Bioinformatics, YYYY, 0–0
doi: 10.1093/bioinformatics/xxxxx
Advance Access Publication Date: DD Month YYYY
Manuscript Category

Systems biology

Optimization of drug-target affinity prediction methods through feature processing schemes

Xiaoqing Ru [1], Quan Zou [2,3] and Chen Lin [4,*]

[1]Department of Computer Science, University of Tsukuba, Tsukuba,Japan., [2]Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, China., [3]Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou, Zhejiang, China., [4]School of Informatics, Xiamen University, Xiamen, Fujian, China.

* Corresponding author: chenlin@xmu.edu.cn

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Numerous high-accuracy drug-target affinity (DTA) prediction models, whose performance is heavily reliant on the drug and target feature information, are developed at the expense of complexity and interpretability. Feature extraction and optimization constitute a critical step that significantly influences the enhancement of model performance, robustness, and interpretability. Many existing studies aim to comprehensively characterize drugs and targets by extracting features from multiple perspectives; However, this approach has drawbacks: i. An abundance of redundant or noisy features. ii. The feature sets often suffer from high dimensionality.

Results: In this study, to obtain a model with high accuracy and strong interpretability, we utilize various traditional and cutting-edge feature selection and dimensionality reduction techniques to process self-associated features (SAFs) and adjacent associated features (AAFs). These optimized features are then fed into learning to rank (LTR) to achieve efficient DTA prediction. Extensive experimental results on two commonly used datasets indicate that, among various feature optimization methods, the regression tree-based feature selection method is most beneficial for constructing models with good performance and strong robustness. Then, by utilizing Shapley Additive Explanations (SHAP) values and the incremental feature selection (IFS) approach, we obtain that the high-quality feature subset consists of the top 150-dimensional features and the top 20-dimensional features have a breakthrough impact on the DTA prediction. In conclusion, our study thoroughly validates the importance of feature optimization in DTA prediction and serves as inspiration for constructing high-performance and high-interpretable models.

Availability: https://github.com/RUXIAOQING964914140/FS_DTA.

Contact: ruxiaoqing2020@gmail.com

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

In drug-target interaction research, wet experiments have been complemented by computational methods due to time and resource constraints (Chen, et al., 2016). A variety of computational concepts and methods have been applied to drug-target affinity (DTA) prediction and have achieved progressive results (Chen, et al., 2018; Chen, et al., 2023; Huang, et al., 2020; Yamanishi, et al., 2010). Upon summarizing the characteristics of existing methods, we find that suboptimal performance can be attributed to two main factors: i. the sparsity of samples, an inherent obstacle that cannot be immediately overcome or resolved in the short term within this research topic. ii. inappropriate design of various steps during the model construction. Consequently, most studies concentrate on aspects of design and model construction that can be improved through human intervention (Wang and Zou, 2021; Chen, et al., 2023; Luo, et al., 2017; Mei, et al., 2013; Pahikkala, et al., 2015; Ru, et al., 2022; Yuan, et al., 2022; Zeng, et al., 2021).

The process of building a machine learning model for DTA prediction can be divided into four steps: data collection and processing, feature extraction and optimization, learner selection, and model training and testing. Feature extraction and optimization is an important step that has a significant impact on improving model performance, enhancing robustness, and increasing interpretability. Many existing studies aim to comprehensively characterize drugs and targets by extracting features from multiple perspectives [7-9, 13-15], but this approach has its drawbacks: i. Inevitable information overlap between differing perspectives may lead to an abundance of redundant or noisy features. ii. Feature sets extracted from multiple perspectives tend to have high dimensionality, which requires more storage space and potentially leads to the curse of dimensionality or overfitting of the training data.

In this study, we first extract self-associated features (SAFs) and adjacent associated features (AAFs) of drugs and targets based on their similarity and sharing properties. Then SAFs and AAFs are optimized by eliminating low variance features and employing methods such as Principal Component Analysis (PCA), Least Absolute Shrinkage and Selection Operator (Lasso), ivis ,XGBoost ,LightGBM ,and Catboost. Subsequently, the optimized feature sets are input into a learning to rank (LTR) algorithm—MART for DTA prediction. The overall architecture of this study is illustrated in Fig. 1. Extensive experiments on two widely-used datasets show that XGBoost, LightGBM, and Catboost are most effective methods for achieving good performance and strong robustness.

To enhance the model's interpretability and inspire future research, we assess feature importance based on Shapley Additive Explanations (SHAP) values, and find that the top 150-dimensional features constitute the high-quality feature set and the top 20-dimensional features have a breakthrough impact on the DTA prediction for two datasets through the incremental feature selection (IFS) approach. Furthermore, we observe the common features appearing in the top 20 features ranked by XGBoost, LightGBM, and Catboost, and analyze the type of these features. It can be concluded that different types of features exhibit distinct performances under different circumstances, but both AAFs and SAFs positively influence model performance.

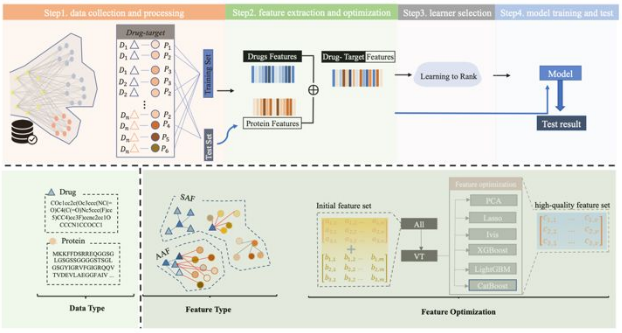

Fig. 1. The overall architecture of this study.

