A Voting-based Robust Model for Disk Failure Prediction*

Manjie Li[1][0000−0002−5815−0605], Jingjie Li[2][0000−0002−8006−7824], and Ji Yuan[1,3][0000−0002−4369−647X]

[1] Shanghai East Low Carbon Technology Industry CO., LTD., Shanghai 200052, China
limanj@elc.cn
[2] Beijing Megvii Co., Ltd., Beijing 100871, China
lijingjie@megvii.com
[3] ERA Group, Technical University of Munich, Munich 80333, Germany
yuanji1239@gmail.com

Abstract. The hard drive failure prediction is a vital part of operating and maintainance issues. With the fast growth of the data-driven artificial intelligence algorithms, more and more recent researches focus on its application on the current topic. Its effectiveness and powerfulness can be observed through a large number of data experiments. Nevertheless, the prediction accuracy is still a challenging task for dealing with extreme imbalance samples, particularly in big data cases. Rather than merely applying one well-defined LGB model, this study develops a novel ensemble learning strategy, i.e. a voting-based model, for imporving the prediction accuracy and the reliance. The experiment results show a progress in scores by employing this voting-based model in comparison to the single LGB model. Additionally, a new type of feature, namely the day distance to important dates, was proven to be efficient for improving overall accuracy.

Keywords: Voting-based Strategy · SMART · LGB model · Hard Drive Failure Prediction.

1 Introduction

With the fast development of modern cloud datacenters, the number of the hard disk drives deployed has grown dramatically, alongside with the absolute number of disk failures. Since these failures have unneglectable influences on the cloud service quality, the demands of the disk failure detection is increasing as well. Traditional methods mainly follow the rule-based logic by employing SMART (Self-Monitoring, Analysis and Reporting Technology) logs while recent researches show that artificial intelligence algorithm can be a competitive tool to enhance the prediction accuracy and hence gradually becomes a major solution in reality projects. Thereby, for the aim of improving application of AI

* Supported by Alibaba and Shanghai East Low Carbon Technology

algorithms, this study builds up a well-defined LGB model and subsequently attempts to develop a voting-based strategy. The experiment data source of this project is from 2020 Alibaba AI Ops Competition on Tianchi Platform, i.e. from [1].

For this specific project, comparing to the SMART data collected from other previous applications, it faces to following challenges during the modeling process:
a.Extensive data;
b.Missing records on a daily basis, probably due to hardware or network issues;
c.Difficult to capture the failure status before the failure occurs;
d.Extreme imbalance samples between the healthy and fault disks;
e.Efficient feature construction.

Multiple recent studies have attempted to address aforementioned problems. To facilitate and simplify computation and modeling, extensive data can be splitted into several segments of time series and the most relevant time segment is then chosen for the modeling. Missing records problems are widely distributed in the projects and a natural possible way is to apply filling techniques, e.g. forward and backward filling, liner and nonlinear interpolation methods, etc. Former studies have found that cubic spline interpolation ensures a "smooth" change and achieves a higher TPR (True positive rate) comparing to the spline filling and other methods [2]. Notice that, not many researches specifically emphasized the solution of an imbalanced dataset. For the extreme imbalance cases, naive upsampling and downsampling are potential source of over-fitting [3]. [3] utilized SMOTE (Synthetic Minority Oversampling Technique) for oversampling, yet the precision and recall of the model were decreased.

Except from straightforwardly utilizing given SMART attributes, new features construction greatly influences the prediction accuracy. As a time series problem, statistical sliding window features can be generated to illustrate the distribution of SMART attributes. [4] applied a gradient-based strategy to measure value transitions before disk failures, and its efficiency is validated through the data experiments. A feature combination idea was also brought up in [4], i.e. to take different fault types into account. Nevertheless, it was not clearly presented in [4] how original SMART attributes were combined with new features. Counting the number of attributes that are above zero is another potential approach to combined features [5]. Data from Backblaze shows that when there are more attributes that are above zero (in a certain group), it is more likely to indicate a disk failure.

