Missing Value Imputation using XGboost for Label-Free Mass Spectrometry-Based Proteomics Data

Jian Song^{1,2,3} and Changbin Yu^{2,3}

¹ Zhejiang University, Hangzhou, Zhejiang Province, China

² School of Engineering, Westlake University, 18 Shilongshan Road, Hangzhou 310024, Zhejiang Province, China

³ Institute of Advanced Technology, Westlake Institute for Advanced Study, 18 Shilongshan Road, Hangzhou 310024, Zhejiang Province, China
{yuchangbin}@westlake.edu.cn

Abstract. The label-free mass spectrometry-based proteomics data inevitably suffer from the problem of missing values. The existence of missing values prevents the downstream analyses which need a complete data matrix. Our motivation is to introduce the state-of-art machine learning algorithm XGboost to realize a method of imputation which can improve the accuracy of imputation. But in practical, XGboost has many parameters need to be tuned to deliver on its potential high performance. Although cross validation may find the best parameters, it is much time-consuming. Alternatively, we empirically determined the parameters to two kinds of base learners of XGboost. To explore the robustness and performance of XGboost based imputation with predetermined parameters, we conducted tests on three benchmark datasets. As a comparative, six common imputation methods were also experimented in terms of normalized root mean squared error and Pearson correlation coefficient. The comparative experimental results indicated that the XGboost based imputation method using the linear base learner is competitive to or outperforms its competitors, including the random forest based imputation, by achieving smaller imputation errors and better structure preservation under the empirical parameters for the three benchmark datasets.

Keywords: Imputation algorithm · XGboost · Proteomics.

1 Introduction

The label-free mass spectrometry-based (MS) quantitative method has become a major means of proteomics. But the proteome data matrix acquired by the label-free MS method may suffer from missing, accounting for biochemical and analytical to bioinformatics mechanisms. The missing rate frequently ranges between 10% to 50% [1]. Unfortunately, the downstream analyses of proteomics data such as clustering and supervised learning need complete data without missing values. Of course, one solution to the missing data problem is to repeat the

experiment. But this strategy is expensive and time-consuming. Another option of ignoring missing values would dramatically reduce the size and completeness of the data and limit the researchers' ability to infer information from the data. Thus, one of the major challenges of proteomics data studies is to impute this missing data appropriately.

2 Related works

Over the past decade, a variety of imputation algorithms have been developed and subsequently discussed in the literature [2]. For the sake of illustration, we assume $\mathbf{O}_{m \times n}$ to be a $m \times n$ -dimensional proteomics data matrix. Each row of $\mathbf{O}_{m \times n}$ is one observation represented by $\mathbf{o}_{1 \times n}$, each column of $\mathbf{O}_{m \times n}$ means one protein variable denoted by $\mathbf{v}_{n \times 1}$, and $\mathbf{v}_{i,j}$ means the i -th observation in the j -th variable. Firstly, [3] published the first two missing value imputation algorithms based on the k -nearest neighbours (kNNImpute) and singular value decomposition (SVDImpute). The kNNImpute searches the most similarity observations and calculates the weighted sum as the imputation value. Mathematically,

$$\mathbf{o}^* = x_1 \mathbf{o}_1 + x_2 \mathbf{o}_2 + \cdots + x_k \mathbf{o}_k \quad (1)$$

where $\mathbf{o}^*$ is the targeted observation needed to be imputed, k ($1 \leq k \leq n$) is the selected observations' numbers that mostly similarity to $\mathbf{o}^*$. In most cases the k is the hyper-parameter needed to be determined by experiments. $x_i, i \in [1, k]$ is the corresponding coefficient between the selected observation with the target observation. The weight x_i is calculated as follow equation:

$$x_i = \frac{1/d_i}{\sum_{i=1}^k 1/d_i} \quad (2)$$

where $d_i = \|\mathbf{o}_i - \mathbf{o}^*\|_2$ is the Euclidean distance between $\mathbf{o}_i$ and $\mathbf{o}^*$. The SVDImpute first imputes the missing values with mean of variables to obtain a complete matrices, then repetitively decomposes the matrices and ignores the small singular value term until the total change in the matrix falls below the determined threshold. In math,

$$\begin{aligned} (\mathbf{O}_{m \times n})_{\text{mean imputed}} &= \mathbf{U}_{m \times m} \mathbf{\Sigma}_{m \times n} \mathbf{V}_{n \times n}^T \\ &\simeq \sigma_1 \mathbf{O}_1 + \sigma_2 \mathbf{O}_2 + \cdots + \sigma_k \mathbf{O}_k \end{aligned} \quad (3)$$

where $\mathbf{U}$, $\mathbf{V}$ is orthogonal matrix, and $\mathbf{\Sigma}$ is the diagonal matrix of singular values. Selecting the first k maximum singular values ($\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k$) and calculating the sum can restore the raw matrix approximately.

