Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

The Comparison of LightGBM and XGBoost Coupling Factor Analysis and Prediagnosis of Acute Liver Failure

Dongyang Zhang*[1] **and Yicheng Gong*** [1,2]

[1]Department of Mathematics and Statistics, Science College, Wuhan University of Science and Technology, Wuhan 430065, China

[2]Hubei Province Key Laboratory of Systems Science in Metallurgical Process, Wuhan University of Science and Technology, Wuhan 430065, China

Corresponding author: Dongyang Zhang (e-mail: 1044600738@qq.com), Yicheng Gong (e-mail: gongyicheng@wust.edu.cn)

This work was supported in part by Hubei Provincial College Students Innovation and Entrepreneurship Training Program (201810488097), Hubei Provincial Key Laboratory of Metallurgical Industry Process (Y201906) and the National Natural Science Foundation of China(Grant No. 72031009,71871171). *Asterisk indicates corresponding authors*.

ABSTRACT This paper focuses on the comparison of dimensionality reduction effect between LightGBM and XGBoost-FA. With respect to XGBoost, LightGBM can be built in the effect of dimensionality reduction via both Gradient-based One-Side Sampling(GOSS) and Exclusive Feature Bundling(EFB) algorithms, while XGBoost coupling with traditional dimensionality reduction tool Factor Analysis (XGBoost-FA) may also have dimensionality reduction effect. To present the empirical comparison, the prediagnosis dataset for the 2018 Kaggle competition Acute Liver Failure has been chosen as the research object. And pairwise comparison has been conducted among XGBoost, LightGBM, XGBoost-FA and LightGBM-FA. Concerning the test set, the vector (accuracy, log loss function, training time) of the above first four prediagnostic models are (0.75014, 0.569707, 10.5s), (0.75811, 0.576059,15.1s), (0.67786,0.663924,5.7s) and (0.67274,0.676019, 4.1s) respectively. It’s been found that the training time of XGBoost-FA (external dimensionality reduction) is shorter than that of LightGBM (build-in dimensionality reduction). Considering (accuracy, training time) being (0.82, 3.1s) published on Kaggle, the algorithm (logogram as K2a) is better than the four XGBoost-FA and LightGBM in both training time and accuracy. However, K2a removes more than 50% samples with missing values and only performs binary classification. For multi-class classification or data with a large number of missing values, XGBoost-FA is more suggested if higher operational time is required, while LightGBM is preferred if higher predictive accuracy is required. With XGBoost-FA or LightGBM being employed in AI medical services, doctors are more productive in diagnosis and treatment due to much more data support and less workload. Both complement each other.

INDEX TERMS ***LightGBM, XGBoost, Factor Analysis, Prediagnosis, Acute Liver Failure***

I. INTRODUCTION

Based on Gradient Boosting Decision Tree (GBDT), XGBoost and LightGBM are both popular and cutting-edge boosting integrated algorithms in machine learning in recent years. By virtue of less training time and higher accuracy, many scholars have applied XGBoost and LightGBM in various scenarios.

In the information technology field, Jin built real-time intrusion detection system based on LightGBM and parallel intrusion detection mechanism [1]. Concerning people's livelihood, Xie *et al.* used the historical data and three machine learning models (GBDT, XGBoost and LightGBM) to predict the monthly housing rent. The prediction results showed that XGBoost and LightGBM are superior to the traditional GBDT [2]. Parsa *et al.* applied XGBoost to establish a prediction model for highways, which can detect real-time accident and analyze features [3]. In the remote sensing image field, Samat investigated the performance of XGBoost in remote sensing image classification tasks [4]. In the medical field, Zhang *et al.* compared XGBoost with Artificial Neural Networks (ANN) and Random Forest (RF) in log loss function and training time based on dataset about Acute Liver Failure. The results indicate that XGBoost is the most advantageous in classification accuracy [5]. To determine whether a sample has Alzheimer's disease by the samples of brains pictures, Zhou *et al.* found that LightGBM has more advantages over Support Vector Machine (SVM)

[6]. The above application scenarios show that XGBoost and LightGBM perform better, especially in medical field.

Many scholars adopted XGBoost while analyzing and modeling medical data. Shen *et al.* optimized the diagnostic technology of breast cancer [7]. Leng *et al.* classified gene based on XGBoost which performs better than Logistic Regression Classifier (LRC) and SVM [8]. Yang *et al.* used XGBoost to build a user network score prediction model to mine behavior features, whose accuracy and efficiency are higher than those of LRC and RF [9]. Guo *et al.* established a physical health assessment model based on XGBoost and used adolescent physiological data to assess youths' physical health [10]. While some other scholars used LightGBM to deal with medical data. Zhang *et al.* constructed an acute kidney injury prediction model for ICU patients, and provided auxiliary decision support for clinical medical staff, which showed that LightGBM performs better than LRC and RF [11]. In the above medical scenarios, both XGBoost and LightGBM perform well.

In light of the explosive growth of medical data and their feature dimensions, many scholars pay further attentions to data dimensionality reduction in order to shorten the training time of machine learning. Thus, dimensionality reduction method is introduced into XGBoost and LightGBM in this paper.

LightGBM can remedy the defects of XGBoost via both the Gradient-based One-Side Sampling (GOSS) and the Exclusive Feature Bundling (EFB) algorithm. GOSS and EFB play the built-in role of dimensionality reduction and efficiency improvement in LightGBM. Additionally, XGBoost coupling external dimensionality reduction tool can realize dimensionality reduction as well. To explore the specific difference between the external dimensionality reduction effect and the built-in dimensionality reduction effect, it is actually worth comprising.

In general, dimensionality reduction methods include Chi2, MI, LDA, PCA, Factor Analysis(FA) and autoencoder dimensionality reduction, and so on. While processing the sound signal data, Ali *et al.* used linear discriminant analysis (LDA) and Chi2 statistical model (Chi2) rank to detect automatic Parkinson's disease (PD) [12-14]. Based on Particle Swarm Optimization (PSO) for feature selection, Souri*et al.* established a fault prediction model in Internet of Things Applications [15]. In terms of image-based medical data processing, Kong *et al.* used Common Space Pattern (CSP) to select a low-dimensional and efficient feature set and applied Convolutional Neural Network (CNN) classifier to identify brain waves [16]. To effectively classify and recognize the ultrasonic signals, Huang *et al.* adopted Principal Component Analysis (PCA) to simplify features and realize the coupling with SVM [17]. Aiming at the problem of further improving the recognition rate of facial expression, Zhang *et al.* proposed a deep learning-based Stacked Hybrid Auto-Encoder (SHAE) method to realize dimensionality reduction [18]. While processing the medical data, Ali *et al.* proposed a two-stage decision support system, where mutual information (MI) reduced the data dimension and then neural network predicted the heart failure (HF) [19]. To explore the impact of interaction between two genes upon schizophrenia, Zhang *et al.* used multi-factor dimensionality reduction (MDR) to analyze gene-gene interactions [20]. Concerning non-linear high-dimensional medical data dimensionality reduction, Weng *et al.* introduced an isometric feature mapping (Isomap) method and applied it to detect lung cancer and breast cancer [21]. Gong et al. used PCA for dimensionality reduction and predicted blood glucose levels based on GBDT [22]. The above empirical analysis shows that machine learning method incorporating dimensionality reduction has achieved better prediction results, and the model performance has also been improved to a certain extent.

Some scholars have proposed state-of-the-art dimensionality reduction methods and managed to reduce feature redundancy. M. Eftekhari and F. Saberi-Movahed remove duplicate features according to the theory of information gain and matrix factorization respectively [23,24]. Wang *et al.* proposed two feature selection methods, two-steps attribute reduction based on fuzzy dependency (TARFD) algorithm and fuzzy rough artificial bee colony (FRABC) algorithm, and applied them to radar signal processing [25]. Li *et al.* proposed a regular regression model with a generalized uncorrelated constraint for feature selection [26]. Wang *et al.* devised a multi-network feature extraction model using pre-trained deep convolution neural networks (DCNNs) and used it to detect breast cancer [27]. These new dimensionality reduction methods may play an important role in medical scenarios in the near future.

Considering the actual meaning behind the medical data is of great importance in the relevant analysis, it's advised to perform data actual interpretation after dimensionality reduction. Among the above dimensionality reduction methods, FA enables the new common factors to carry the message of the realistic background without changing the original variables. That's the reason why FA is chosen for dimensionality reduction.

To explore the impact of built-in dimensionality reduction and external dimensionality reduction, corresponding comparison has been made among XGBoost, LightGBM, XGBoost-FA and LightGBM-FA in this paper. Specifically, Acute Liver Failure dataset of the Kaggle competition is used to present the empirical comparison.

II. INTRODUCTION TO THE PRINCIPLES OF FACTOR ANALYSIS (FA), XGBOOST AND LIGHTGBM

A. The basic principles of Factor Analysis (FA)

Factor Analysis is a multidimensional statistical method that converts multiple measured variables into a few unrelated comprehensive indicators. So, FA can reduce dimensionality and simplify data [28].