2 Materials and methods

2.1 Feature extraction

AAFs and SAFs are extracted according to the drug-drug similarity, drug-drug sharing, protein-protein similarity, protein-protein sharing, and drug-protein binding affinity information. Drug-drug similarity is the 2D chemical structure similarity calculated with the structure clustering server at PubChem. Protein-protein similarity is the sequence similarity represented by the normalized Smith-Waterman [16] value. Drug-drug sharing is the number of shared targets between two drugs. Protein-protein sharing is the number of shared drugs between proteins.

The representation and source of SAFs are as follows:

$$SAF = \{\partial_i \oplus \beta_i \oplus \gamma_1\} \tag{1}$$

And,

$$\{\partial_1, \partial_2, \partial_3, \partial_4, \partial_5 | \partial_i \in f_i[S]\} \tag{2}$$

$$\{\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_{7-11}, \beta_{12-16} | \beta_i \in f_i[A^-]\} \tag{3}$$

$$\left\{\gamma_1 | \gamma_1 = {\sum_{n=1}^{m} a_{1n}} \Big/ {m}, a_{1n} \in A\right\} \tag{4}$$

Where S represents the similarity matrix between drugs or between targets, A represents the affinity matrix between drugs and targets, A^- represents the matrix obtained after removing unknown affinity elements from A . f_i represents a function that obtains special values of a set. ∂_1-∂_5 represent the mean, 50th percentile, 75th percentile, 85th percentile, and 95th percentile of the elements in the S, respectively. β_1-β_6 represent the mean, count, mode, 25th percentile, 50th percentile, and 75th percentile of the elements in the A^-, respectively. β_{7-11} represent the higher 5 affinity values, respectively. β_{12-16} represent the lower 5 affinity values, respectively.

AAFs are features of the target object that are derived from the SAFs of its neighbors. There are two ways to obtain neighbors: i. Drugs or targets that exhibit a level of similarity or sharing with the target object that exceeds a predefined threshold. ii. The top 5 drugs or targets with the utmost similarity to the target object, as well as the top 5 drugs or targets most similar to each of these 5 drugs or targets.

2.2 Feature optimization

In this study, various feature selection and dimensionality reduction techniques are employed to optimize the initially obtained feature set, including commonly used technique like PCA, a deep learning-based tool like ivis , and several regression-based methods such as LASSO, XGBoost, LightGBM, and Catboost. Notably, XGBoost, LightGBM, and Catboost are GBDT variants that adopt an underlying principle: A strong learner is formed by integrating multiple weak learners.

2.2.1 GBDT

GBDT, a classic version of the gradient boosting algorithm, operates on iteratively enhancing predictive accuracy by integrating multiple decision trees. Each constituent tree is constructed in response to the residuals produced by its immediate predecessor, with the ultimate prediction being the cumulative sum of predictions made by all regression trees (Rao, et al., 2019; Ye, et al., 2009).

The CART algorithm is the base learner in GBDT. A regression CART tree $T(x,\theta)$ can be defined as follows (Liang, et al., 2020; Rao, et al., 2019) ：

$$T(x,\theta) = \sum_{j=1}^{J} c_j I\ (x \in R_j) \tag{5}$$

where J represents the leaf node tree, c_j denotes the predicted value output by the regression tree, I stands for the event function, and R_j represents the set of predicted values of the leaf nodes.

Given $\{x_i,y_i\}_{i=1}^{N}$ denotes the features x_i and label y_i of the samples, where N represents the number of samples. The calculation process of GBDT can be described as follows:

Firstly, the initial tree $f_1(x)$, which is obtained by directly fitting the target value using the regression tree, is defined as follows (Liang, et al., 2020; Rao, et al., 2019):

$$f_1(x) = \underset{c}{\operatorname{argmin}} \sum_{i=1}^{N} L(y_i,c) \tag{6}$$

where L is the loss function, and c is the initial constant value.

Then, the 2nd to m-th regression trees are obtained through iterative training. The training objective for each tree is computed during this process in the following manner (Liang, et al., 2020; Rao, et al., 2019):

$$r_{mi} = y_i - f_{m-1}(x_i) = -[\frac{\partial L(y_i,f(x_i))}{\partial f(x_i)}]_{f(x) = f_{m-1}(x)} \tag{7}$$

The regression tree at the m-th iteration is defined as follows (Liang, et al., 2020; Rao, et al., 2019):

$$\begin{aligned} f_m(x) &= f_{m-1}(x) + T_m(x,\theta_m) \\ &= f_{m-1}(x) - \sum_{j=1}^{J} c_{mj} I\ (x \in R_{mj}) \end{aligned} \tag{8}$$

and (Liang, et al., 2020; Rao, et al., 2019),

$$c_{mj} = \underset{c}{\operatorname{argmin}} \sum_{x \in R_{mj}} (r_{mi} - T_m(x_i,\theta_m))^2 \tag{9}$$

Finally, the predicted result is the sum of these regression trees (Liang, et al., 2020; Rao, et al., 2019):

$$F(x) = \sum_{m=1}^{M} f_m(x) \tag{10}$$

2.2.2 XGBoost

XGBoost (Chen and Guestrin, 2016) is an upgraded version of GBDT. It introduces several optimizations in algorithm implementation, such as feature parallelization, cache awareness, and approximate algorithms for splittable nodes, to accelerate training and improve accuracy. Moreover, XGBoost applies L1 and L2 regularization techniques along with penalties based on feature importance to mitigate the propensity toward overfitting.

