mahendranath Reddy[1], Markku Mustonen[1], Kamran Paynabar[1], and Karim Pourak[1]

{cranjan, mreddy, markku, kpaynabar, kpourak}@processminer.com
715 Peachtree Street N.E. 100, Atlanta, GA 30308.

Abstract. A real-world dataset from a pulp-and-paper manufacturing industry was provided in [1]. In this Data Challenge, we provide a data from the same source with a few differences with a focus on developing data augmentation techniques. The data contains binary labels corresponding to sheet breaks in a paper mill, and is extremely unbalanced ($\tilde{0}.7\%$ positive labels). The objective of the data challenge is to develop a data augmentation approach for a binary labeled rare event multivariate time series data. A guideline for using the data and method development is provided. The efficacy of a method will be gauged based on the improvement in classification accuracy after data augmentation.

Keywords: multivariate time series · data augmentation · real world data · rare event · classification

1 Problem

Large datasets typically benefit a classification model. However, in an unbalanced data, the benefit is limited by the amount of positively labeled samples in it. In our dataset, the positive labeled samples constitute only about 0.7%. Due to this, building a classifier with high precision and recall is quite challenging.

Typically, over and/or under sampling have been used on this dataset. In this data challenge, we will focus on another technique: **data augmentation**.

Data augmentation is a well known technique where synthetic data are generated to supplement the available training data. For example, in image classification, images are rotated and flipped. This data augmentation technique has been quite successful in Deep Learning and other classification models. However, they do not generalize to a time series data.

This is because the visual integrity of images do not alter with rotations, while for a time series data, any transformation may have an unintended alteration. As a result, the data augmentation of time series data has been typically limited to simple techniques, such as, slicing and time warping.

In this data challenge, we have a dataset from a paper mill. The dataset is taken from the same source as in [1] with a few differences mentioned in

* By ProcessMiner, Inc.

Fig. 1: A typical paper manufacturing machine.

Section 2. This is a labeled data for sheet breaks in the mill. A sheet break is a "rare" event in a mill, which in the industry means when the paper reel under production tears while it is still in the machine. When a sheet break happens, the entire paper machine is stopped, the sheet is taken out, and any required machine fix is done, before resuming the process. This causes significant losses to the mill, and hence, should be prevented.

[1] aimed at developing classification models for this problem. Here, we will aim at developing data augmentation approaches that can further improve the classification models. Therefore, the objective of the problem is: **develop a data augmentation approach for a rare event time series binary labeled data that improves the classification precision and recall**.

2 Data

We have a binary labeled time series dataset for sheet breaks at a paper mill. The column y has labels: 1 for sheet break, and 0 for machine running state.

The dataset has predictors, $x1-x61$. All the predictors are continuous, except $x28$ and $x61$. $x61$ is a binary variable, and $x28$ is a categorical variable. The center and scale of the data is changed, and the variable names are omitted for anonymization.

The observations are taken every 2 minutes, meaning the rows in the data are 2 minutes apart with the timestamp present in column *time*. If, suppose, a break happened at time t, the machine stays in the *break* status for about 30 minutes to 1 hour. Thus, the rows for $t : (t + k)$ will have $y = 1$.

In [1], we removed rows $(t+1) : (t+k)$ after the break at time t. The reason is, adding data $(t+1) : (t+k)$ for training the classifier will result in the classifier

learning to predict a break after it has already occurred (at time t). This makes the classifier poor at predicting a break ahead in time.

Here, we provide two versions of the data,

(a) complete data with all break rows, and
(b) data with consecutive break rows removed and break labels moved 4-minute ahead.

In data-(a), the break rows $(t+1) : (t+k)$ after the first instance of break at time t are present. These rows of data provide information about the process during a break. This can potentially help a data augmentation approach.

Data-(b) is provided to build and test a classifier, f. Additionally, as will be mentioned in Section 3, we are restricting the classifier to a 4-minute ahead prediction. To that end, we have shifted the labels as, $y_t \leftarrow y_{t+l}$, where $l = 4$ minutes in data-(b).