In general, the mainly contributions of this paper can be concluded as follows: 1) strongly correlated features are extracted based on the data analysis and experiments, i.e. distance to important date, disk usage life, time series slope features and division features; 2) the correlation analysis, which relies on Pearson

and Spearman correlation coefficient, is employed for the feature selection, and latter is proved to be more effective in this specific case; 3) a voting-based strategy is developed to ensemble several LGB models with different hyperparameters to imporve the accuracy of the disk failure prediction.

2 Feature Engineering

As the basis of a robust prediction model, feature engineering has its irreplaceable place throughout every machine learning process. In this section, various feature processing methods will be illustrated, and their impact on predicting the result will be further discussed in section 4.

2.1 Data Analysis

Dataset provided by 2020 Alibaba AI Ops Competition includes daily SMART logs ranging from July 2017 to July 2018, where all disks belong to a single manufacturer "A" but with different models (model 1 and model 2). The goal of the competition was to predict failure time of disks, and results are evaluated by F1 scores in next 30 days. Thus, for the aim of simplification, the disk would fail or not in next 30 days, denoted as 0 and 1, respectively. Directly afterwards, data exploration was conducted to gain an overall impression on the data.

First of all, Fig. 1 demonstrates failed samples only occupies round 0.08% within the entire train set. It indicates the fact that the train data fed into the model are extremly imbalanced. Undersampling approach, i.e. bagging, was tested on Alibaba's dataset with some modification. Different from other imbalanced problems such as financial fraud where samples are mostly independent from each other, predicting disk failures is more difficult because each model contains continuous data points. To prevent information leak by simply doing bagging on SMART daily logs, bagging on disk serial numbers was applied.

Second, not all original features given are useful for this task. 510 original features were provided by the organizer, i.e. from smart_1 to smart_255. Each attribute possesses both a raw and a normalized value, where the former is measured by the sensors, and relied on the former, the latter is normalized by the manufacturer. For the case with a large number of attributes, attributes selection is full of challeges and arts. Consider the usefulness and completeness of the SMART attributes, we remove features that contains only Nan values and that does not change for all training and testing data samples. Therefore, the dimension of the original attributes is truncated from 510 to 45.

Third, some SMART attribute pairs are highly correlated. As mentioned, normalized attributes are generated by the corresponding raw attributes. It can be inferred that, there could be a strong linear correlation between each pair.

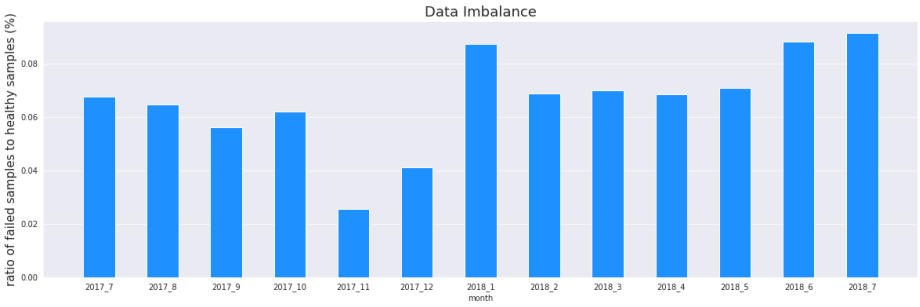

Fig. 1. Samples Imbalance Situation of the Dataset

Furthermore, some SMART attributes share similar physical meanings and could be non-linearly correlated. This influences might further reach to model training, sometimes negative to some extend, and could lead to potential over-fitting problem. Since they could provide duplicate information. From this perspective, the original feature collection should be restrained by removing strong-correlated features. During the testing, both Pearson and Spearman methods are applied to evaluate the correlation between SMART attributes and labels.