Assume that the number of related random variables are p, which contains m independent factors. FA can be expressed as formula (1).

$$\begin{cases} X_1 = a_{11}F_1 + a_{12}F_2 + \cdots + a_{1m}F_m + \varepsilon_1 \\ X_2 = a_{21}F_1 + a_{22}F_2 + \cdots + a_{2m}F_m + \varepsilon_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ X_p = a_{p1}F_1 + a_{p2}F_2 + \cdots + a_{pm}F_m + \varepsilon_p \end{cases} \cdots\cdots \quad (1)$$

In the formula (1), F_1, F_2, $\cdots$, F_m are known as common factors, which are unobtainable variables. Their coefficients a_{11}, a_{12}, $\cdots$, a_{pm} are referred to as the loading factors. ε_1, $\cdots$, ε_p are special factors and cannot be included in public factors. The flow of FA is shown in the following 7 steps.

Step 1. Standardize the raw data to eliminate the differences between magnitude and dimension of variables. As shown in formula (2).

$$x_i = \frac{x_i - E(x_i)}{\sqrt{Var(x_i)}} \cdots\cdots\cdots\cdots\cdots\cdots\cdots \ (2)$$

Step 2. Calculate the correlation matrix of the standardized data. The value of the correlation coefficient r is between -1 and 1. And when |r| trends to 1, the linear correlation between the variables grows strong.

Step 3. Calculate the variance contribution and the cumulative variance contribution of the public factor, as shown in formula (3) and formula (4).

$$\frac{\lambda_i}{\sum_{k=1}^{p} \lambda_k} (i = 1,2,\cdots,p) \cdots\cdots\cdots\cdots\cdots\cdots\cdots \ (3)$$

$$\frac{\sum_{k=1}^{i} \lambda_k}{\sum_{k=1}^{p} \lambda_k} (i = 1,2,\cdots,p) \cdots\cdots\cdots\cdots\cdots\cdots\cdots \ (4)$$

Where λ_i is an eigenvalue of the correlation matrix R and the corresponding eigenvector l_i of the standard orthogonality.

Step 4. Select the common factors. Set F_1, F_2, $\cdots$, F_p as the p factors, which contains the total amount of data information (its cumulative contribution rate is 100%). If the accumulated data information of the first m factors reaches up to or is over 80%, they shall be taken to represent the original evaluation index.

Step 5. Factors rotation. If the first m factors cannot be determined or its actual meaning is unclear, the factors should be rotated to obtain a clearer practical meaning. The specific transformation is shown in formula (5).

$$a_{ij} = \sqrt{\lambda_i} l_{ij} (i, j = 1,2,\cdots,p)$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{1m} \\ \vdots & \vdots & \cdots & \vdots \\ a_{p1} & a_{p2} & \cdots & a_{pm} \end{bmatrix}$$

$$= \begin{bmatrix} \sqrt{\lambda_1} l_{11} & \sqrt{\lambda_2} l_{12} & \cdots & \sqrt{\lambda_m} l_{1m} \\ \sqrt{\lambda_1} l_{21} & \sqrt{\lambda_2} l_{22} & \cdots & \sqrt{\lambda_m} l_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ \sqrt{\lambda_1} l_{p1} & \sqrt{\lambda_2} l_{p2} & \cdots & \sqrt{\lambda_m} l_{pm} \end{bmatrix}$$

$$= (\sqrt{\lambda_1} l_1, \sqrt{\lambda_2} l_2, \cdots, \sqrt{\lambda_m} l_m) \cdots\cdots\cdots\cdots \ (5)$$

Step 6. Use the linear combination of the original variables to obtain the score for each factor. Use regression estimation or Bartlett estimation to calculate the factor score.

Step 7. Comprehensive score. The comprehensive evaluation index function is the linear combination of each factor, as shown in formula (6).

$$F = \frac{\gamma_1 F_1 + \gamma_2 F_2 + \cdots + \gamma_m F_m}{\gamma_1 + \gamma_2 + \cdots + \gamma_m} = \sum_{i=1}^{m} \omega_i F_i \cdots\cdots\cdots \ (6)$$

In formula (6), ω_i is the weight of F_i and ω_i is the variance contribution rate of the pre-rotation or post-rotation factors.

Step 8. Score ranking. Use the composite score analysis to obtain a score ranking.

B. *The basic principle of XGBoost*

The full name of XGBoost is eXtreme Gradient Boosting, proposed by Dr. Tianqi Chen who worked in the University of Washington in 2014. XGBoost is a tree integration model, which uses the cumulative sum of the predicted values of a sample in each tree as the prediction of the sample in the XGBoost system [29]. XGBoost reduces the risk of overfitting by adding regular terms and directly uses the first derivative and second derivative value of the loss function. The specific algorithm flow is performed in the following 8 steps, as shown in Figure 1.

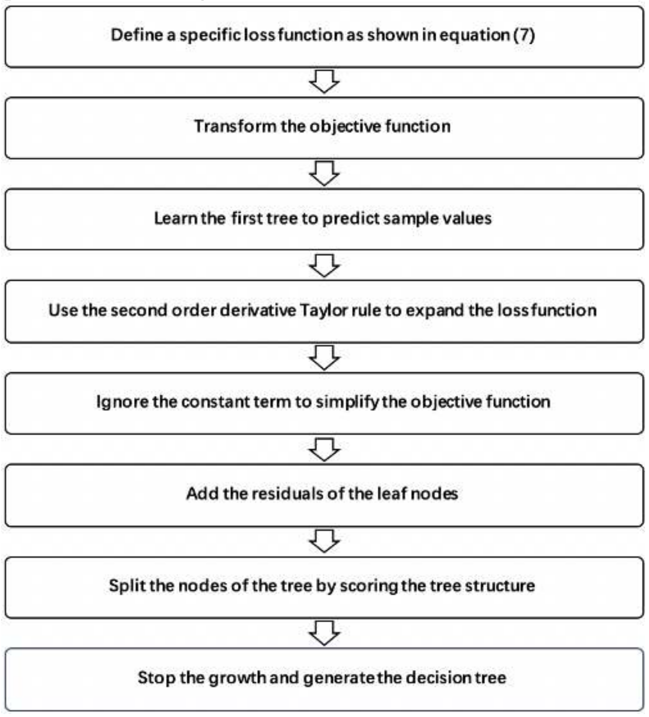
Define a specific loss function as shown in equation (7)
Transform the objective function
Learn the first tree to predict sample values
Use the second order derivative Taylor rule to expand the loss function
Ignore the constant term to simplify the objective function
Add the residuals of the leaf nodes
Split the nodes of the tree by scoring the tree structure
Stop the growth and generate the decision tree

FIGURE 1. The basic principle of XGBoost

Step 1. Define a specific loss function. The objective function is divided into two major terms as shown in formula (7).

$$L(\emptyset) = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \cdots\cdots\cdots\cdots \ (7)$$

In formula (7), $\hat{y}_i$ is the predictive output, y_i is the label value (true value), f_k is the kth tree model, T is the number of leaf nodes in the kth tree, γ is the leaf tree penalty regular term. $\sum_i l(y_i, \hat{y}_i)$ is the loss sum per sample, and XGBoost's loss function l can be customized variously. $\Omega(f_k) = \gamma T + \frac{1}{2}\lambda\|w\|^2$ is the regular term, where w is the kth tree's leaf node weight value and λ is the leaf weight regular penalty term, which act as a smoothing factor to calculate gain in the process of splitting points. And both parts of the penalty terms can prevent overfitting.

Step 2. Transform the objective function based on the boosting algorithm and decision tree algorithm. In the boosting algorithm, the final predictive value is the sum of the outputs of multiple trees, as shown in formula (8).

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \cdots\cdots (8)$$

In formula (8), $\mathcal{L}^{(t)}$ is the objective function of the t tree. $\hat{y}_i^{(t-1)}$ is the sum of the output values of the $\mathrm{t}-1$ trees, which constitutes the predictive value of the first $\mathrm{t}-1$ tree $f_t(x_i)$ is the output of the t th tree. The sum of the two parts constitutes the latest predictive value. The regular term becomes the corresponding term of the current tree.

Step 3. Learn the first tree to predict sample values.

Step 4. Use the second order derivative Taylor rule to expand the loss function $l(y_i, \hat{y}_i^{(t-1)})$ to learn the t th tree as formula (9).

$$\mathcal{L}^{(t)} \sim \sum_{i=1}^{n}[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + \Omega(f_t) \cdots\cdots (9)$$

In formula (9), $g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ is the first order partial derivative of the previous tree's loss function. $h_i = \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$ is the second order partial derivative of the previous tree's loss function.

Step 5. Ignore the constant term to simplify the objective function. The first part of the function $\sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)})$ is the loss function from the previous round of tree building. It is a known constant in the current round of tree building, and the value does not affect our tree building process in this round. We can ignore it and thus simplify the objective function as shown in formula (10).