[4] proposed a new missing value estimation method based on Bayesian principal component analysis (BPCAImpute). BPCAImpute consists of three elementary processes: (1) principal component regression, (2) Bayesian estimation, and (3) an expectation-maximization repetitive algorithm. BPCAImpute is a

non-parameter method and can provide accurate and convenient estimation for missing values. Further, [5] and [6] introduced the least squares regression to imputation (LSImpute and LLSImpute). To simplify the description, we assume only the first element, i.e. $o_{1,1}$ is missing, devoted by η . Therefore, the $\mathbf{O}$ can be divided into four parts:

$$\mathbf{O} = \begin{bmatrix} \eta & \mathbf{b}^T \\ \mathbf{w} & \mathbf{A} \end{bmatrix} \quad (4)$$

where the $\mathbf{w}$ and $\mathbf{b}^T$ are the first column and row removed the $o_{1,1}$ respectively and $\mathbf{A}$ is the remaining matrix removed the first row and column. Constructing the least squares regression model as

$$\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{w}\|_2 \quad (5)$$

This is a well-know regression problem, and the coefficient vector $\mathbf{x}$ can be worked out by

$$\mathbf{x} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{w} \quad (6)$$

Then, the missing value η is estimated as

$$\eta = \mathbf{b}^T \mathbf{x} \quad (7)$$

Same with this linear regression way, [7] and [8] trained the linear regression model equation (5) with regularization terms:

$$\min_{\mathbf{x}} \{\|\mathbf{Ax} - \mathbf{w}\|_2 + \alpha \|\mathbf{x}\|_1 + \lambda \|\mathbf{x}\|_2\} \quad (8)$$

where α is the L_1 regularized term coefficient, and λ is the L_2 regularized term coefficient. [8] found that only reserving the λ can have better imputation result on test datasets. In this case, the η is given in

$$\eta = ((\mathbf{A}^T \mathbf{A} - \lambda \mathbf{I})^+ \mathbf{A}^T \mathbf{w}) \mathbf{x} \quad (9)$$

where the $\mathbf{I}$ is an identity matrix, and $()^+$ means to calculate the pseudo-inverse. After imputing the $o_{1,1}$, other columns within missing values are estimated in the same process.

At present, ensemble learning categorized into bagging and boosting has shown its advantages in various fields for the application of classification and regression problem. Random forest is the typical of bagging method. [9] employed the random forest to model the $\mathbf{A}$ and $\mathbf{w}$. Based the trained model, the η can be predicted by $\mathbf{b}$ (RFImpute). According to [10] report, the random forest based imputation can achieve the minimum imputation error compared to other imputation methods. For the boosting method, typically gradient boosting machine (GBM) was firstly presented by [11]. and evolved to XGBoost [12]. In brief, the XGboost predicts the result with the sum of tree ensemble models:

$$\hat{\mathbf{w}} = \sum_{k=1}^K f_k(\mathbf{A}), f_k \in \mathcal{F} \quad (10)$$

here K is the regression tree numbers and $\mathcal{F}$ is the space of regression tree, $f_k(\mathbf{A})$ is the result of the k th tree. Then, we minimize the following regularized objective

$$\mathcal{L}(\phi) = l(\hat{\mathbf{w}}, \mathbf{w}) + \sum_k \Omega(f_k) \quad (11)$$

where the l denotes a differentiable convex loss function, $\Omega(f)$ is a regularization term that can penalize the model complexity. However, the tree ensemble model contains functions as parameters is unlikely to be optimized in Euclidean space when adapting traditional optimization methods. Thus, the model is trained as an effective manner that we can add f_t in the loss function. In this case, $\hat{\mathbf{w}}^{(t)}$ will become the prediction of the t th iteration. Hence, the regularized objective can be approximately described to the following formula of iteration t

$$\mathcal{L}^t \simeq \sum_{i=1}^{m-1} [l(\mathbf{w}_i, \hat{\mathbf{w}}^{(t-1)}) + g_i f_t(\mathbf{a}_i) + \frac{1}{2} h_i f_t^2(\mathbf{a}_i)] + \Omega(f_t) \quad (12)$$

where $\mathbf{a}_i$ is one row of $\mathbf{A}$, $f_t(\mathbf{a}_i)$ is the increment. $g_i = \partial_{\hat{\mathbf{w}}^{(t-1)}} l(\mathbf{w}_i, \hat{\mathbf{w}}^{(t-1)})$ and $h_i = \partial_{\hat{\mathbf{w}}^{(t-1)}}^2 l(\mathbf{w}_i, \hat{\mathbf{w}}^{(t-1)})$ are the first and second order gradient statistics in the loss function. The model in XGboost is trained till stop criterion such as early stopping rounds or the number of boosting iterations t is satisfied. Besides the regression tree can be the base learner, the XGboost is also using the linear regression as a choice of base learner.