Fourth, the raw SMART attributes are skewed and needed to be properly transformed. Most machine learning methods are based on the gradient descent algorithm and as known that their performances can be significantly influenced by the given data distribution. Previous data experiments show a distribution, which is close to the Gaussian distribution, usually can provide high stability and good accuracy in comparison to non-Gaussian ditributions. To this end, the log-normal transformation was employed for these raw attributes:

$$f^{'} = log(f + 1) \tag{1}$$

in which $f^{'}$ is the feature after log-normal transformation; f is the original feature.

2.2 Feature Generation and Selection

Distance to Important Dates. During the data exploration phase, the distribution of the failed disks was pictured to investigate the trend of disk failures. It was found that more disks would fail on days when important activities were closeby. Understanding the application of the hard disks can shed some light on the feature generation. By summarizing the number of failed disks on certain days (shown in Fig. 2), following conclusions could be drawn: (a) disks were more likely to fail just 1 or 2 days before the starts of the next month (or the ends of the current month); (b) disks were more likely to fail at a few days before important holidays; (c) some peaks can be found after certain holidays. The highest peak observed occurred on Jan 23, 2018, which was about two weeks before the

Chinese Lunar New Year. These trends brought a thought that a great portion of the disks were utilized for railway ticket reservation or accounting-related application.

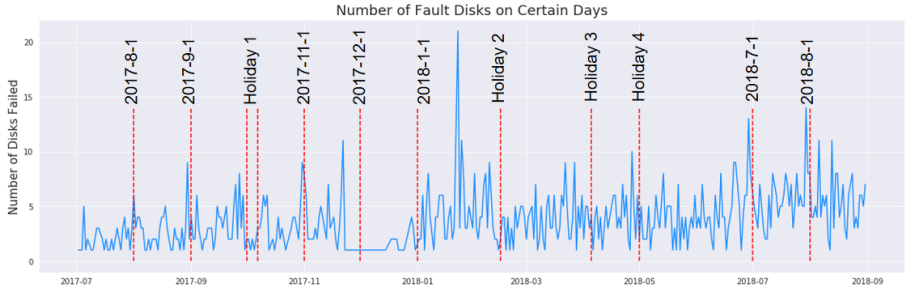

Fig. 2. Trend of failed disks

In Fig. 2, Holiday 1-4 refers to Chinese National Day, Chinese Lunar New Year's Day, QingMing, and Labor's day separately. Among those holidays, the National Day is the longest holiday (7 days). The trend around the National Day shows that peaks appeared before and after the holiday, while maintaining a low level in between. This indicates the fact that whether the day is holiday or not does not matter in this task, but the location relative to the holiday matters. Here a set of new features were proposed: days to next important date, and days to last important date. Important dates were defined based on Chinese holidays and the start of the month in this study. It is worth noting here that researchers should gain some understanding of what disks are used for, since different industries and countries have their own important dates. For instance, it is reasonable for a ticket booking system to observe increased disk failures before holidays but this is not reasonable for a manufacturer quality system due to their different business patterns.

Disk Usage Life. With the disk's usage life increases, the probability of the disk failure becomes higher. Common life span of a hard disk could be in a range of 3-5 years. Although the accumulated training data of Alibaba's disks only last around 1 year and haven't reached normal life end, the disk usage life could still be likely to provide some useful information in failure prediction.

Combined SMART feature. As discussed in the Introduction section, multiple SMART attributes reaching above zero might indicate a potential disk failure. To validate whether the same trend exists in Alibaba's dataset, the whole data set, i.e. dating from June 2018, was used for exploration. In this study, following group of SMART attributes were selected based on their physical importance: smart_5raw, smart_187raw, smart_188raw, smart_189raw, smart_197raw, and smart_198raw. The result is displayed below in Figure 6, where x-axis refers

to the number of attributes that are above zero. When none of the selected six attributes reaches above zero, 99.95% of those samples are healthy. As the number of larger-than-zero attributes grows, the possibility of failure also rises. Fig. 3 indicates a similar trend described in [5], thus can be a potential strong feature in this problem.