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^{n}[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + \Omega(f_t) \cdots\cdots\cdots\cdots (10)$$

Step 6. Add the residuals of the leaf nodes. Firstly, suppose the tree has T leaf nodes, and the set of samples in the jth leaf node is represented as $I_j = \{i|q(x_i) = j\}$. Based upon this, $\sum_{i=1}^{n} g_i f_t(x_i) = \sum_{j=1}^{T}(\sum_{i\in I_j} g_i)w_j$. $\sum_{i\in I_j} g_i$, which is the sum of the first derivatives corresponding to all samples that fall into leaf node j. Accordingly, $\sum_{i=1}^{n}\frac{1}{2}h_i f_t^2(x_i)$ can also be converted to $\sum_{i=1}^{n}\frac{1}{2}\sum_{i\in I_j} h_j w_j^2$. Thus, after the conversion and the merging of the regular terms, the formula (10) is converted to formula (11).

$$\tilde{\mathcal{L}}^{(t)} = \sum_{i=1}^{n}\left[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)\right] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2$$

$$= \sum_{j=1}^{T}\left[(\sum_{i\in I_j} g_i)w_j + \frac{1}{2}(\sum_{i\in I_j} h_i + \lambda)w_j^2\right] + \gamma T \cdots\cdots (11)$$

Step 7. Split the nodes of the tree by scoring the tree structure. The formula is shown in formula (12).

$$\mathcal{L}_{split} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i + \lambda}\right] - \gamma \cdots (12)$$

To put it in simpler terms, we can split the tree when the value of the loss function reduction is larger than γ, which is the gamma parameter added to the denominator while calculating the gain to smooth it out. γ can help to prevent overfitting, which becomes optional rather than mandatory.

Step 8. Stop the growth and generate the decision tree.

C. The basic principle of LightGBM

LightGBM is a lightweight gradient booster, launched by Microsoft in 2017. To solve time-consuming problem in the environment of large high-dimensional data sample, LightGBM was proposed [30]. The specific algorithm flow is performed in the following 8 steps, as shown in Figure 2.

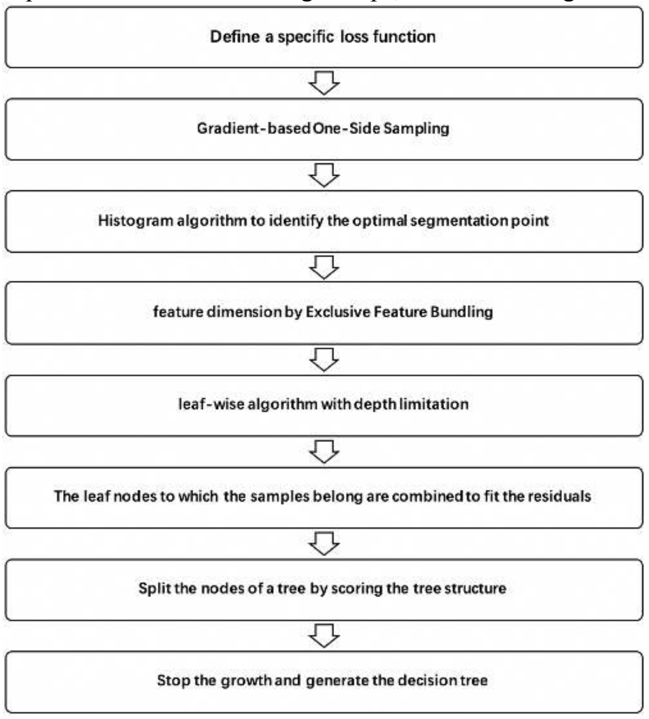
Define a specific loss function
Gradient-based One-Side Sampling
Histogram algorithm to identify the optimal segmentation point
feature dimension by Exclusive Feature Bundling
leaf-wise algorithm with depth limitation
The leaf nodes to which the samples belong are combined to fit the residuals
Split the nodes of a tree by scoring the tree structure
Stop the growth and generate the decision tree

FIGURE 2. The basic principle of LightGBM

LightGBM is an improved version of XGBoost with four aspects improved.

Firstly, LightGBM's algorithm incorporates GOSS (Gradient-based One-Side Sampling) algorithm. GOSS strikes a good balance between the number of samples and precision for the decision tree in LightGBM. In training, more attention is paid to those samples with larger gradients, which also have more influence on the gain.

Secondly, LightGBM uses a histogram to identify the optimal segmentation point. Hhistogram algorithm firstly discretizes successive floating-point eigenvalues into k integers, and simultaneously constructs a histogram with width *k*. Based on the discretized values, the accumulated statistics in the histogram are selected as an index after traversing the data. Then, the computation of segmentation scores is identified. Histogram replaces the traditional Pre-Sorted, so in a sense it sacrifices accuracy for speed.

Thirdly, LightGBM reduces the feature dimension to a certain extent by means of Exclusive Feature Bundling (EFB). For sparse feature space (for example, onehot encoding), it is possible to reduce the number of valid features by binding mutually exclusive features together to form a new feature.

Fourthly, LightGBM uses a leaf-wise algorithm with depth limitation instead of the traditional level-wise, which improves accuracy and prevents overfitting.

III. THE ALGORITHM DIFFERENCES OF XGBOOST AND LIGHTGBM

The algorithmic differences between LightGBM and XGBoost are mainly fall into following three aspects.

1. There are two ways in LightGBM which play a role in dimensionality reduction compared to XGBoost.

1) Exclusive Feature Bundling (EFB). Based on graph algorithm, EFB incorporates some features to reduce the total number of features.

2) Gradient-based One-Side Sampling (GOSS). Firstly, GOSS ranks the samples according to their gradient, selects the samples with a% of large gradient. Then it randomly selects the other samples with b% of small gradient, and combines them to evaluate the information gain, which reduces the number of samples and indirectly reduces the probability of small gradient samples.

2. LightGBM uses the leaf-wise strategy, while XGBoost uses the level-wise strategy. XGBoost splits indiscriminately all nodes in each layer and splits some nodes with small gain, which barely affect the results, leading to resources wasting. In LightGBM, the leaf nodes with the greatest splitting gain at the current are selected, which will lead to overfitting easily and the tree growing much larger than expected. Therefore, the depth of the tree must be set in LightGBM.

3. LightGBM's parallelism policy includes feature parallelism, data parallelism, and voting parallelism, while XGBoost's parallelism policy mainly focuses on feature. Therefore, LightGBM is faster than XGBoost for larger amounts of data.

IV. THE ANALYSIS IDEAS AND FRAMEWORK OF THIS PAPER

Based on the principles of FA, XGBoost and LightGBM (Section II), this paper specifically brings forward the following four steps.

Step 1. To facilitate machine learning later, data preprocessing including variable normalization, quantification of categorical variables, filling of missing values, and quantification of label variables.

Step 2. FA is used to reduce the dimensionality of the preprocessed data, and analyze the dimensionality reduction results and then explain the actual background of the obtained common factors.

Step 3. Based on the preprocessed data, XGBoost and LightGBM are trained. Based on the data after dimensionality reduction, XGBoost-FA and LightGBM-FA are trained.

Step 4. Comparative analyses of the four models of XGBoost, XGBoost-FA, LightGBM and LightGBM-FA, and the algorithm released by Kaggle for the prediagnosis results of acute liver failure.

For all algorithms in this paper, the operating system is macOS version 10.14.6 and the processor is 1.4 GHz Intel Core i5, and the installed memory (RAM) is 8GB. Anaconda3 of Jupyter Notebook and IBM SPSS Statistics 22 are used to run the algorithm program stand-alone.

V. INTRODUCTION AND PREPROCESSING OF ACUTE LIVER FAILURE DATA

From Kaggle's public datasets, this paper chooses one which contains 8785 samples of 8785 adult patients collected by JPAC Health Diagnostics and Control Center in 2014-2015. For each sample, there are 30 relevant variables. To facilitate subsequent machine learning, the raw data shall be preprocessed as the following four steps.

Step 1. Variable renaming. To comply with the software's usage specifications, the variables are renamed with separators in English.

Step 2. Quantification of categorical variables. To facilitate the processing of categorical variables, different categories are replaced with different values (Such as the 'Gender' variable is divided into 'male' and 'female', and it is replaced with '0' and '1'). No sequential relations are involved.

Step 3. Missing data filling. Some of the variables in the raw data contain missing data. For example, 'Waist' is a numerical variable which has 8471 non-empty raw data, whose mean is filled in the rest 314 empty data in this paper. 'HyperTension' is a categorical variable which also has 8471 non-empty raw data, whose mode is filled in the rest 314 empty data in this paper.

Step 4. Identification and quantification of the label variable. According to the research topic of this paper, the variable 'ALF' is used as the dependent variable (label variable), and the rest of variables are used as independent variables (feature variables). In detail, 'diagnosed without acute liver failure' is recorded as '0', 'diagnosed with acute liver failure ' is recorded as '1' and 'acute liver failure has not been diagnosed' is recorded as '2'.

VI. DIMENSIONALITY REDUCTION RESULTS OF ACUTE LIVER FAILURE DATA BASED ON FACTOR ANALYSIS

In dataset of Acute Liver Failure, there are as many as 29 feature variables (independent variables). Factor Analysis (FA, Section II.A) is employed to decrease the number of variable while retaining the common factors with practical meaning. SPSS is used to realize FA. The results are shown in Table I.

TABLE I

VARIANCE EXPLANATION TABLE

Variable	Starting Eigenvalue		
	Total	Variance%	Cumulative%
1	3.399	11.330	11.330
2	3.145	10.482	21.812
…	…	…	…
16	.810	2.701	80.613
…	…	…	…
29	.029	.098	100.000

The fourth column of Table I is the cumulative percentage of initial feature values. The value in the row k, column 4 stands for the percentage of the information contained in the first k common factors of the original total information.