The objective function within XGBoost can be formulated as (Al Daoud, 2019; Liang, et al., 2020):

$$O = \sum_{i=1}^{N} L(y_i,\tilde{y}_i) + \sum_{m=1}^{M} \Omega(f_m) \tag{11}$$

and (Liang, et al., 2020),

$$\tilde{y}_i = \sum_{m=1}^{M} f_m(x_i) = \tilde{y}_i^{m-1} + f_M(x_i) \tag{12}$$

$$\sum_{m=1}^{M} \Omega(f_m) = \sum_{j=1}^{M-1} \Omega(f_j) + \Omega(f_m) \tag{13}$$

where L is the loss function, $\tilde{y}_i$ represents the final predicted value of the i-th sample, and $\Omega(f_m)$ represents the regularization term for m iterations.

The objective function, along with the regularization term, is desired to be as small as possible. To find the f_M that minimizes the objective function, XGBoost performs Taylor expansion of the loss function at the point $\tilde{y}_i^{m-1}$. Consequently, the objective function is as follows (Al Daoud, 2019; Liang, et al., 2020):

$$\begin{aligned} O &= \sum_{i=1}^{N} \left[g_i f_M(x_i) + h_i f_M^2(x_i)\right] + \Omega(f_M) \\ &= \sum_{i=1}^{N} \left[g_i W_{q(x_i)} + \frac{1}{2} h_i W_{q(x_i)}^2\right] + \Omega(f_M) \end{aligned} \tag{14}$$

$$\Omega(f_M) = \gamma T + \frac{1}{2}\lambda \sum_{t=1}^{T} (w_t)^2 \tag{15}$$

where g_i, h_i are the first derivative and second derivative of the loss function respectively. $W_{q(x_i)}$ represents the function assigning sample $q(x_i)$ in the tree to its corresponding leaf node. $f_m(x_i)$ represents the prediction result of the m-th tree for the sample x_i.

2.2.3 LightGBM

LightGBM is an additional GBDT variant that builds upon XGBoost and accomplishes three optimizations, as shown in the following formula (Al Daoud, 2019; Ke, et al., 2017; Liang, et al., 2020):

$$LightGBM = XGBoost + Histogram + GOSS + EFB \tag{16}$$

Histogram, a bucketing algorithm, reduces the quantity of candidate split points to decrease computational complexity and memory consumption. Gradient-based One-Side Sampling (GOSS) optimizes the feature sampling process by selecting samples with a gradient absolute value below a certain threshold while preserving samples with large absolute values. Exclusive Feature Bundling (EFB) improves efficiency by re-encoding the values of certain features and bundling multiple mutually exclusive features into a new feature.

2.2.4 CatBoost

CatBoost (Hancock and Khoshgoftaar, 2020) is an emerging GBDT variant that utilizes symmetric tree regularization techniques to tackle overfitting problems arising from direct calculations in multiple dataset permutations. Moreover, CatBoost is specifically designed to adeptly manage categorical features within GBDT, it effectively eliminates the impact of low-frequency features and noise in categorical variables on decision tree generation by creatively considering a prior distribution term when calculating node gains (Prokhorenkova, et al., 2018), as demonstrated in the following equation (Al Daoud, 2019; Prokhorenkova, et al., 2018):

$$\hat{x}_k^i = \frac{\sum_{j=1}^{p-1} [x_{\sigma_j,k} = x_{\sigma_p,k}] Y_{\sigma_j} + aP}{\sum_{j=1}^{p-1} [x_{\sigma_j,k} = x_{\sigma_p,k}] + a} \tag{17}$$

where σ_j represents the j-th data point, $\hat{x}_k^i$ denotes the k-th discrete feature of the i-th row of data in the training set, a is a prior weight, and p is the prior distribution term, [ ] represents an indicator function.

2.2.5 ivis

Ivis (Tian and Tao, 2020) is a nonlinear technique based on Siamese Neural Networks (SNN). SNN consists of three identical networks, with each network composed of three dense layers and an embedding layer. The embedding layer is configured to a size of 2, aiming to project high-dimensional data into a two-dimensional (2D) space. The weights of these layers are initialized using the LeCun normal distribution.