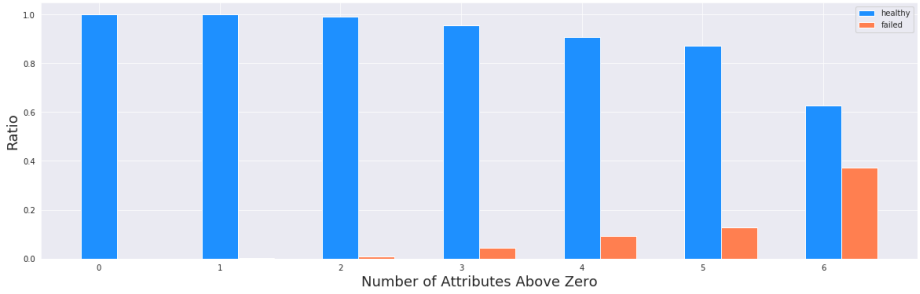

Fig. 3. Trend of sample distribution using combined SMART feature

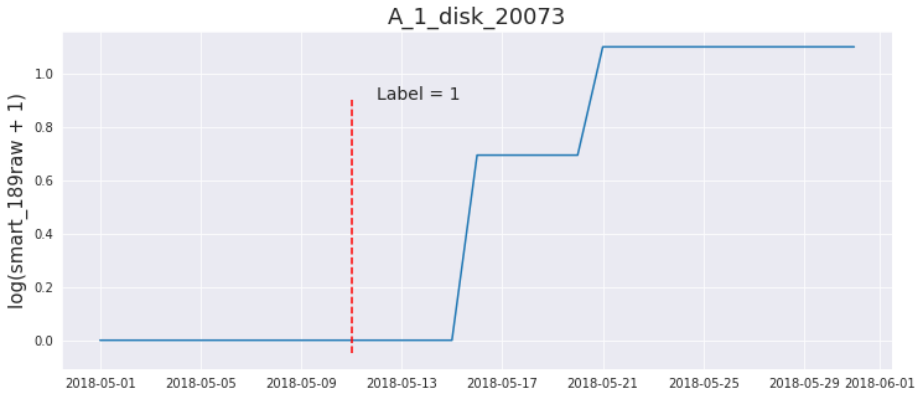

Fig. 4. Smart_189raw change before disk failed

Time Series Slope Features. According to the physical meaning of each SMART attribute [6], many attributes are not real-time but accumulated throughout the running time, such as smart_9 (represents for Power-On Hours). For those attributes, the specific value gives less information than the value changes. Besides, even for attributes that indicate real-time situation, the value changes along the time series could also indicate a status change which relates to disk health. Here one example is shown in Fig. 4: disk 20073 of model 1. The fail date for the disk is June 10, 2018, so based on the label strategy described in section 2.1, all samples after May 11, 2018 (shown as the red dashed line) is labeled

as 1. The y axis is the raw smart_189 after the log-normal transformation [see Eq.(1)]. The two steps up after May 11, 2018 could be a signal that the disk was close to its life end.

Here in this study, the labeling period was kept as 30 days because of the competition requirement. However, this selection may not apply for all disks.

Division Features. There was another Huawei competition about disk failure prediction which took place the same time as Alibaba's AI Ops Competition, and the champion solution [7] was published online. It was noticed that a new type of feature was showed a pretty significant impact. Here these features would be also taken into consideration to validate their capability in a different disk dataset. The division features are constructed as below:

$$d = \frac{f_{raw}}{f_{normalized} + 0.1} \tag{2}$$

in which d is the feature after the division transformation; f_{raw} is the raw feature; $f_{normalized}$ is the corresponding raw feature after the normalization.

It was not clear enough about the physical meaning of this type of feature, but it does provide some trending information on how linearly the raw and normalized attributes are.