According to the formula (1) given in Section II.A, the benchmark is that the common factors accounts for 80% of the original total information. From the 15th, 16th and 17th rows of the 4th column, it shows that the cumulative contribution exceeds 80% for the first time and reaches 80.613% when m=16. Therefore, in dataset of Acute Liver Failure, m=16 is selected in formula (1). According to formula (1), each original variable can be expressed as a linear combination of 16 common factors.

In order to give the common factors explanation that conforms to the actual background, the common factors in Table I are orthogonally rotated to obtain the final common factors FACT1, ⋯, FACT16. Thus, each of 29 original variables can be expressed as a linear combination of FACT1, ⋯, FACT16. The loading factor matrix is shown in Table II.

TABLE II
LOADING FACTOR MATRIX AFTER ROTATION

Variable	common factors			
	FACT1	FACT2	…	FACT16
Weight	0.891	0.076	⋯	-0.007
Waist	0.888	0.058	⋯	0.036
Obesity	0.875	-0.055	⋯	0.002
Education	-0.083	0.871	⋯	0.079
…	…	…	…	…
Diabetes	0.089	0.037	⋯	0.027
Region	0.015	0.013	⋯	0.023
Family-Hepatitis	0.029	0.097	⋯	0.943

In Table II, values of each row stand for the coefficients of common factors FACT1, ⋯, FACT16, and the linear combination equals the corresponding original variable. For example, the value 0.891 in the row 1 and column 2 indicates that the variable 'Weight' has a positive linear correlation to FACT1, and the correlation is as high as 0.891. The number -0.083 in the row 4, column 2 indicates that the variable 'Education' has a negative linear correlation with FACT1 with a weak correlation coefficient. The whole first row indicates the combination relationship between the original variable 'Weight' and common factors FACT1, ⋯, FACT16 as shown in formula (13).

$$X_{\text{Weight}} = 0.891F_1 + 0.076F_2 + \cdots - 0.07F_{16} + \varepsilon_1 \cdots (13)$$

For one thing, from a horizontal viewpoint, Table II contains the linear combination relationship of 29 independent variables and FACT1, ⋯, FACT16, as shown in formula (14).

$$\begin{cases} X_{\text{Weight}} = 0.891F_1 + 0.076F_2 + \cdots - 0.07F_{16} + \varepsilon_1 \\ X_{\text{Waist}} = 0.888F_1 + 0.058F_2 + \cdots + 0.036F_{16} + \varepsilon_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ X_{\text{Family-Hepatitis}} = 0.029F_1 + 0.097F_2 + \cdots + 0.943F_{16} + \varepsilon_{16} \end{cases} \quad (14)$$

For another, from a longitudinal viewpoint, each column in Table II shows the relationship between the corresponding common factor and all the original variables. For example, in column 2, row 4, the number -0.083 indicates FACT1 has a weak negative linear correlation with the original variable 'Education' (whether it is educated). Considering that the load factor is supposed to be no less than 0.85, the whole common 2 indicates FACT1 heavily depends upon first three variables 'Weight', 'Waist' and 'Obesity' with a positive correlation, therefore FACT1 can be interpreted as an obesity factor. Similarly, FACT2 can be interpreted as an education factor. FACT16 can be understood as a family hepatitis genetic factor.

After dimensionality reduction based on FA, Python is used to divide the dataset into training set and the test set with the proportion of training set to test set being 8: 2 and random numbers are set. The training set has a total of 7028 samples, and the test set has a total of 1757 samples.

VII. PREDIAGNOSIS OF ACUTE LIVER FAILURE BASED ON FOUR MODELS

A. Learning and prediagnosis of XGBoost

First, the hyperparameters are tuned based on the training set. Considering that there is only one sample data for each subject to be diagnosed in the dataset processed in this paper, artificial overlapping phenomenon shall not appear. To save time, cross-validation (CV) is used for the re-sampling selection strategy to divide the data set into k mutually exclusive subsets of similar size. Each time the union of $k-1$ subsets are used as the training set, and the remaining subset is used as the validation set, so that trainings and validations have been performed for k times, and the mean value of the k results are output finally. The command 'KFold' of the package 'sklearn.model_selection' is used to create a CV generator, where the first parameter 'n_splits' represents the number of folds in CV. And 'n_splits' is set as the default value 3. The second parameter 'shuffle' is used to indicate whether the order of samples needs to be shuffled, if yes, it is set as 'True', otherwise it is set as 'False'. The third parameter 'random_state' represents the random number seed, which is set as 123 without loss of generality.

There are a variety of algorithm options for hyperparameter tuning, such as Genetic Algorithm (GA) , Randomized Search CV and GridSearch CV. To simplify this tuning process, GridSearch CV is chosen in this paper. In the package 'sklearn.model_selection', 'GridSearch CV' is set to create a grid search hyperparameter tuner. The parameter 'scoring' represents the model performance index, which is set as 'accuracy' due to the classification tasks in this paper.

The subsequent XGBoost-FA, LightGBM and LightGBM-FA have been set the same parameter for hyperparameter tuning to make comparison among the four models.

Firstly, the search range of hyperparameters param_grid is defined, which includes two parameters of 'max_depth' and 'lambda'. The initial parameters 'max_depth' (the maximum depth of the tree) are set as 2, 3, 4, 5, and 6, and the initial parameter 'lambda' (L2 regular term parameters in the weight of complexity) are set as 2, 3, and 4. The results of hyperparameter tuning are shown in Figure 3.

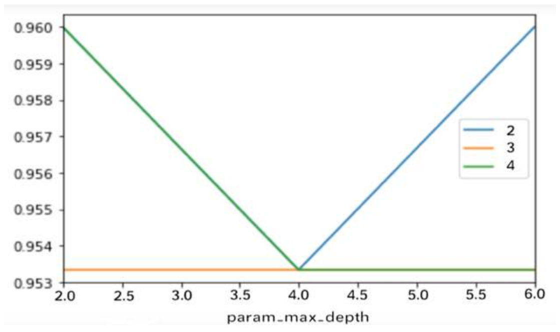

FIGURE 3. **Hyperparameter tuning results of XGBoost**

In Figure 3, the abscissa indicates the depth of the tree and the ordinate indicates the average precision of CV. The blue line stands for 'lambda' with the value being set as 2, the orange with the value being set as 3, and the green with the value being set as 4.

From Figure 3, it can be seen that when 'lambda' = 2, the average precision of CV stays at a lower level and remains the same in the interval 'max_ depth'= [2, 4]. It starts to increase gradually and reaches its maximum value at 'max_depth' = 6 in [4, 6]. When 'lambda' = 3, the average precision of CV stays at a lower level and remains the same in the interval 'max_depth' = [2, 6]. When 'lambda' = 4, the average precision of CV begins to decline gradually and reaches a minimum value at 'max_depth' = 6 in the interval 'max_depth' = [2, 4]. It is at a lower level and remains the same level in [4, 6].

When the value of 'max_depth' is more than 4, the line where 'lambda' = 2 is always on top of the other two lines, which means 'lambda'= 2 is the optimal choice. Besides, when 'lambda' = 2 and 'max_depth' = 6, the average precision of CV is the maximum value. Therefore, the optimal hyperparameter combination is 'max_depth' = 6, 'lambda' = 2.

After the optimal combination of hyperparameters is selected, and the times of iterations is set as 500 to obtain the XGBoost prediagnosis model. A part of the 72nd decision tree is selected as shown in Figure 4 to present a clear visualization effect considering the integrated decision tree is too complex.

The overall structure of Figure 4 is a vertical binary tree. The first node in the tree indicates that 'Age' is the optimal segmentation variable among the 29 feature variables in the current subset of preprocessed data, and the corresponding optimal segmentation point is 23. When 'Age' ≥ 23, it enters the second node of Figure 2. The second node shows that for the subset which does not meet 'Age' < 23, 'Age' is still the optimal segmentation variable among the 29 feature variables, and the corresponding optimal segmentation point is 66 at this time. The output leaf node value is 0.0109748961 when 'Age' < 66. It is 0.00188306055 when the sample does not meet 'Age' < 66. The value of the corresponding leaf nodes (error fitting value) in each tree are added up to the predictive value of the first tree as the final predicted value.

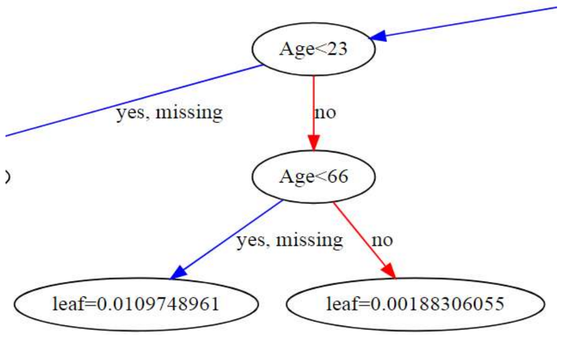

FIGURE 4. **Part of the 72nd tree diagram of XGBoost**

Some of prediagnose on the test set are shown in Table III.