For the embedding layer, a linear activation function is used, and the weights are initialized using Glorot uniform distribution. To mitigate overfitting, each dense layer is accompanied by a dropout layer with a default dropout rate of 0.1. The scaled Exponential Linear Units (SELU) activation function is applied to the dense layers, defined as follows (Klambauer, et al., 2017):

$$selu(x) = \lambda\begin{cases} x, & if\ x > 0 \\ aexp(x) - a, & if\ x \leq 0 \end{cases} \qquad (18)$$

The triplet loss function is employed as the training loss function for the model. K-nearest neighbors (KNN)(Zhang, 2016) are used to generate data for the triplet loss function , with the tuning parameter k set at 100. During each iteration, a point from the dataset is chosen as the anchor. A positive point is randomly selected from the k closest neighbors around the anchor, and a negative point is randomly chosen outside of these neighbors. The objective of the triplet loss function is to minimize the distance between the anchor and the positive point while simultaneously maximizing the distance between the anchor and the negative point. The triplet loss function can be obtained as follows (Tian and Tao, 2020):

$$L_{tru}(\theta) = \left[\sum_{a,p,n} D_{a,p} - \min\left(D_{a,n}, D_{p,n}\right) + m\right] \qquad (19)$$

where a,p,n represent the anchor, positive, and negative points, respectively, D signifies the euclidean distance and m represents the margin.

2.3 Feature importance calculation

Shapley Additive Explanations (SHAP) (Lundberg and Lee, 2017) is a method rooted in game theory designed to improve machine learning model interpretability and performance assessment. SHAP assigns an importance value to each feature in relation to a specific prediction. Aggregating these SHAP values across all instances provides a global explanation of the model's behavior.

For a given feature i, its corresponding Shapley value can be expressed as follows (Lundberg and Lee, 2017):

$$\emptyset_i = \sum \frac{|S|!(n - |S| - 1)}{n!}\left[f(S \cup \{i\}) - f(S)\right] \qquad (20)$$

where n represents the number of features, S represents the feature subset that does not include feature i, $|S|$ denotes the size of the feature subset, $f(S)$ indicates the model prediction value with the feature subset S, and $f(S \cup \{i\})$ represents the prediction value after adding feature i to the subset S.

2.4 Learning to Rank

Learning to rank (LTR) (Cao, et al., 2007; Liu, 2009) strives to sort a set of objects based on relevance or priority. A typical application of this is when a user inputs a query into a search engine; a series of related documents are returned and presented in descending order of relevance to the query. In fact, the relationship between the documents and the query fed into LTR is one-to-many, as is the association between drugs and their targets. Therefore, LTR is applicable to DTA prediction. The multiple Additive Regression Tree (MART)(Burges, et al., 2005) used in this study can be considered an LTR algorithm with regression properties, as it not only focuses on the relative order of objects but also emphasizes the fit between predicted values and actual values.

3 Experiments and results

3.1 Datasets and evaluation metrics

In this study, numerous experiments are conducted in two different scenarios across two commonly-used datasets. The two scenarios include S1 — Predicting the association between known targets and new drugs with proteins as the queries, and S2 — Predicting the association between known drugs and new targets with drugs as the queries. We consider three evaluation metrics: Concordance index(CI) (Gönen and Heller, 2005), mean squared error (MSE), and r_m^2 (Nguyen, et al., 2021; Öztürk, et al., 2018).

The basic information of the datasets is shown in Table 1:

Table 1. The basic information of the datasets

Dataset	Protein/S1_qid	Drug/S2_qid	Training	testing
Davis	442	68	25046	5010
KIBA	229	2111	98545	19709

Note: S1_qid represents the number of queries in S1. S2_qid represents the number of queries in S2.

3.2 Parameter setting

In this study, we use grid search to determine the optimal parameters for each feature optimization method within specified ranges. Table 2 shows the hyperparameters and the specified search range involved in each method.

Variance threshold (Fida, et al., 2021) is employed to eliminate features with low variance. It operates by calculating the variance of each feature and subsequently removing features with variance below a predetermined threshold. The "threshold" is a user-defined parameter that dictates the criteria for retaining or removing features.

PCA (Wold, et al., 1987) is a widely used unsupervised linear dimensionality reduction method, designed to decrease feature dimensionality while preserving the maximum variance among the features. "n_components" is the main parameter used to determine the number of features, with "mle" signifying the automatic determination of feature dimensions based on the input data. "svd_solver" represents the approach for solving singular value decomposition (SVD), with "full" indicating the employment of the standard SVD method.

Lasso is a regularization method for linear regression that facilitates feature selection and sparse representation of model parameters by adding an L1 regularization term to the loss function (Tibshirani, 1996). The primary parameter, "alpha", signifies the weight of the L1 regularization term. In this study, an appropriate alpha value is determined through 5-fold cross-validation.

Table 2. The hyperparameters and the specified search range involved in each method.

method	Parameter Setting
VarianceThreshold	Threshold: numpy.linspace (0.01,0.1,5)
PCA	n_components:mle, svd_solver:full
Lasso	alpha
XGBoost	n_estimators:[200,400,600,800], max_depth: [3,4,5,6], learning_rate:[0.02,0.04,0.06,0.08]
LightGBM	boosting_type:gbdt, n_estimators:[200,400,600,800], max_depth: [3,4,5,6], learning_rate:[0.02,0.04,0.06,0.08]

CatBoost	iterations:[200,400,600,800], depth:[3,4,5,6], loss_function:RMSE, eval_metric:RMSE, min_data_in_leaf:[2,3,4,5], learning_rate:[0.02,0.04,0.06,0.08]
Ivis	embedding_dims: [128, 256, 512,1024,2048], k: range(10,151,30), n_epochs_without_progress:range(10,21,5), model: maaten, supervision_metric:mse

Note: range(start, stop[, step]) is a function that represents a sequence of numbers starting from 'start' and ending at 'stop' with a step size of 'step'. numpy.linspace (start, stop, num) represents a function that returns "num" evenly spaced numbers between "start" and "stop".