3 Voting Strategy for the Probalistic Approach

3.1 Basic Model

Consider many of the features involved include Nan values and are not ideally continuous, tree-based models are more likely to provide a robust prediction than other type of models such as SVM. Therefore, Microsoft's LightGBM was selected for its high training performance and efficient memory usage [8]. Because of the extreme imbalanced rate, it was seen that high learning rate and more iterations could potentially lead to over-fitting to the specific training data used. Thus, after a few experiments the parameters were kept with low learning rate and less iterations.

Several down-sampling approach were tested but none of them showed better result. As a result, LightGBM's embedded setting was used to mitigate the imbalance issue. In LightGBM, the parameter "is_unbalanced" provides an approach to deal the imbalance problem with an adjusted loss function [9]:

$$L(y) = -\frac{I_+}{n_+} logP_+ - \frac{I_-}{n_-} logP_- \tag{3}$$

in which denote the modeled conditional probabilisties by $P_+ := P(y = 1|x)$ and $P_- := P(y = 0|x)$, and define indicators $I_+(x_i) = 1$ if $y_i = 1$, and $I_-(x_i) = 1$ if

$y_i = 0$, vice versa. n_+ and n_- are the number of postive and negative samples, respectively.

Even though the loss function was adjusted for treating imbalanced problem, it was still a logistic approach, and traditional logistic binary classification chooses 0.5 as the threshold to separate negative and positive samples. Nevertheless, with extreme imbalanced dataset used in this task, sticking to 0.5 would categorize almost all samples as healthy. Therefore, a self-defined threshold was put on the calculated probability to better filtering the most-likely failed samples.

3.2 Voting Strategy Framework

Although single LGB model could provide accurate prediction for a group of disk failures, it can also provide completely different result when the combination of training parameters were slighted changed or a different month of data was used for training. There is a possibility that not all LGB models are equally good at predicting all samples, and thus some measures are needed to combine multiple models.

Traditional model ensemble approaches are mostly depended on model blending or stacking, but when investigating the predicted probability for disk samples, totally different probability distribution can be seen for a same set of validation data. This phenomenon implied that predicted probability for failure could be so different that an outlier result can dominate the final result in simple blending or stacking operation. Therefore, we propose a more robust voting strategy to minimize the influences of calculated probability and reliance on a single threshold.

4 Case Study

In order to quantify the effect of features proposed in section 2 and the voting strategy in section 3, multiple cases were tested on Alibaba's docker platform. The models were tested without knowing the specific test data, providing a fairly close-to-real-world environment. Furthermore, we limited the number of disks submitted to 140-160, so that results can be compared without the effect of submissions.

4.1 Attributes Filtering

In section 2.1, it was found that many SMART attributes are highly correlated, therefore, filtering out those high-correlated attributes might bring potential benefit to the prediction. Table 2 lists the related tests and the corresponding

Table 1. Pseudocode for the voting-based robust model.

Algorithm: The voting-based strategy via a series of LGB models
Step 1: Build up a well-defined LGB model for the binary classification.
i) set the objective function as "binary";
ii) define the metric function as "binary_logloss".
Step 2: Select a series of hyperparameters for LGB models.
i) choose a series of values for hyperparameters, i.e. "learning_rate", "n_estimators" and "subsample";
ii) define combinations of those parameters;
Step 3: Make probability predictions.
For each model, obtain the disks failure probability, i.e.
$Pr_i[y = 1\|x], i = 1, 2, ..., m$
in which m is number of models;
Step 4: Define the appropriate threshold.
i) strategy 1: constant threshold θ, e.g. θ =0.005, 0.006, etc.;
ii) strategy 2: adaptive threshold θ, e.g. $\theta = pencentile(Pr_i[y = 1\|x], 10)$.;
Step 5: Output disk failure time via a voting-based strategy.
if $Pr_i[y = 1\|x] >= \theta$: count += 1
if count >m/2: failure disk;
else: healthy disk.

results.