TABLE III
PARTIAL LEARNING RESULTS ON THE TEST SET BASED ON XGBOOST

Sample **N**umber	Prediagnosis of ALF	The **A**ctual **S**ituation of ALF
1	1	1
2	0	0
⋯	⋯	⋯
1757	0	0

Firstly, we use the log loss function to evaluate the effect of the model, as shown in formula (15).

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\left[\sum_{j=1}^{k} I_{\{y^j=j\}}\log\left(h_\theta\left(y^j = j|x\right)\right)\right] \cdots\cdots \quad (15)$$

The number of samples in this example is 1757 and the categories ranges from 0 to 2. Comparing the second and third columns of Table III, the log loss function value of the XGBoost model on the test set is shown in formula (16).

$$logloss = -\frac{1}{1757}\sum_{i=1}^{1757}\left[\sum_{j=0}^{2} I_{\{y^j=j\}}\log\left(h_\theta\left(y^j = j|x\right)\right)\right] = 0.569707 \cdots\cdots \quad (16)$$

The prediagnosis accuracy of XGBoost is 0.75014 and its merror is 0.24986.

Besides, the efficiency of the model is evaluated by the running time. According to results, the running time of XGBoost is 10.5 seconds.

B. Learning and prediagnosis results of XGBoost-FA

Firstly, 'GridSearch CV' is set to create a grid search hyperparameter tuner on the training set. The search range of hyperparameters param_grid is defined in advance, which includes two parameters of 'max_depth' and 'lambda'. Setting the initial parameter 'max_depth' (the maximum depth of the tree) is set as 2, 3, 4, 5, and 6 respectively, and setting the initial parameter 'lambda' (L2 regular term parameters in the weight of complexity) to 2, 3, and 4. The results of hyperparameter tuning are shown in Figure 5.

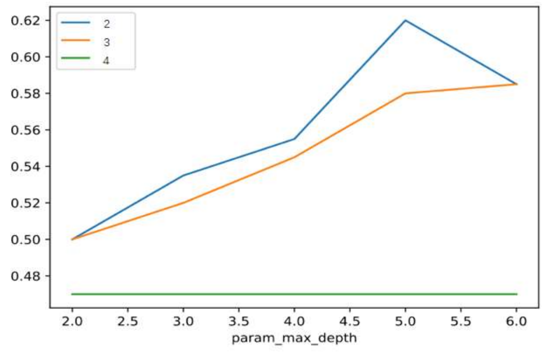

FIGURE 5. Hyperparameter tuning results of XGBoost-FA

In Figure 5, the abscissa indicates the depth of the tree and the ordinate indicates the average precision of the cross-validation. 'Lambda' is represented by lines with three different kinds of color with the blue line standing for 2, the orange line standing for 3, and the green line standing for 4 respectively.

From Figure 5, it can be seen that when 'lambda' = 2, the average precision of CV, starts to rise gradually and reaches its maximum value at 'max_depth' = 5 in the interval max_ depth = [2, 5], then it starts to decrease gradually in [5, 6]. When 'lambda' = 3, the average precision of CV gradually increases in the interval 'max_depth' = [2, 6]. When 'lambda' = 4, the average precision of CV, it is always at a lower level and remains unchanged in [2, 6].

The line where 'lambda' = 2 is always on top of the other two lines, which means 'lambda' = 2 is the optimal choice. Also, when 'lambda' = 2 and 'max_depth' = 5, the average precision of CV is the maximum value. Therefore, the optimal hyperparameter combination is 'max_depth' = 5, 'lambda' = 2.

After the optimal combination of hyperparameters is selected, and the times of iterations is set as 500 to obtain the XGBoost-FA prediagnosis model. To visualize the model, the parameter 'xgb.to_graphviz' is used in this paper. A part of the 77th decision tree is selected as shown in Figure 6 to present the visualization effect considering the integrated decision tree is too complex.

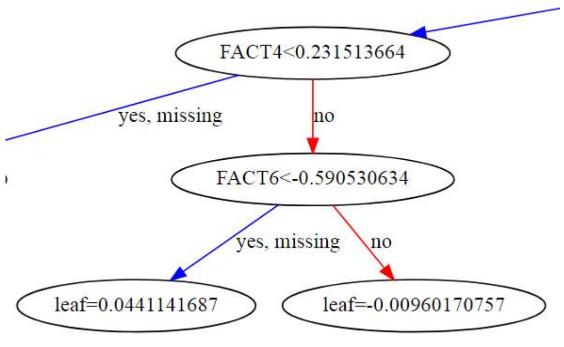

FIGURE 6. Part of the 77th tree diagram of XGBoost-FA

The overall structure of Figure 6 is a binary tree which shows the current subset of data after dimensionality reduction. The first node in the tree indicates that FACT4 is the optimal segmentation variable among 16 common factors in the current data subset, and the corresponding optimal segmentation point is 0.231513664. When FACT4 $\geq$ 0.231513664, it enters the second node of Figure 4. The second node shows that for the subset of FACT4 < 0.231513664, FACT6 is the new optimal segmentation variable among the 16 common factors. In this case, the corresponding optimal segmentation point is -0.590530634. When FACT6 <-0.590530634, the output leaf node value ($\hat{y}_i^{(72)}$) = 0.0441141678. When FACT6 $\geq$ -0.590530634, the output leaf node value ($\hat{y}_i^{(72)}$) = -0.00960170757. The value of the corresponding leaf nodes (error fitting value) in each tree are added up to the predicted value of the first tree as the final predicted value.

Some of prediagnose on test set are shown in Table IV.

TABLE IV
THE PARTIAL LEARNING RESULTS ON TEST SET BASED ON XGBOOST-FA

Sample **N**umber	Prediagnosis of ALF	The **A**ctual **S**ituation of ALF
1	0	0
2	0	1
…	…	…
1757	0	0

According to formula (15) and Table IV, the log loss function value of the XGBoost-FA model on the test set, as is shown in formula (17).

$$logloss = -\frac{1}{1757}\sum_{i=1}^{1757}\left[\sum_{j=0}^{2} I_{\{y^j=j\}}\log\left(h_\theta\left(y^j=j|x\right)\right)\right] = 0.663924\cdots\cdots \quad (17)$$

The prediagnosis accuracy of XGBoost-FA is 0.67786 and its merror is 0.32214.

In addition, the efficiency of the model is evaluated by the running time, and the running time of the XGBoost-FA is 5.7 seconds according to results.

C. Learning and prediagnosis results of LightGBM

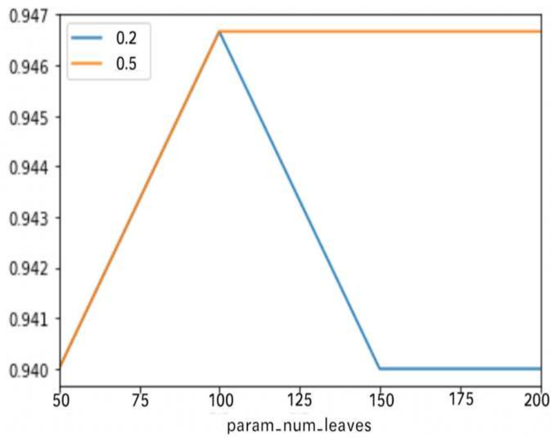

FIGURE 7. Hyperparameter tuning results of LightGBM

Firstly, 'GridSearch CV' is set 'GridSearch CV' to create a grid search hyperparameter tuner on the training set. The search range of hyperparameters param_grid is defined, which includes two parameters of 'num_leaves' and 'lambda'. The initial parameter 'num_leaves' (number of leaf nodes) is set as 50, 100, 150, and 200 respectively, and the initial parameter 'lambda' (L2 regular term parameters in the weight of complexity) is set as 0.2 and 0.5 respectively. The results of hyperparameter tuning are shown in Figure 7.

In Figure 7, the abscissa indicates the number of leaf nodes and the ordinate indicates the average precision of CV. 'Lambda' is represented by lines with different kinds of color with the blue line standing for 0.2, the orange line standing for 0.5 respectively.

According to Figure 7, when 'lambda' = 0.2, the average precision of CV starts to gradually increase and reaches 100 as the maximum value at 100 in the interval 'num_leaves' = [50, 100]. It begins to gradually decrease in [100, 150], and reaches a minimum value in 'num_leaves' =150. The average precision of CV stays at minimum and remains the same in [150, 200]. When 'lambda' = 0.5, the average precision of CV begins to rise gradually and reaches 100 as the maximum in the interval 'num_leaves' = [50, 100]. It stays always at a maximum level and remains unchanged in [100, 200].

When the 'num_leaves' is over 100, the line where 'lambda' = 0.5 is always on top of the other line, which means 'lambda' = 0.5 is the optimal choice. And, when 'num_leaves' >100 and 'lambda' = 0.5, the average precision of CV is the maximum value. Therefore, the optimal hyperparameter combination is 'num_leaves'=200, 'lambda'=0.5.

After the optimal combination of hyperparameters is decided, the boosting type 'boosting_type' is set to 'gbdt' (gradient boosting decision tree) for training the LightGBM prediagnosis model. The 'lgb.create_tree_digraph' command is called in this paper to present the visualized effect. Considering the integrated decision tree is too complicated, a part of the 46th decision tree is selected as shown in Figure 8 to present the visualized effect.