For XGBoost (https://xgboost.readthedocs.io/en/stable/), "learning_rate" controls the contribution of each iteration (i.e., each tree) to the final prediction result. "n_estimators" represents the number of trees, while "max_depth" denotes the maximum depth for each tree.

For LightGBM (https://lightgbm.readthedocs.io/en/latest/index.html), "boosting_type" refers to the boosting type, with "gbdt" being the default selection for this study. The meanings of the other three parameters are consistent with those in XGBoost.

For CatBoost (https://catboost.ai/en/docs/), it is imperative to establish the loss function, with this study electing to employ the Root Mean Square Error (RMSE). "iterations" refers to the number of trees to be constructed. 'eval_metric' serves to delineate the performance metric during the training procedure. "min_data_in_leaf" regulates the minimum number of samples in each leaf node of the decision tree, helping to prevent overfitting by ensuring the decision tree refrains from formulating leaf nodes that comprise an insufficient number of samples.

For Ivis (https://bering-ivis.readthedocs.io/en/latest/index.html), "n_epochs_without_progress" refers to the number of consecutive epochs without progress before training is terminated in the early stopping strategy. "k" represents the number of neighbors and affects the preservation

of local structure in the low-dimensional space. Smaller values of "k" highlight the local structure, while larger values emphasize the global structure. "model" dictates the neural network architecture. "embedding_dims" is the dimension of the processed data. "supervision_metric" determines how to measure the difference between the processed data and the target (label) after dimensionality reduction.

3.3 Performance of various features

We conduct comparative experiments using SAFs and AAFs against two representative types of features to demonstrate their superiority. The methods involved are detailed as follows:

Gen: Models that rely on features extracted from drug molecular descriptors (Cano, et al., 2017) and basic protein sequence information (Bonidia, et al., 2022).

Graph: Models that utilize features obtained via the feature extraction algorithm in GraphDTA (Nguyen, et al., 2021).

It can be observed from Table 3 that our method achieves excellent performance in both scenarios across the two datasets. Regarding accuracy and robustness, our features outperform both the Gen features, which are based on fundamental sequence information, and the Graph features that are learned by neural networks. Such a result implies that AAFs and SAFs are more effective in capturing drug and target information.

Table 3. Performance of various features

	Davis						KIBA					
	S1			S2			S1			S2		
	CI	MSE	r_m^2	CI	MSE	r_m^2	CI	MSE	r_m^2	CI	MSE	r_m^2
Gen	0.886	0.283	0.654	0.875	0.308	0.616	0.839	0.209	0.699	0.836	0.213	0.686
Graph	-	-	-	0.885	0.263	0.672	-		-	0.870	0.163	0.760
Our method	0.937	0.149	0.818	0.931	0.162	0.798	0.895	0.130	0.813	0.893	0.131	0.807

3.4 Performance of various feature processing methods

The performance of various feature processing methods on two scenarios for two datasets are shown in Tables 4-7. "All" represents models built on all features. "VT" represents models built on the feature set after removing low-variance features.

It can be observed from Tables 4 and 5 that low-variance features have a minimal impact on model performance. For Davis dataset, on S1, the model based on all features yields CI, MSE, and r_m^2 values of 0.937, 0.149, and 0.818. On S2, these values are 0.931, 0.162, and 0.798. After removing 54 low-variance features, the model performance decreases by approximately 0.1%, which can be considered negligible. For KIBA dataset, which contains 552 low-variance features. On S1, the model based on VT features shows CI, MSE, and r_m^2 values of 0.891, 0.133, and 0.809. On S2, these values are 0.890, 0.135, and 0.801. The model performance in both scenarios is slightly inferior to that based on all features. However, prioritizing computational speed over model accuracy in the trade-off between the two is evidently a sensible approach.

Table 4. Performance of various feature processing methods for Davis

	Davis(S1)				Davis(S2)			
	CI	MSE	r_m^2	num	CI	MSE	r_m^2	num
All	0.937	0.149	0.818	3208	0.931	0.162	0.798	3208
VT	0.937	0.151	0.816	3154	0.930	0.161	0.799	3154
PCA	0.915	0.182	0.777	620	0.903	0.204	0.746	620
Lasso	0.936	0.150	0.817	293	0.930	0.166	0.793	215
Ivis	0.877	0.307	0.625	128	0.856	0.348	0.566	128

Table 5. Performance of various feature processing methods for KIBA

	KIBA(S1)	KIBA(S2)

	CI	MSE	r_m^2	num	CI	MSE	r_m^2	num
All	0.895	0.130	0.813	3177	0.893	0.131	0.807	3177
VT	0.891	0.133	0.809	2625	0.890	0.135	0.801	2625
PCA	0.827	0.260	0.626	1367	0.822	0.261	0.617	1367
Lasso	0.891	0.134	0.806	557	0.888	0.138	0.797	325
ivis	0.754	0.420	0.395	128	0.748	0.429	0.369	128

Table 6. Performance of regression tree-based feature selection methods for Davis