If you desire to do this shift in data-(a), you can move the labels up by two rows, i.e., $y_i \leftarrow y_{i+2}$, where row $i+2$ corresponds to 4 minutes after row i. The label shifting on this data is not always necessary for building a data augmentation approach, and hence, is left to the researcher.

3 Approach

A guideline for the approach is provided in Figure 2. As shown in the flowchart, we will first develop a Data Augmentation approach using data-(a).

To evaluate the efficacy of the approach, we will build a classifier on data-(b), and the augmented data-(b), referred to as f_o and f_a, respectively.

For consistency across different results, we fix the classifier method to **logistic** regression. The parameters for the logistic regression on using `sklearn.linear_model.LogisticRegression` in Python are `class_weight = 'balanced', max_iter = 10000, penalty = 'l1', solver = 'saga', C = 0.01`. If using a different logistic regression package, it is recommended to have similar parameters.

The efficacy will be measured based on the improvement in the classifier accuracy between the original data, f_o, and the augmented data, f_a. The accuracy metrics are: F1 score, precision, and recall. There is a held-out test data on which the efficacy will be evaluated. It is recommended that the researcher also creates a train-test split of the provided data to ensure better performance on the held-out test set.

4 Download link

The data can be downloaded from here. Follow the access instructions on the link.

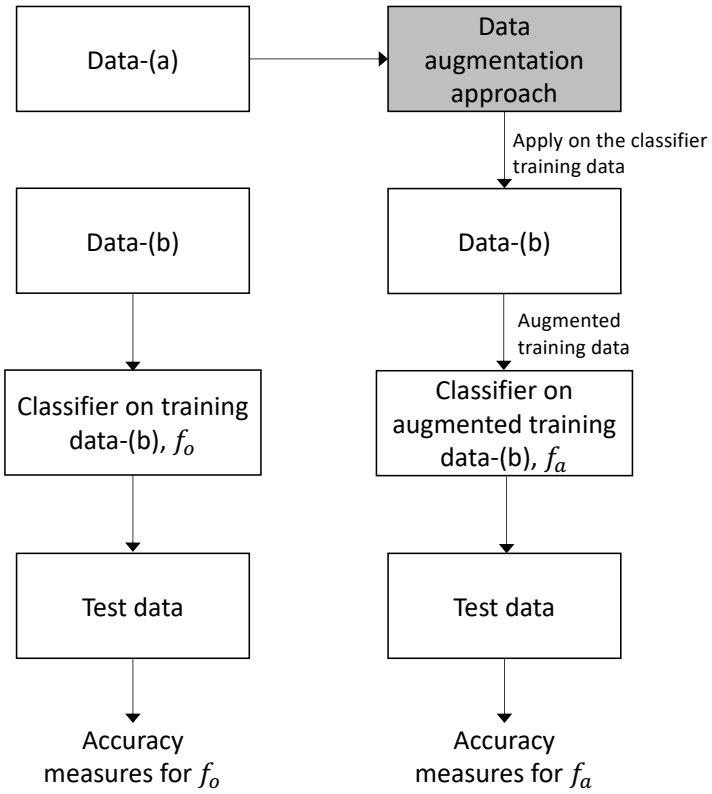

Fig. 2: Flowchart for developing and evaluating a solution.

5 Submission

Submit your code to cranjan@processminer.com and wg152@rutgers.edu. Github or Bitbucket submissions are preferred. You are encouraged to use open source codes. Few open-source resources are given in the references. Please have a ReadMe included that explains 1. how to execute the code, 2. the prediction accuracy measures on your test data (divide the given data into training and testing). The submission will be tested on a held out test data. Your submitted code should be able to directly read a test file and output their predictions.

References

1. Ranjan, C., Mustonen, M., Paynabar, K., Pourak, K. (2018). Dataset: Rare Event Classification in Multivariate Time Series. arXiv preprint arXiv:1809.10717.