Before additional features were introduced, high-correlated attributes filtered by

Table 2. Result Comparison for Attribute Selection

No.	Method Applied	Online Score
1a	Single LGB model with non-Nan features	20.57%
1b	Single LGB model using Pearson correlation to filter attributes	19.65%
1c	Single LGB model using Spearman correlation to filter attributes	22.21%
2	Compared to 1a, add distance to important dates feature	21.79%
3	Compared to 1b, add distance to important dates feature	21.88%
11	Compared to 1b, add all generated features and apply voting strategy	22.48%
13	Compared to 1c, add all generated features and apply voting strategy	21.53%

Pearson correlation slightly reduced the online scores (see 1a and 1b). However, both 1a and 1b indicated the low prediction accuracy and were needed to be improved. This circumstance did not change until the distance to important dates were defined and added as features (compare 1a and 2, 1b and 3). Notice that, with filtering out high-correlated attributes by Pearson correlation presented a little better result than with original attributes (see 2 and 3). This unexpected phenonmenon reveals that the features addition might be nonlinear. Compare to Pearson correlation, Spearman correlation is more suitable for evaluating the statistical dependence between the rankings of two variables and has wider applications. Hence, Spearman correlation test 1c was carried out and a significant

improvement can be seen (compare 1b and 1c). Neverthless, the benefit of the Spearman correlation filtering is not always improving the prediction accuracy (see 11 and 13), it can boost the model under certain conditions but not always the case.

4.2 Features addition

In section 2.2, five types of features were proposed. To better understand the effect of each feature, 7 tests were conducted step by step in Fig. 5.

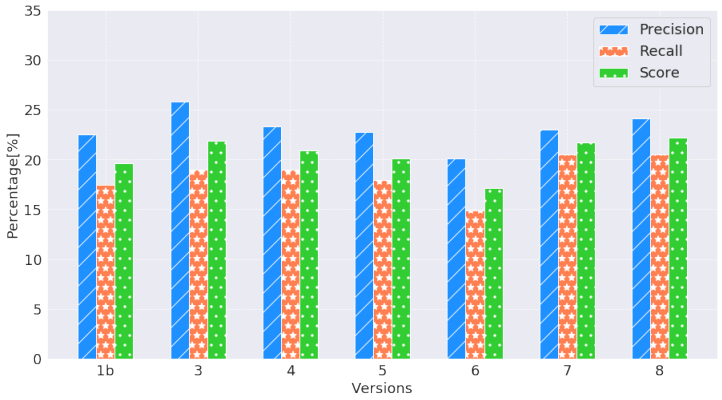

Fig. 5. Result Comparison for Generated Features

No.	Method Applied	Online Score
1b	Single LGB model with only select low-correlated attributes	19.65%
3	Compared to 1b, add distance to important dates feature	21.88%
4	Compared to 3, add hard disk usage life feature	20.94%
5	Compared to 4, add combined SMART feature	20.08%
6	Compared to 5, add time series slope features	17.10%
7	Compared to 5, add corresponding division features	21.68%
8a	Compared to 5, add time series slope features and corresponding division features	22.16%

The number of features involved were added one by one from No.1b to No.8. From the result, it was seen that following constructed features did show a significant contribution: distance to important dates, division features. The influences caused by time series slope features were not clear enough because it lowered the score when comparing No.5 and No.6 but improved the score when comparing No.7 and No.8.

4.3 Sample Imbalance

As explained in section 2.1, the hard disk dataset is extremely imbalanced. Traditional approach for dealing with imbalance problem includes down-sampling and up-sampling, however, with time-series data as the SMART log data, simple up-sampling and down-sampling approach would result in information leak. Therefore, the approach here was to do bagging or duplication on disk serial numbers instead of simply up-sampling or down-sampling daily records. Result Comparison can be found in Table 3.