		Davis(S1)				Davis(S2)			
		CI	MSE	r_m^2	num	CI	MSE	r_m^2	num
XGB	SFM	0.938	0.150	0.816	1401	0.931	0.161	0.799	2235
	SHAP	0.938	0.150	0.816	1401	0.931	0.161	0.799	2235
Light	SFM	0.937	0.148	0.819	2108	0.931	0.161	0.800	2212
	SHAP	0.937	0.148	0.819	2108	0.931	0.161	0.800	2212
Cat	SFM	0.937	0.150	0.816	1278	0.930	0.162	0.798	1351
	SHAP	0.937	0.150	0.816	1278	0.930	0.162	0.798	1351

Table 7. Performance of regression tree-based feature selection methods for KIBA

		KIBA(S1)				KIBA(S2)			
		CI	MSE	r_m^2	num	CI	MSE	r_m^2	num
XGB	SFM	0.891	0.133	0.809	2044	0.890	0.135	0.801	2256
	SHAP	0.891	0.133	0.809	2044	0.890	0.135	0.801	2256
Light	SFM	0.891	0.132	0.810	2127	0.890	0.135	0.801	1935
	SHAP	0.891	0.132	0.809	2127	0.890	0.135	0.801	1935
Cat	SFM	0.891	0.133	0.809	1105	0.890	0.135	0.802	1163
	SHAP	0.891	0.133	0.809	1105	0.890	0.135	0.802	1163

Other feature processing methods optimize the feature set based on VT. As evidenced in Tables 4 and 5, the performance of the Ivis-based models is notably suboptimal, exhibiting limited robustness. This is particularly evident in both scenarios for KIBA dataset, where the model's CI value is only around 75%, and r_m^2 falls below 0.5. The performance of the PCA-based model is significantly inferior to those built based on all features. Feature optimization techniques that rely on regression concepts exhibit superior performance. The LASSO-based model which only uses 293D features can achieve CI and MSE values almost equivalent to those built on all features. It can be concluded from Tables 6 and 7 that the models utilizing the three feature optimization methods based on regression trees yield promising results, showing superior performance with low dimensions. Additionally, the robustness of these models is evidenced by r_m^2 values.

3.5 Performance of important features

As observed in Tables 4-7, the three feature selection methods—XGBoost, LightGBM, and CatBoost—all based on regression trees, show distinct advantages. To enhance the interpretability of the models, this study conducts a comprehensive analysis of the features selected by these three methods.

In addition to calculating the feature importance using the methods embedded in these three methods, this study ranks features according to their SHAP values. It is revealed that in Tables 6 and 7, the two feature importance calculation methods produce almost identical results, as they construct models with equivalent performance based on optimal feature subsets with the same dimensions.

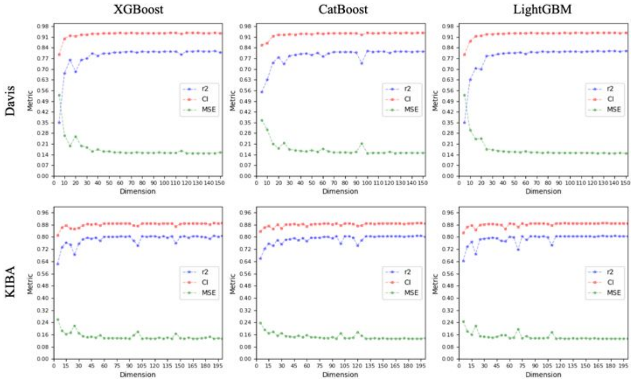

Fig.2. Performance of the model under each feature dimension on S1

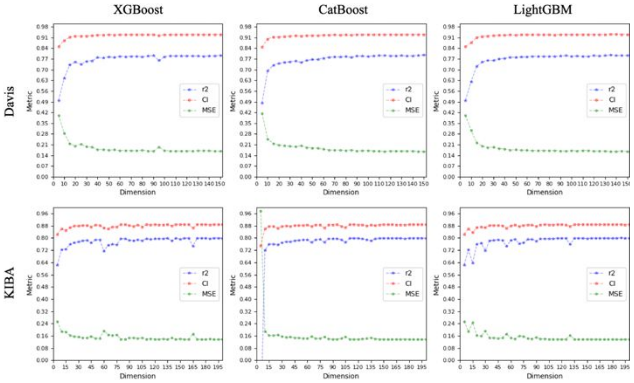

Fig.3. Performance of the model under each feature dimension on S2

In this study, we obtain the optimal feature set by examining their SHAP values and using the incremental feature selection (IFS). Fig.2. and Fig.3. illustrate the feature increment process with a step size of 5 on both scenarios for the two datasets, it is evident that the models' performances tend to be stable when the feature dimension reaches 150. In other words, models built on the top 150 features perform almost as well as those built on all features, yet their dimensions constitute only one-twentieth of the latter. On both scenarios for KIBA dataset, the models using the three methods have CI values of approximately 88.7%, MSE values below 14%, and r_m^2 around 79%. On both scenarios for Davis dataset, the CI, MSE, and r_m^2 values for the three methods are roughly 93%, 16%, and 80%, respectively.

3.6 Analysis of important features

As depicted in Fig. 2 and Fig. 3, the first 20 features significantly impact model performance. The top 20 features ranked by the three feature optimization methods are displayed in Fig. 4 and Fig. 5.