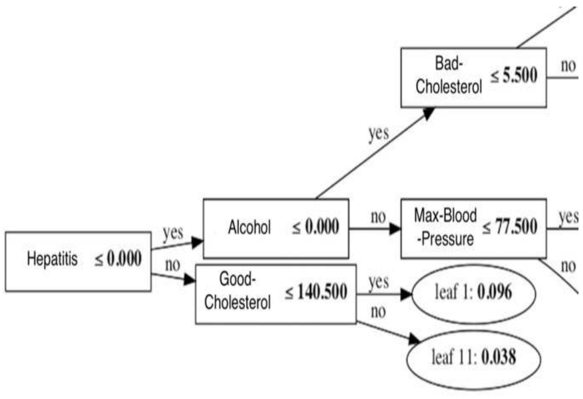

FIGURE 8. Part of the 46th tree diagram of LightGBM

The overall structure of Figure 8 is a horizontal binary tree. The first node in the tree indicates that 'Hepatitis' is the optimal segmentation variable among the 29 feature variables in the current subset of preprocessed data, and the corresponding optimal segmentation point is 0 in this case. When 'Hepatitis' ≤ 0, it enters the second node of Figure 8. The second node shows that for the current subset, 'Alcohol' is the new optimal segmentation variable among the 29 feature variables, and the corresponding optimal segmentation point is 0 at this time. When 'Hepatitis' > 0, it enters the third node in Figure 6. The third node shows that 'Good-Cholesterol' is the new optimal segmentation variable among the 29 feature variables for the current subset and the corresponding optimal segmentation point is 140.500 in this case. The value of the corresponding leaf nodes (error fitting value) in each tree are added up the predictive value of the first tree as the final predictive value.

Some of prediagnose on test set are shown in Table V.

TABLE V
THE PARTIAL LEARNING RESULTS ON TEST SET BASED ON LIGHTGBM

Sample Number	Prediagnosis of ALF	The Actual Situation of ALF
1	2	1
2	0	0
…	…	…
1757	0	0

According to formula (15) and Table V, the value of the log loss function of the LightGBM model on the test set is shown in formula (18).

$$logloss = -\frac{1}{1757}\sum_{i=1}^{1757}\left[\sum_{j=0}^{2} I_{\{y^i=j\}}\log\left(h_\theta\left(y^j = j|x\right)\right)\right] = 0.576059\cdots\cdots \quad (18)$$

The prediagnosis accuracy of LightGBM is 0.75811 and its merror is 0.24189.

In addition, the efficiency of the model is evaluated by the running time, and the running time of LightGBM is 15.1 seconds according to results.

D. Learning and Prediagnosis results LightGBM-FA

Firstly, 'GridSearch CV' is set to create a grid search hyperparameter tuner on the training set. The search range of hyperparameters param_grid is defined, which includes two parameters of 'num_leaves' and 'lambda'. The initial parameter 'num_leaves' (number of leaf nodes) is set as 100, 200, 300, 400, and 500 respectively, and the initial parameter 'lambda' (L2 regular term parameters in the weight of complexity) is set as 0.5 and 0.8 respectively. The results of hyperparameter tuning are shown in Figure 9.

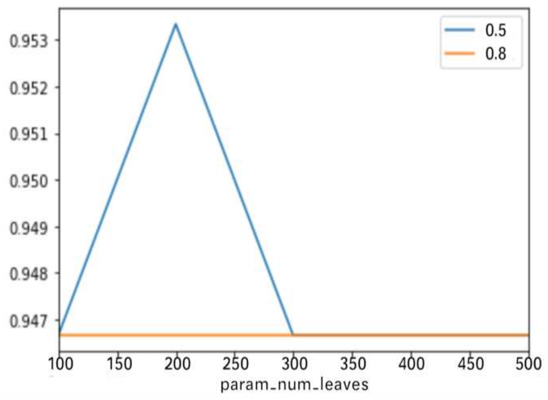

FIGURE 9. Hyperparameter tuning results of LightGBM-FA

In Figure 9, the abscissa indicates the number of leaf nodes and the ordinate indicates the average precision of CV. 'lambda' is represented by lines with different colors with 0.5 being set as blue line and 0.8 being set as orange line.

According to Figure 9, when 'lambda' = 0.5, the average precision of CV starts to gradually increase and reaches 200 as the maximum value in the interval 'num_leaves' = [100, 200]. It begins to gradually decrease and reaches a minimum value at the point of 300 within the range of [200, 300]. The average precision of CV stays at minimum and remains the same in the interval 'num_leaves'= [300, 500]. When 'lambda' = 0.8, the average precision of CV always stays at minimum and remains unchanged.

When the 'num_leaves' is in [100, 300], the line where 'lambda' = 0.5 is always on top of the other line, which means 'lambda' = 0.5 is the optimal choice. when 'num_leaves' = 200 and 'lambda' = 0.5, the average precision of CV is the maximum value. Therefore, the optimal hyperparameter combination is 'num_leaves'=200, 'lambda'=0.5.

After the optimal combination of hyperparameters is decided, the boost type 'boosting_type' is set as 'gbdt' (gradient boost decision tree) for training the LightGBM-FA prediagnosis model. The 'lgb.create_tree_digraph' command is used in this paper to visualize the model. Considering the integrated decision tree is too complicated, a part of the 56th decision tree is selected as shown in Figure 10 to present the visualized effect.

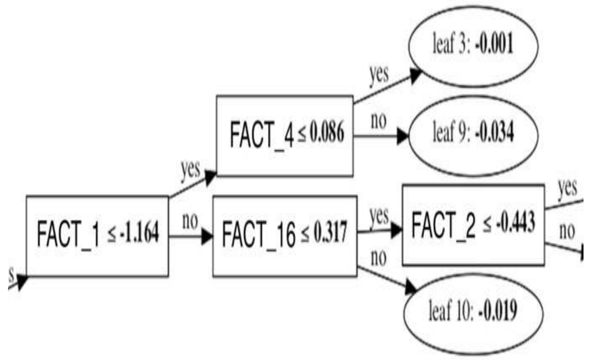

FIGURE 10. Part of the 56th tree diagram of LightGBM-FA

The overall structure of Figure 10 is a horizontal binary tree which shows the current subset of data after dimensionality reduction. The first node in the tree indicates that FACT1 is the optimal segmentation variable among 16 common factors in the current data subset, and the corresponding optimal segmentation point is -1.164. When FACT1 ≤ -1.164, it enters the second node of Figure 10. The second node shows that FACT4 is the new optimal segmentation variable among the 16 common factors for the subset of FACT1 ≤ -1.164. In this case, the corresponding optimal segmentation point is 0.086. When FACT1> -1.164, it enters the third node of Figure 10. The third node shows that FACT16 is the new optimal segmentation variable among the 16 common factors for the current subset. At this time, the corresponding optimal segmentation point is 0.317. The value of the corresponding leaf nodes (error fitting value) in each tree are added up to the predictive value of the first tree as the final predictive value.

Some of prediagnose on test set are shown in Table VI.

TABLE VI
THE PARTIAL LEARNING RESULTS ON TEST SET BASED ON LIGHTGBM-FA

Sample Number	Prediagnosis of ALF	The Actual Situation of ALF
1	2	1
2	0	0
…	…	…
1757	0	0

According to formula (15) and Table VI, the value of the log loss function of the LightGBM-FA model on the test set is shown in formula (19).

$$logloss = -\frac{1}{1757}\sum_{i=1}^{1757}\left[\sum_{j=0}^{2} I_{\{y^i=j\}}\log\left(h_\theta\left(y^j = j|x\right)\right)\right] = 0.676019\cdots\cdots \quad (19)$$

The prediagnosis accuracy of LightGBM-FA is 0.67274 and its merror is 0.32726.

In addition, the efficiency of the model is evaluated by the running time, and the running time of LightGBM-FA is 4.1 seconds according to results.

VIII. RELEVANT COMPARISONS OF PROGNOSTIC RESULTS

A. Comparative analysis of prediagnosis results for XGBoost, XGBoost-FA, LightGBM and LightGBM-FA

The learning results of the four prediagnosis models including XGBoost, XGBoost-FA, LightGBM and LightGBM-FA are summarized, as shown in Table VII.

TABLE VII
PREDIAGNOSED RESULTS OF XGBOOST, XGBOOST-FA, LIGHTGBM AND LIGHTGBM-FA

Model	Accuracy	Log loss	Training time (s)
XGBoost	0.75014	0.569707	10.5
XGBoost-FA	0.67786	0.663924	5.7
LightGBM	0.75811	0.576059	15.1
LightGBM-FA	0.67274	0.676019	4.1

Firstly, in Table VII, LightGBM has the greatest value in terms of accuracy, but the difference of accuracy among the four prediagnostic models is no more than 0.09. The biggest difference is 0.08537 (LightGBM and LightGBM-FA), which indicates that the accuracy of LightGBM decreases largely after using FA dimensionality reduction. And the accuracy of XGBoost-FA is 0.07228 lower than that of XGBoost. Therefore, the accuracy of both XGBoost and LightGBM decreases after coupling FA.

