Report 6

LI, Ruizhe | 1155076990

Github: https://github.com/rzli6/ML-Storage.git (private)

Target Paper: Improving Storage System Reliability with Proactive Error Prediction

Goal: Realize the method descripted in the paper.

Data Set: BlackBlaze 2017 3rd quarter, 3 months in total.

General Description of the proposed method:

Note that this paper is not so elaborate, there are several challenges we have to deal with. Later I will list out the ambiguous words. In this section, I want to talk about my understanding of the proposed method and its procedure.

First of all, we need to know "what we are going to predict". According to the paper, I use "the number of days between the data is collected and the first error is detected" as my labels. Thus, the first step is to convert our data's collecting date to the number of days before its first error. And because the label is not binary, it is not a classification problem, instead, it is a regression.

After we obtained the labels, the second step should be processing the inputs, or features. Particularly, there are two sub-steps: manipulating the data into *diff, sigma, and var forms,* as described in the paper, and secondly, split the data into two parts: training set and validation set.

Ambiguous words and Challenges:

1. Microsoft Server Installation

   Problem: This is the first challenge I met, and it is still troubling me now. I cannot download and set environment for Microsoft Machine Learning Server, which containing the significant fasttree algorithm.

   Solution: Thanks for my partner Zhilin's help, I finally avoided this problem by using Decision Tree Algorithm. According to the Microsoft's online official guide "rx_fast_trees is an implementation of FastRank. FastRank is an efficient implementation of the MART gradient boosting algorithm." Noting that MART gradient boosting algorithm is implemented as GradientBoostingRegressor in sklearn. Thus, finally I use the GradientBoostingRegressor as the "inefficient" version of rx_fast_trees.

2. "First Error" Definition

   Problem: The term "First Error" is rather ambiguous. It can indicate any disk error,

Smart_5_raw: reallocated sectors count

such as read error (smart 1), reallocate sector count (smart 5), SATA downshift error count (smart 7), so on and so forth, or just simply a disk failure. Indeed, there is a plausible solution to integrate all of these errors and as long as one of it goes up we say an error occurs. However, more than half smart attributes indicate some kind of error. If we use all of them to build labels, then few left to be used as input.

Solution: As this paper is a descendent of ATC 17, I used SMART 5 as the single label indicator. If SMART 5 is larger than 0, then we say an error has occurred on the disk.

3. Accuracy in Feature Selection
   Problem: The feature selection method described in the paper is basically a for-loop. The loop goes through each attribute and monitors the influence of the absence of the attribute. However, the paper does not explain what "better" means. In other words, the metrics used in feature selection is not explicitly shown to readers.

   Solution: Personally speaking, the metric I am using is: keeping NPR under a certain level, and the ones with higher FPR are better.

4. Training and Validation Data Ratio
   Problem: This paper only used one month's data. It may be sufficient if the data is attracted from the extremely huge Microsoft database. However, the only data accessible to us is the BlackBlaze dataset. And it is obviously insufficient for machine learning. For another thing, the paper does not illustrate a reasonable ratio for training and validation. How much data we should use to validate our model is an unanswered question.

   Solution: Up to now, we still do not have a sound solution for this problem. In my implementation, I divided the dataset to 2:1, 2 months for training and 1 month for validation.

Smart_5_raw: reallocated sectors count

Result:

1. Feature selection:

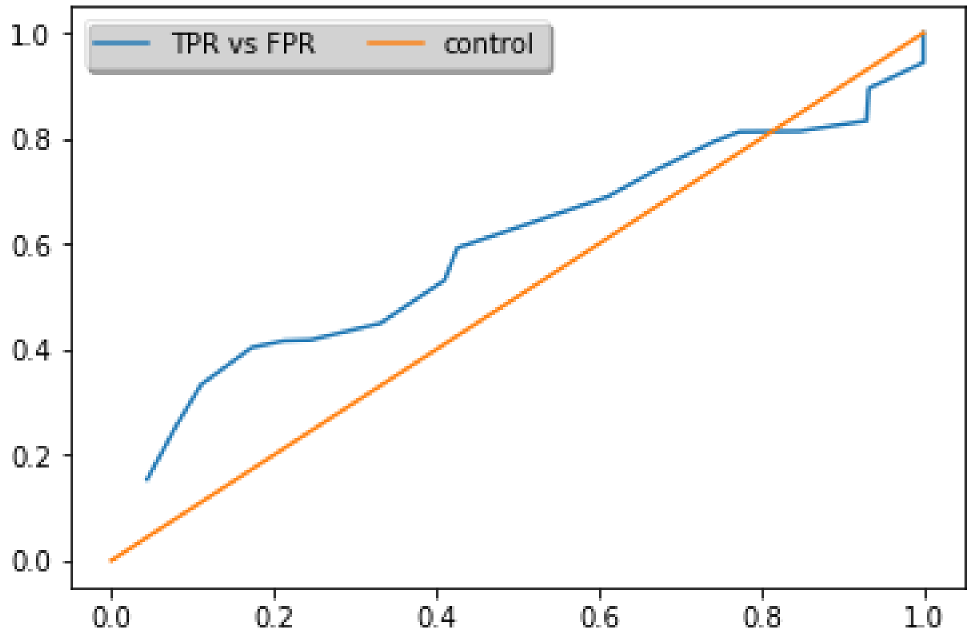

Figure 1: CDEF with proposed features selection

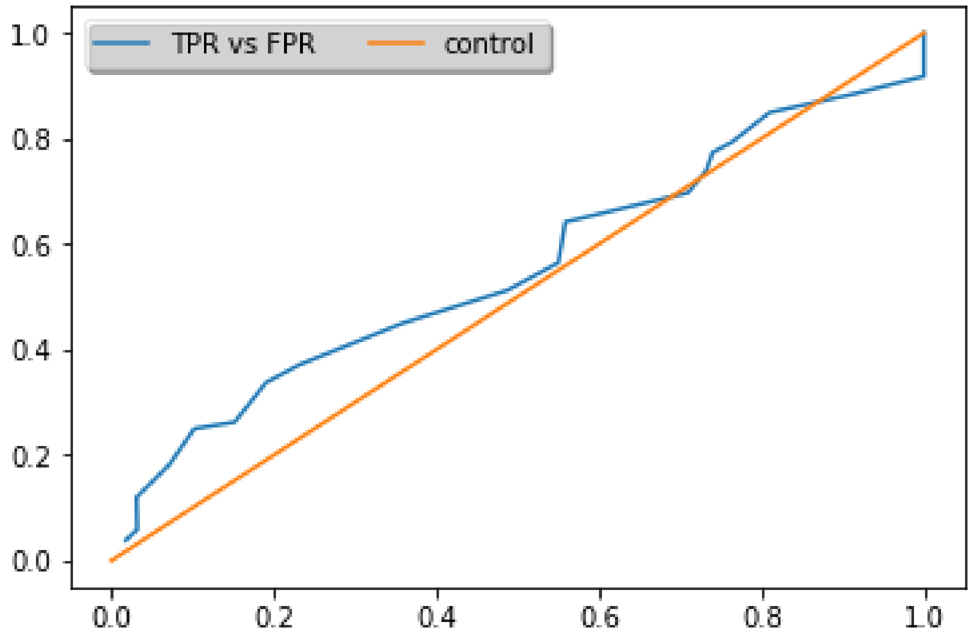

Figure 2: RF feature selection

Smart_5_raw: reallocated sectors count

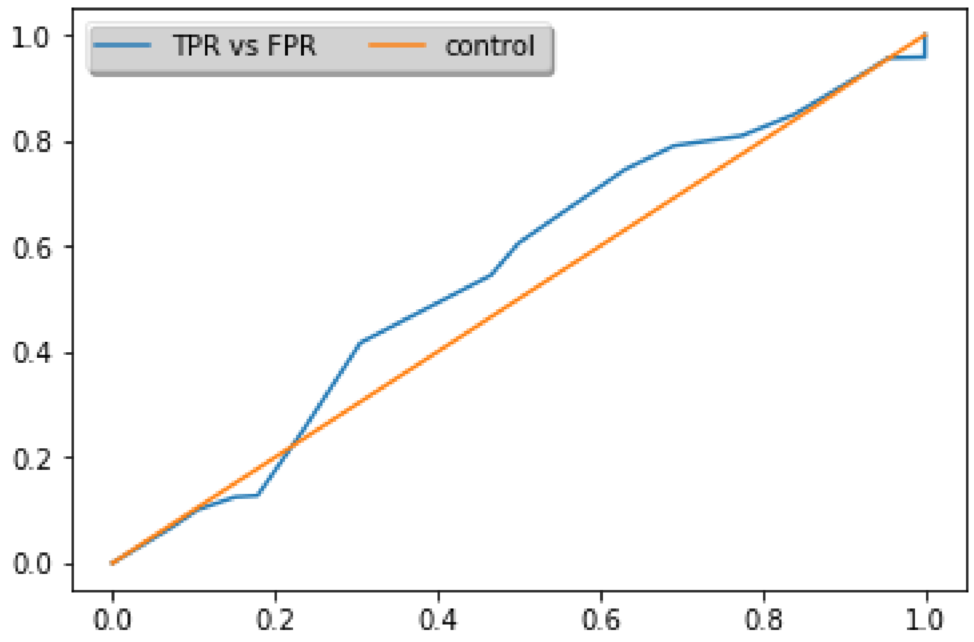

Figure 3: Chi2 feature selection

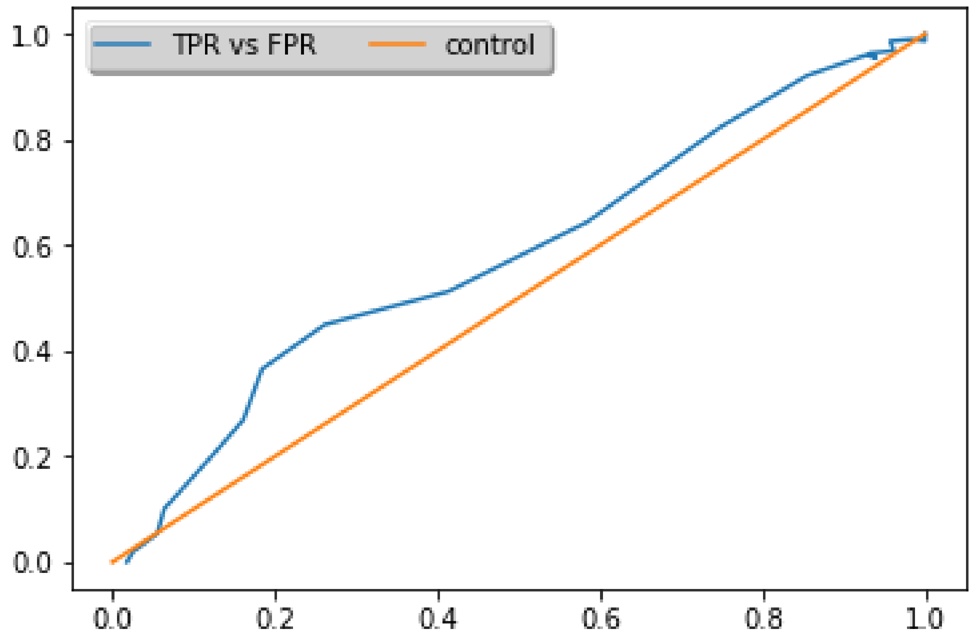

Figure 4: Mutual info selection

Smart_5_raw: reallocated sectors count

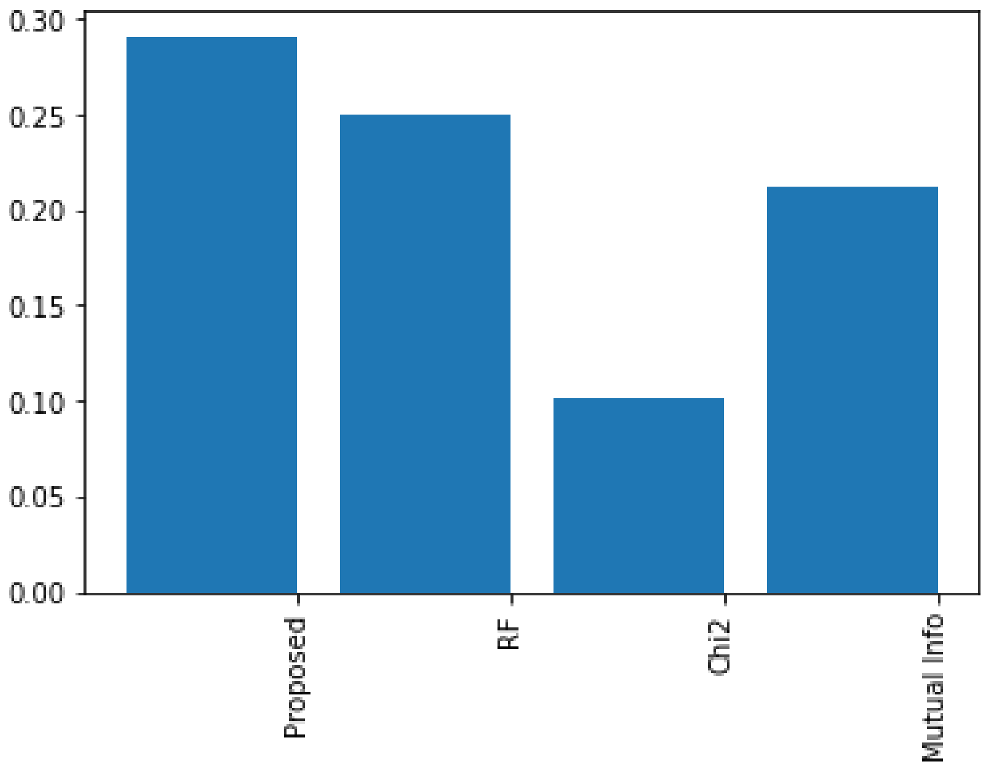

Figure 5: The feature selection comparison