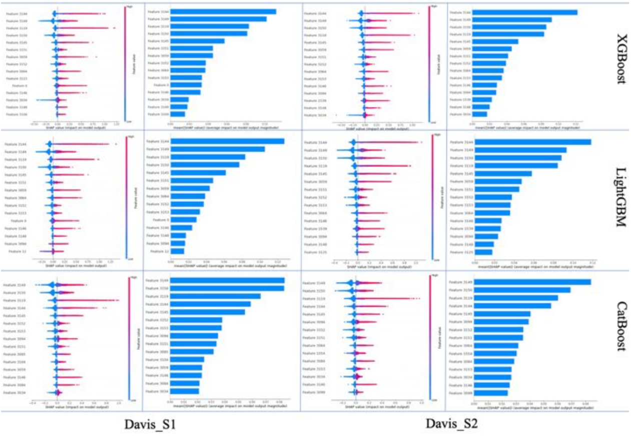

Fig.4. The top 20 features of regression tree-based feature selection methods for Davis

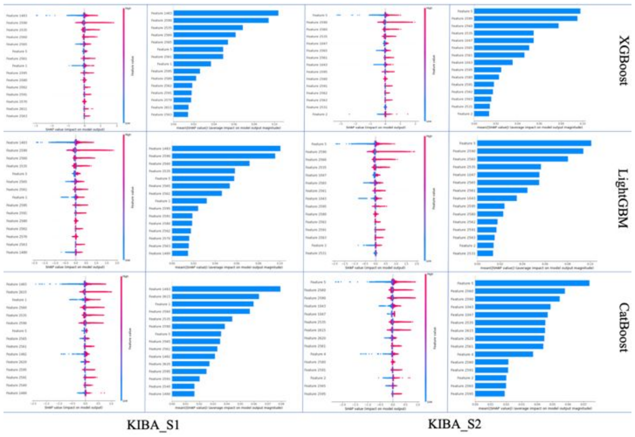

Fig.5. The top 20 features of regression tree-based feature selection methods for KIBA

For Davis, 15 common features appear among the top 20 ranked features in both scenarios S1 and S2, as determined by the three methods. Of these 15 features, 14 are AAFs—with 7 AAFs based on similarity and 7 AAFs based on sharing. The remaining feature is a SAF, which is derived from the drug-target affinity information of the target itself.

For KIBA, on S1, 14 common features among the top 20 ranked features are shared by the three regression algorithms. On S2, they share 15 common features. It's evident that similarity-based AAFs play a crucial role in both scenarios. On S1, 9 of the top 14 important features are AAFs, 8 of which are similarity-based. On S2, 10 of the top 15 important features are AAFs, 7 of which are similarity-based. Additionally, it's noteworthy that both scenarios share five SAFs derived from drug-target affinity information. This suggests that SAFs also demonstrate competitiveness within the KIBA dataset.

As a result, it can be concluded that different types of features exhibit varying performance under distinct circumstances. Although SAFs may not exhibit comparable efficiency as AAFs, it's undeniable that they share a complementary relationship, and each contributes positively to the overall performance of the model.

Furthermore, we can also observe the following: i. Some SAFs, such as the mean and quantile values of similarity between proteins/drugs, have little or even no impact on model performance. ii. During SAFs extraction, overlapping neighbors for objects are obtained based on a set of similarity and sharing thresholds, which result in repetitive or redundant proximity information. iii. When identifying additional neighbors that are most similar to the top 5 most similar neighbors, overlapping and intersecting information also occur. In summary, it is inevitable that the initially acquired features contain redundant and duplicate information. Consequently, feature optimization is a necessary process to build models with fast calculation speed, strong generalization ability, and high interpretability while maintaining consistent performance.

4 Conclusion

In this study, our main contribution lies in applying various feature processing techniques to optimize the extracted AAFs and SAFs, and then build a DTA prediction model with high accuracy and robust interpretability.

Extensive experimental results indicate that the three GBDT variant methods—Xgboost, LightGBM, and CatBoost—all exhibit superior feature selection capabilities. Furthermore, we employ SHAP values and IFS to rank significant features and determine the optimal feature subset. Models built on the optimal feature subset perform almost as well as those built on all features. In conclusion, feature optimization is crucial for building models that offer strong generalization ability and high interpretability while maintaining consistent performance.

There are still several avenues for enhancing DTA prediction methods, particularly with regard to feature engineering. For instance, one could leverage additional relationships to obtain more comprehensive feature information about drugs or targets from various perspectives, such as through protein-protein interactions or drug-disease associations. Additionally, most existing studies simply concatenate the features of drugs and targets. This approach may fail to fully explore and reflect deeper, underlying information and could lead to the curse of dimensionality. Therefore, deriving features from a range of perspectives and developing advanced feature selection or feature fusion methods will be central to our future endeavors.

Funding

This work has been supported by the National Natural Science Foundation of China (No. 62131004, No.62250028, No.62101094), and the Municipal Government of Quzhou (No. 2022D040).

Conflict of Interest: none declared.

References

Al Daoud, E. Comparison between XGBoost, LightGBM and CatBoost using a home credit dataset. *International Journal of Computer and Information Engineering* 2019;13(1):6-10.