Additionally, the log losses of both XGBoost and LightGBM slightly exceed than those of both XGBoost-FA and LightGBM-FA in Table VII. It could be seen that the log loss will increase after XGBoost and LightGBM coupling FA. That is, the total difference between the predictive value and the actual value will increase.

At last, in terms of training time, the efficiency of the algorithm improves greatly after coupling FA as shown in Table VII. XGBoost-FA is 4.8 seconds shorter than XGBoost. LightGBM-FA is 11 seconds shorter than LightGBM. Among them, LightGBM-FA runs most efficiently among the four models, but its accuracy is the lowest. Therefore, LightGBM-FA is generally not recommended.

Comparing the training time of XGBoost-FA and LightGBM, XGBoost-FA (5.7 s) is 9.4 seconds shorter than LightGBM (15.1s), which means that efficiency of external dimensionality reduction is increases by about 1.6 times than built-in dimensionality reduction with respect to XGBoost. In terms of accuracy, XGBoost-FA is almost 0.08 lower than LightGBM. Therefore, in cases with lower run time requirements, LightGBM is the best choice to ensure the highest accuracy. For applications with higher efficiency requirements, such as instant diagnostics, XGBoost-FA is preferable.

B. Comparing the algorithm on the Kaggle and the algorithms used in this paper

For the Acute Liver Failure dataset, the prediction algorithm from Kaggle (author: kernel4d6502ba2a) has removed all of the sample data which has not yet been diagnosed as having acute liver failure or has missing values. The algorithms on the Kaggle use LRC based on the remaining 4034 sample data which has labeled variables consisting of only two categories, 0 (diagnosed as not having acute liver failure) and 1 (diagnosed as having acute liver failure).

The comparison of the algorithm on the Kaggle with the methods used in this paper is shown in Table VIII.

TABLE VIII
COMPARISON OF ALGORITHMS FROM KAGGLE AND THIS PAPER

Model	Samples	Issues addressed	Training time (s)	Accuracy
LRC (Kaggle Algorithm)	4034	binary classification	3.1	0.82
XGBoost-FA	8785	multi-class classification （three）	5.7	0.67786
LightGBM	8785	multi-class classification （three）	15.1	0.75811

In terms of accuracy, the official website's public algorithm LRC is 0.14214 higher than XGBoost-FA and 0.06189 higher than LightGBM. In terms of training time, LRC is 2.6 seconds shorter than XGBoost-FA and 12 seconds shorter than LightGBM. However, the algorithms used in this paper processes about twice as much data as the publicly available algorithm on the official website, while the running time of XGBoost-FA is not twice as long as the official website's public algorithm. Thus, XGBoost-FA is more efficient than the official website's algorithm.

The reason why the accuracy of the algorithm used in this paper is lower than that of Kaggle falls into two parts. Firstly, the problem dealt with in this paper is multi-class classification (including 'Confirmed not ill', 'Confirmed ill' and 'Not yet confirmed to be ill'), compared with the binary classification on the official website (including 'Confirmed not ill', 'Confirmed to be ill'), which are more complex. Secondly, the data dealt with in this paper consists of variables with a large number of missing values, of which numeric variables have been filled in with the mean while categorical variables have been filled in with the mode. Although being affected by the above two reasons, the accuracy of LightGBM still reaches 92% of the accuracy of the official website algorithm while the accuracy of XGBoost-FA also reaches 83% of the accuracy of the official website algorithm.

All things considered, the official algorithm is advised to fix the issue of simpler binary classification and no missing values.

While dealing with multi-classification problems or data with a large number of missing values, XGBoost-FA is preferred to have better operating efficiency. LightGBM is preferred to have a higher prediagnosis accuracy.

IX. APPLICATION AND IMPLICATION ABOUT PREDIAGNOSIS OF ACUTE LIVER FAILURE

A. Significance of prediagnosis of acute liver failure

It’s common to suffer acute liver failure in modern lives. Its timely and accurate diagnosis is conducive to effectively controlling and treating the disease, as well as increasing the probability of recovery. Take these into consideration, the data set of acute liver failure is selected as a representative example in this paper.

Clinically, the diagnosis of acute liver failure requires a series of rigorous clinical tests such as liver puncture and prothrombin time (PT) measurement. Experienced doctors are also needed to judge the severity of liver damage. Therefore, the diagnosis efficiency of acute liver failure depends upon the two factors including medical conditions and doctor's diagnosis experience.

Based on the characteristic variables of a sample, this paper can predict whether the patient suffers from acute liver failure, which is helpful for doctors in prediagnosis before medical testing. This prediagnosis has nothing to do with the medical conditions of the hospital or the doctor’s diagnostic experience. Instead, it provides rigorous scientific support for the diagnosis of acute liver failure.

B. Significance of model application

From the perspective of model comparison, LightGBM has two built-in algorithms with dimensionality reduction and sampling functions, EFB and GOSS. Compared with XGBoost, LightGBM is more powerful. But compared with XGBoost-FA, LightGBM is slightly higher in prediction accuracy and is almost as three times in training time. Additionally, the dimensionality reduction effect of LightGBM is not as direct as the traditional dimensionality reduction tool FA. The reason lies in the extent to which the original information is retained could be decided according to the results of FA. At last but not least, the newly generated common factors are explanatory variables with practical meaning. Therefore, with respect to XGBoost, external dimensionality reduction (XGBoost-FA) is preferred to built-in dimensionality reduction(LightGBM).

Based on the prediagnosis model of this paper, the common factors FACT1 is an important indicator to identify acute liver failure (Section VI.), which is a fusion of the three original variables of Weight, Waist and Obesity with a positive impact on it. Therefore, doctors can use obesity as an important indicator for prediagnosing acute liver failure while residents can use weight control as an important measure to prevent acute liver failure.

In addition, doctors can make three corresponding judgments based on the prediagnosis results of the samples in this paper. When the predicted result is 0, the prediagnosis sample is not ill. It is recommended that the subject should be observed at home and the scarce medical resources should be saved for more urgent patients. When the predicted result is 1, the prediagnosed patient has acute liver failure. At this time, the patient should be arranged for hospitalization as soon as possible to be treated timely. When the predicted result is 2, it is deduced that the subject is suspected of being ill, and further medical examinations and hospitalization observations are required.

LightGBM-FA is recommended as shown from Table VII that the model takes the shortest time. In dealing with multi-category with missing values, in order to balance efficiency and accuracy, XGBoost-FA is advised as a prediagnosis model. According to TABLE VIII, XGBoost-FA has the shortest prediagnosis time, and the prediagnosis accuracy of XGBoost-FA is higher than that of LightGBM-FA.

In the dataset of Acute Liver Failure used in this paper, more than 50% of the samples have missing values. It's easy to find the common constraints of various factors such as the scope of statistics, indicators and objects, imperfect data with large sample sizes and high dimensions in real life. It is difficult to collect clean and perfect data in real life set. In another word, it truly reflects the authenticity of the prediction results and the actual situation of our work. It's unavoidable to encounter relatively unsatisfactory prediction results for this kind of unsatisfactory data.

To deal with multi-category prediagnosis problems with so many missing values, FA is used to remove redundant variables to increase training efficiency and accuracy. XGBoost-FA is selected as the diagnostic model and obtained good results, with the accuracy rate 0.67786 and training time 5.7 seconds.

C. Significance of applying algorithms to AI Medical

In many medical scenarios, people are usually troubled by the imperfect of the public medical management system, high medical costs, few channels, and low coverage. AI (artificial intelligence) Medical is designed to integrate the multi-dimensional data of different systems and departments in the hospital via the AI technology including expert systems, medical natural language processing and data mining.

AI medical can be developed in two directions. Firstly, AI medical can be improved on more multiple source data such as voice medical records, image data, and multiple indicators of the body. Secondly, algorithm updating can improve AI medical in technology. Both should be organically combined to complement each other.

As a popular and cutting-edge boosting integrated algorithm, XGBoost can help AI medical improved in technology. Because the newly generated common factors by FA can be explained by practical meaning, it is easier for doctors to understand and explain the predicted outcome by FA. Applying XGBoost-FA to AI Medical can help doctors proactively participate in diagnosis of many diseases with higher efficiency and accuracy.

Take tumor surgery as an example. The planning process for tumors often takes long hours. If machine learning algorithms can help doctors improve diagnosis accuracy in a shorter time, it will undoubtedly greatly reduce the workload of doctors and increase the success rate of surgery.

D. The constraints of machine learning combined with AI Medical

First of all, only when the accuracy of AI Medical products in actual medical applications satisfies a certain standard, it is eligible to enter the clinic.

In addition, AI Medical care may encounter an unavoidable problem, that is, whether data acquisition leads to the violation of patient's privacy.

Thirdly, there is a lack of comprehensive and unified standards for issues such as defects in AI diagnosis and judgment basis for medical negligence.

X. CONCLUSION

This paper mainly compared the accuracy and efficiency of built-in dimensionality reduction (LightGBM), external dimensionality reduction (XGBoost-FA), and built-in combined with external dimensionality reduction (LightGBM-FA).