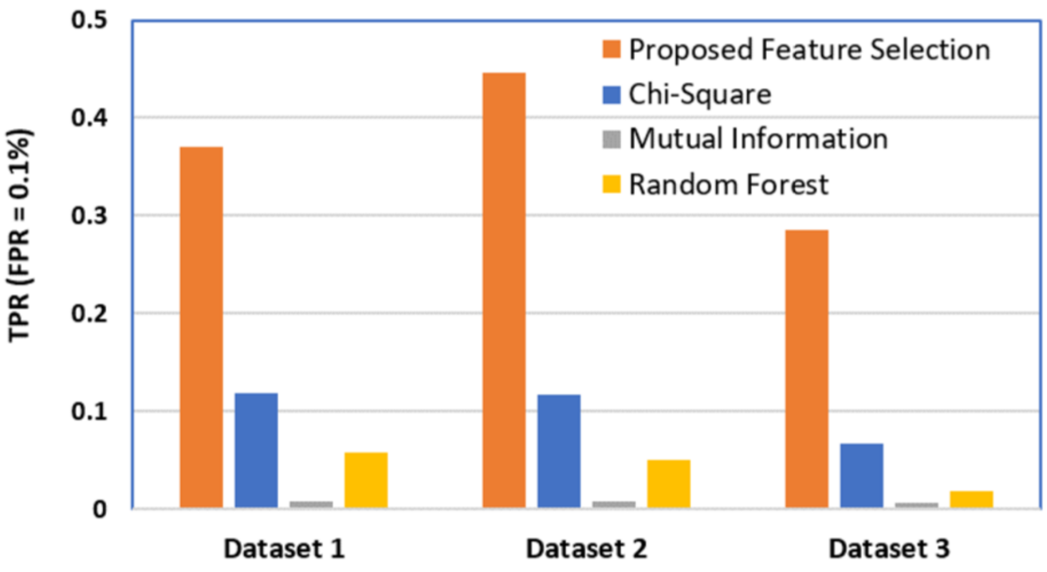

Figure 6: The feature selection comparison in paper

Explanation: As what is shown above, I got a similar result that the "Proposed Feature Selection" achieved a better score than the other three. However, it did not show an advantage as large as shown by the paper. And the overall order was inconsistent either.

Possible reasons:

1. Procedure: As abovementioned, this paper has several points unclear to readers. Thus, the implementing procedure may not be exactly same as the paper.