Bonidia, R.P., *et al.* MathFeature: feature extraction package for DNA, RNA and protein sequences based on mathematical descriptors. *Briefings in bioinformatics* 2022;23(1):bbab434.

Burges, C., *et al.* Learning to rank using gradient descent. In, *Proceedings of the 22nd international conference on Machine learning*. 2005. p. 89-96.

Cano, G., *et al.* Automatic selection of molecular descriptors using random forest: Application to drug discovery. *Expert Systems with Applications* 2017;72:151-159.

Cao, Z., *et al.* Learning to rank: from pairwise approach to listwise approach. In, *Proceedings of the 24th international conference on Machine learning*. 2007. p. 129-136.

Wang, C. and Zou, Q. A Machine Learning Method for Differentiating and Predicting Human-Infective Coronavirus Based on Physicochemical Features and Composition of the Spike Protein. *Chinese Journal of Electronics* 2021;30(5):815-823.

Chen, R., *et al.* Machine learning for drug-target interaction prediction. *Molecules* 2018;23(9):2208.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In, *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016. p. 785-794.

Chen, X., *et al.* Drug–target interaction prediction: databases, web servers and computational models. *Briefings in bioinformatics* 2016;17(4):696-712.

Chen, Y., *et al.* Deep generative model for drug design from protein target sequence. *Journal of Cheminformatics* 2023;15(1):38.

Chen, Y., *et al.* Molecular language models: RNNs or transformer? *Briefings in Functional Genomics* 2023:elad012.

Fida, M.A.F.A., Ahmad, T. and Ntahobari, M. Variance Threshold as Early Screening to Boruta Feature Selection for Intrusion Detection System. In, *2021 13th International Conference on Information & Communication Technology and System (ICTS)*. IEEE; 2021. p. 46-50.

Gönen, M. and Heller, G. Concordance probability and discriminatory power in proportional hazards regression. *Biometrika* 2005;92(4):965-970.

Hancock, J.T. and Khoshgoftaar, T.M. CatBoost for big data: an interdisciplinary review. *Journal of big data* 2020;7(1):1-45.

Huang, K., *et al.* DeepPurpose: a deep learning library for drug–target interaction prediction. *Bioinformatics* 2020;36(22-23):5545-5547.

Ke, G., *et al.* Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 2017;30.

Klambauer, G., *et al.* Self-normalizing neural networks. *Advances in neural information processing systems* 2017;30.

Liang, W., *et al.* Predicting hard rock pillar stability using GBDT, XGBoost, and LightGBM algorithms. *Mathematics* 2020;8(5):765.

Liu, T.-Y. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 2009;3(3):225-331.

Lundberg, S.M. and Lee, S.-I. A unified approach to interpreting model predictions. *Advances in neural information processing systems* 2017;30.

Luo, Y., *et al.* A network integration approach for drug-target interaction prediction and computational drug repositioning from heterogeneous information. *Nature communications* 2017;8(1):573.

Mei, J.-P., *et al.* Drug–target interaction prediction by learning from local information and neighbors. *Bioinformatics* 2013;29(2):238-245.

Nguyen, T., *et al.* GraphDTA: predicting drug–target binding affinity with graph neural networks. *Bioinformatics* 2021;37(8):1140-1147.

Öztürk, H., Özgür, A. and Ozkirimli, E. DeepDTA: deep drug–target binding affinity prediction. *Bioinformatics* 2018;34(17):i821-i829.

Pahikkala, T., *et al.* Toward more realistic drug–target interaction predictions. *Briefings in bioinformatics* 2015;16(2):325-337.

Prokhorenkova, L., *et al.* CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems* 2018;31.

Rao, H., *et al.* Feature selection based on artificial bee colony and gradient boosting decision tree. *Applied Soft Computing* 2019;74:634-642.

Ru, X., *et al.* NerLTR-DTA: drug–target binding affinity prediction based on neighbor relationship and learning to rank. *Bioinformatics* 2022;38(7):1964-1971.

Tian, H. and Tao, P. ivis Dimensionality Reduction Framework for Biomacromolecular Simulations. *Journal of chemical information and modeling* 2020;60(10):4569-4581.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)* 1996;58(1):267-288.

Wold, S., Esbensen, K. and Geladi, P. Principal component analysis. *Chemometrics and intelligent laboratory systems* 1987;2(1-3):37-52.

Yamanishi, Y., *et al.* Drug-target interaction prediction from chemical, genomic and pharmacological data in an integrated framework. *Bioinformatics* 2010;26(12):i246-i254.

Ye, J., *et al.* Stochastic gradient boosted distributed decision trees. In, *Proceedings of the 18th ACM conference on Information and knowledge management*. 2009. p. 2061-2064.

Yuan, W., Chen, G. and Chen, C.Y.-C. FusionDTA: attention-based feature polymerizer and knowledge distillation for drug-target binding affinity prediction. *Briefings in Bioinformatics* 2022;23(1):bbab506.

Zeng, Y., *et al.* Deep drug-target binding affinity prediction with multiple attention blocks. *Briefings in bioinformatics* 2021;22(5):bbab117.

Zhang, Z. Introduction to machine learning: k-nearest neighbors. *Annals of translational medicine* 2016;4(11).