Table 3. Result Comparison for Data Up-sampling and Down-sampling

No.	Method Applied	Online Score
8a	Single LGB model with all generated features, using May-June 2018 data	22.16%
8b	Compared to 8, down-sampling negative samples only	17.76%
8c	Compared to 8, up-sampling positive samples and down-sampling negative samples	16.98%

Due to the memory limits, a full up-sampling test was not able to be processed. Hence a middle approach was utilized, where down-sampling and up-sampling were combined. It was observed that although these processing method decreased the imbalance extent of the dataset, the model was easily gaining high precision and recall score on the validation set even with low learning rate and less iterations.

4.4 Training Data

Since the imbalance issue unpreventably brought over-fitting, the selection of the training data becomes essential. Three tests were conducted where only training dataset were varied. The results verifies the hypothesis that training data affects the prediction result to a great extent. This phenomenon raises a concern that there may not exist a set of perfect training data that generates the best result for all testing conditions, meaning it would be hard to know which data should be used for training. However, it was still reasonable to utilize more recent data than data long ago.

4.5 Voting Strategy

As described in section 3, single LGB model does not provide robust results, thus using voting strategy does not only improve the accuracy, but also reduce model's reliance on the probabilistic threshold. In Table 5, it is seen that voting strategy is able to increase the prediction accuracy to some degree, but a bad set of sub-models can also greatly harm the result.

Table 4. Result Comparison for Different Training Data

No.	Method Applied	Online Score
8a	Single LGB model with all generated features, using May-June 2018 data	22.16%
9	Compared to 8, use June-July 2018 as training data	14.53%
10	Compared to 8, use July 2018 as training data	17.15%

Table 5. Result Comparison for Voting Strategy

No.	Method Applied	Online Score
8a	Single LGB model with all generated features	22.16%
11	Compared to 8, use voting strategy instead of a single LGB model: train 7 sub-models with May-June 2018 data	22.48%
12	Compared to 11, add 7 more sub-models trained by June-July 2018 data	17.59%

5 Conclusion

This study investigated various feature construction and filtering methods and proposed a new type of feature, i.e. day distance to important dates. Moreover, a developed voting-based strategy algorithm was applied rather than one well-defined LGB model. During the competition, it was seen that day distance features contributed significant improvement and the voting-based strategy ensured a better result. Noted that, although the tests were designed to gradually add optimized sub-approaches, the benefit of these sub-approaches was not seen to be added linearly. It would be of great worth to investigate the impact of feature combination and interaction. Additionally, this study is worthwhile for further analysis on voting-based approach and can be expanded to include other models, e.g. XGBoost, CatBoost etc.

6 Acknowledgements

This study and experiment sources are strongly support by the Shanghai East Low Carbon Technology Industry CO., LTD. and Beijing Megvii Co., Ltd. Thanks to Alibaba, PAKDD for hosting and supporting this competition.

References

1. https://github.com/alibaba-edu/dcbrain/tree/master/diskdata
2. Shujie Han, etc. Robust Data Preprocessing for Machine-Learning-Based Disk Failure Prediction in Cloud Production Environments.
3. Nicolas Aussel, Samuel Jaulin, Guillaume Gandon, Yohan Petetin, Eriza Fazli, et al.. Predictive models of hard drive failures based on operational data. ICMLA 2017 : 16th IEEE International Conference On Machine Learning And Applications, Dec 2017, Cancun, Mexico. pp.619 - 625, ff10.1109/ICMLA.2017.00-92ff. ffhal-01703140

4. Wenjun Yang, etc. Hard Drive Failure Prediction Using Big Data, 2015 IEEE 34th Symposium on Reliable Distributed Systems Workshops
5. https://www.backblaze.com/blog/what-smart-stats-indicate-hard-drive-failures/
6. https://en.wikipedia.org/wiki/S.M.A.R.T.
7. https://mp.weixin.qq.com/s/LEsJvrB4V3YyOAZP-PGLFA
8. LightGBM Repository. https://github.com/microsoft/LightGBM
9. Burges, Christopher JC. "From ranknet to lambdarank to lambdamart: An overview." Learning 11.23-581 (2010): 81.