Based on the dataset of Acute Liver Failure for classification and prediagnosis, FA is used to realize dimensionality reduction to remove redundant information of variables. And then the prediagnosis results are obtained in combination with XGBoost and LightGBM learning. The prediagnosis examples for Acute Liver Failure show that XGBoost-FA and LightGBM-FA have greatly improved the efficiency of prediagnosis compared with single XGBoost and LightGBM, but their accuracy has decreased. At the same time, the results show that although the accuracy of the built-in dimensionality reduction algorithm is slightly higher than that of the external one, its training time will be doubled. Based on the prediagnosed effect, XGBoost-FA performs better while dealing with multi-classification problems or data with a large number of missing values.

One thing worth mentioning is the imperfect medical-related data feature in AI Medical. Although dimensionality reduction is an extremely common step in data processing, its role is crucial. The effect of dimensionality reduction is related to the accuracy and efficiency of subsequent models. Embedding some dimensionality reduction algorithms in the model does improve the accuracy a little, but it sacrifices the operating efficiency of many models. Traditional dimensionality reduction tools could be selected according to different application requirements.

More examples of AI Medical and theoretical comparison of the two algorithms are expected. From the perspective of data dimensionality reduction processing, FA is chosen in this paper. However, dimensionality reduction methods vary with different types of data. The impact of dimensionality reduction methods including Chi2, MI, LDA and autoencoder dimensionality reduction, etc. upon the machine learning models needs to be further explored.

REFERENCES

[1] D. Jin，Y. Lu and J. Qin, "SwiftIDS: Real-time intrusion detection system based on LightGBM and parallel intrusion detection mechanism," *Computers & Security*, vol. 97, 2020.

[2] Y. Xie, W. Xiang, M. Ji, J. Peng, and Y. Huang, "Application Analysis of Predicting Monthly House Rental Based on Xgboost and LightGBM Algorithms," *Computer Applications and Software*, vol. 36, no. 9, pp. 151-155, 2019.

[3] A. B. Parsa, A. Movahedi, H. Taghipour, S. Derrible, A. Mohammadian, "Toward safer highways, application of XGBoost and SHAP for real-time accident detection and feature analysis," *Accident Analysis and Prevention*, vol. 136, pp. 23-27, 2020.

[4] A. Samat，E. Li，W. Wang，S. Liu，and J. Abuduwaili, "Meta-XGBoost for Hyperspectral Image Classification Using Extended MSER-Guided Morphological Profiles," *Remote Sensing*, vol. 12, no.12, pp. 1973, 2020.

[5] D. Zhang and Y. Gong, "XGBoost compares neural network and random forest coupling factor analysis to predict acute liver failure," *Mathematics in Practice and Theory*, vol. 50, no. 13, pp.141-152, 2020.

[6] W. Zhou, Y. Wang, C. Li, H. Xiao, and S. Xing, "Application of LightGBM algorithm to classification of structural magnetic resonance imaging in Alzheimer's disease," *Chinese Journal of Medical Physics*, vol. 36 no.4, pp. 408-413, 2019.

[7] Q. Shen, F. Shao, and R. Sun, "A Breast Cancer Prediction Model Based on XGBoost," *Journal of Qingdao University (Natural Science Edition)*, vol.32, no. 1, pp. 95-100, 2019.

[8] F. Leng and W. Li, "Classification and prediction of lung squamous cell carcinoma and lung adenocarcinoma based on XGBoost," *Journal of Capital Medical University*, vol. 12, pp. 1-5 ,2019.

[9] G. Yang, X. Xu, and F. Zhao, "User Rating Forecast Model and Application Based on XGBoost Algorithm," *Data Analysis and Knowledge Discovery*, vol. 3, no.1, pp. 118-126, 2019.

[10] J. Guo, L. Yang, R. Bie, J. Yu, Y. Gao, Y. Shen, and A. Kos, "An XGBoost-based physical fitness evaluation model using advanced feature selection and Bayesian hyper-parameter optimization for wearable running monitoring," *Computer Networks*, vol. 151, pp. 166-180, 2019.

[11] Y. Zhang, C. Feng, K. Li, Z. Zhang, D. Cao, and T. Li, "LightGBM Prediction Model for the Risk of Acute Renal Injury in ICU Patients," *Journal of PLA Medical College*, vol. 40, no. 4, pp. 316-320. 2019.

[12] L. Ali, C. Zhu, Z. Zhang and Y. Liu, "Automated Detection of Parkinson’s Disease Based on Multiple Types of Sustained Phonations Using Linear Discriminant Analysis and Genetically Optimized Neural Network," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 7, pp. 1-10, 2019.

[13] L. Ali, C. Zhu, and M. Zhou, "Early Diagnosis of Parkinson's Disease from Multiple Voice Recordings by Simultaneous Sample and Feature Selection," *Expert Systems with Applications*, pp. 137, 2019.

[14] L. Ali, C. Zhu, N. A. Golilarz, A. Javeed, M. Zhou, and Y. Liu, "Re- liable parkinsons disease detection by analyzing handwritten drawings: Construction of an unbiased cascaded learning system based on feature selection and adaptive boosting model," *IEEE Access*, pp. 1 – 1, 2019.

[15] A. Souri, A. Mohamm，and M. Potrus, "Formal Verification of a Hybrid Machine Learning-Based Fault Prediction Model in Internet of Things Applications," *IEEE Access*, vol. 8, pp. 23863-23874, 2020.

[16] X. Kong, L. Ma, and H. Bo, "Research on EEG feature extraction and recognition method combining CNN and CSP," *Signal Processing*, vol. 034, no. 2, pp. 164-173, 2018.

[17] Z. Huang, Y. Wang, and Q. Wang, "Research on automatic characterization of cardiovascular atherosclerotic plaque tissue based on intravascular ultrasound images," *Computer Science*, vol. 45, no. 5, pp. 260-265, 2018.

[18] Z. Zhang, R. Wang, M. Wei, and J. Zhou, "Facial expression recognition method based on stacked hybrid autoencoder," *Computer Engineering and Applications*, vol. 55, no. 13, pp.140-144, 2019.

[19] L. Ali, and S. Bukhari, "An Approach Based on Mutually Informed Neural Networks to Optimize the Generalization Capabilities of Decision Support Systems Developed for Heart Failure Prediction," *IRBM*, 2020.

[20] C. Zhang, B. Xie, Y. Du, J. Fan, Y. Liu, S. Yu, and Y. Fang, "The relationship between DNA methyltransferase 3B gene and dopamine D1 receptor gene interaction and schizophrenia," *Chinese Medical Journal*, no. 43, pp. 3059-3062, 2010.

[21] S. Weng, C. Zhang, and X. Zhang, "The application of nonlinear dimensionality reduction in high-dimensional medical data processing," *Journal of Tsinghua University (Natural Science Edition)*, no. 4, pp. 54-57, 2004.

[22] Y. Gong, C. Du, Y. Zhang, and L. Yu, "Prediction of blood glucose level based on principal components and GBDT, " *Mathematics in Practice and Theory*, vol. 49, no. 14, pp. 116-122, 2019.

[23] M. Eftekhari, F. Saberi-Movahed and A. Mehrpooya, "Supervised feature selection via information gain, maximum projection and minimum redundancy," in *SLAA10 Seminar on Linear Algebra and its Applications*, pp. 29-35, 2020.

[24] F. Saberi-Movahed, M. Eftekhari and M. Mohtashami, "Supervised feature selection by constituting a basis for the original space of features and matrix factorization," *International Journal of Machine Learning and Cybernetics*, vol. 11, no.7, pp. 1405-1421, 2020.

[25] S. Wang, C. Gao, Q. Zhang, V. Dakulagi, and B. Zong, "Research and experiment of radar signal support vector clustering sorting based on feature extraction and feature selection," *IEEE Access*, vol. 1, no. 1, pp. 99, 2020.

[26] X. Li, H. Zhang，R. Zhang, and F. Nie, "Discriminative and Uncorrelated Feature Selection with Constrained Spectral Analysis in Unsupervised Learning," *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, vol. 29, no.1, pp. 2139-2149, 2020.

[27] Y. Wang, B. Lei, A. Elazab, and E. Tan, "Breast Cancer Image Classification via Multi-Network Features and Dual-network Orthogonal Low-rank Learning." *IEEE Access*, vol. 99, PP. 1-1, 2020.

[28] Z. Liu, Q. Zhan, and G. Tian, "A Review of Comprehensive Evaluation of Factor Analysis," *Statistics and Decision*, vol. 5, no. 19, pp. 68-73, 2019

[29] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*,2016.

[30] G. Ke, Q. Meng, and T. Finley, "Light GBM A highly efficient gradient boosting decision tree," *Annual Conference on Neural Information Processing Systems*, 2017.

Dongyang Zhang received the B.S. degree in Nanyang Normal University, Henan, China, in 2018. She continued her graduate studies in Wuhan University of Science and Technology, Hubei, China. Her research interests include AI Medical and decision, machine learning, artificial intelligence, statistics and economic statistics.

Yicheng Gong received the B.S. degree in Hubei University, Hubei, China, in 1998, the M.S. degrees in Wuhan University of Science and Technology, in 2004, and the Ph.D. degree in Wuhan University, Hubei, China, in 2012. She is currently an Associate Professor in the School of Science, Wuhan University of Science and Technology, China. Her current research interests include AI Medical and decision, Game machine learning, machine learning, artificial intelligence, statistics and economic statistics.