2. Dataset: The data used in the paper contains both smart attributes and system signals.

Smart_5_raw: reallocated sectors count

And the sizes of datasets are different either.

3. ML method: Although fasttree is an efficient implementation of MART algorithm, it is still possible that there are some optimal steps included in the Microsoft ML Server. Since I used the sklearn GradientBoostingRegression, it might not be implemented in the same way as Microsoft fasstree.

2. ML model comparison:

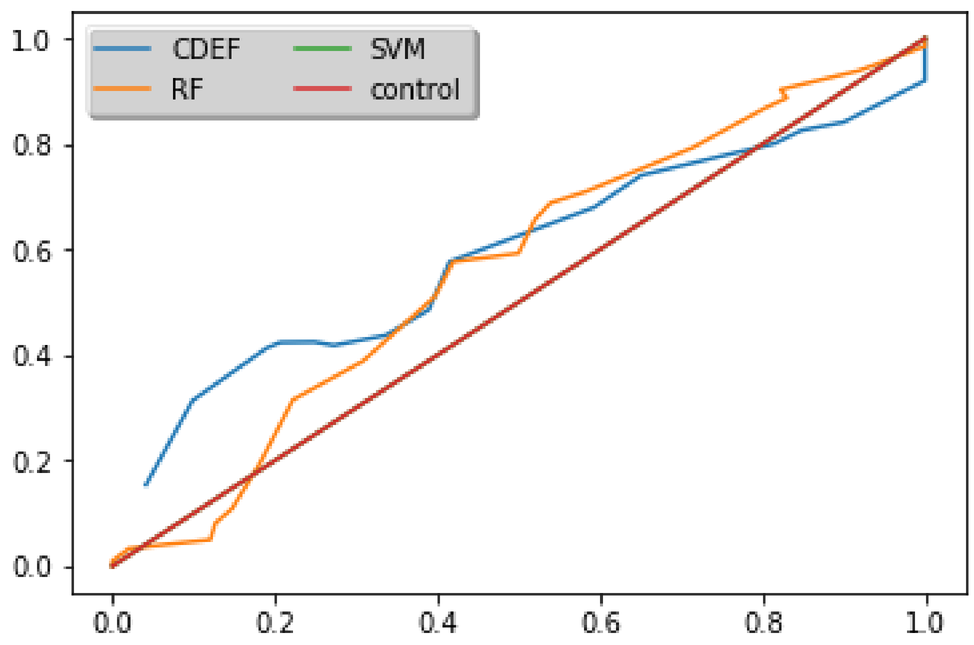

Figure 7: Different model test

Explanation: The result has certain differences with what we get in the paper. The most significant is that the fasttree did not show an advantage in the implementation.

Conclusion:

This paper has several ambiguous points which may affect the implementation in an unexpected way. Thus, the result I got is different from the paper's result.

Smart_5_raw: reallocated sectors count