In this study, we modelled the $\mathbf{A}$ and $\mathbf{w}$ using XGboost, the gradient boost algorithm to improve the imputation accuracy for label-free mass spectrometry-based proteomics data. Meanwhile, we compared our method with meanImpute (imputation using constant mean value), kNNImpute, SVDImpute, BPCAImp, LLSImpute and RFImpute. Comparisons were performed on three benchmark datasets coming from different papers and using different proportions of missing values.

3 Approach

3.1 Methods

The proteomics data are displayed as a matrix with rows and columns representing observations/samples and variables/proteins, respectively. Throughout the paper, we will use $\mathbf{O}_{m \times n}$ to denote the data matrix with m observations and n variables. First, let's look at the case that only one column has the missing values. For instance, the first column is including missing values. Then we can divide the matrix into four parts as Fig.1 shows:

1. the missing values vector of first column denoted as $\mathbf{x}$, and this is what we want to predict
2. the variables other than the first column with the same row index as $\mathbf{x}$ denoted by $\mathbf{B}$

Fig. 1. The illustration of data matrix division when only the first column need to be imputed.

3. the vector of first column excluding $\mathbf{x}$ denoted as $\mathbf{b}$
4. the remaining of matrix denoted by $\mathbf{A}$

After the definition of the matrix, the missing values $\mathbf{x}$ can be estimated through two phases: train and prediction. First, we use the matrix $\mathbf{A}$ and vector $\mathbf{b}$ to train the XGboost model, then, we feed the trained model with new data $\mathbf{B}$ to obtain the final estimated value.

For the case of missing values within multiple columns, we first need to make an initial guess for the missing values in $\mathbf{O}$ using mean imputation or another imputation method. Then, sorting the variables $\mathbf{v}_i, i = 1, \dots, n$ according to the amount of missing values and starting imputation with the lowest amount missing values variable. For each variable $\mathbf{v}_i$, the missing values are imputed by the above process. After all the variables are imputed once, the imputation procedure is repeated until a stopping criterion is met. The pseudo Algorithm 1 exhibits a representation of the XGboost (xgbImpute) based imputation method inspired by [9].

The stopping criterion γ is met when the repetition times reaches a predetermined value or as soon as the difference between the newly imputed data matrix and the previous one increases for the first time. In that case, the γ is defined as

$$\gamma = \frac{\|\mathbf{O}_{new} - \mathbf{O}_{old}\|_2}{\|\mathbf{O}_{old}\|_2} \quad (13)$$

3.2 Evaluation Metrics

To evaluate the effectiveness of xgbImpute, normalized root mean squared error (NRMSE) [3] and Pearson correlation coefficient (PCC) [8] are used. The normalized root mean squared error measures the overall deviation of estimated values from their corresponding true value as defined below:

$$\text{NRMSE} = \frac{\sqrt{\text{mean}[(\mathbf{y}_{predicted} - \mathbf{y}_{true})^2]}}{\text{std}(\mathbf{y}_{true})} \quad (14)$$

Algorithm 1 Impute missing values with XGboost

Require: $\mathbf{O}$ an $m \times n$ matrix with missing values, stopping criterion γ

```

1:  $\mathbf{k} \leftarrow$  vector of sorted indices of columns in  $\mathbf{O}$ 
   w.r.t. increasing amount of missing values;
2: Mean imputation for missing values;
3: while not  $\gamma$  do
4:    $\mathbf{O}_{old} \leftarrow$  store previously imputed matrix;
5:   for  $s$  in  $\mathbf{k}$  do
6:     Separate the matrix into:  $\mathbf{x}, \mathbf{B}, \mathbf{b}$  and  $\mathbf{A}$  based on  $s$ th column;
7:     Train a XGBoost model:  $\mathbf{b} \sim \mathbf{A}$ ;
8:     Predict  $\mathbf{x}$  by feeding  $\mathbf{B}$  into the trained model;
9:      $\mathbf{O}_{new} \leftarrow$  update  $s$ th column using predicted  $\mathbf{x}$ ;
10:  end for
11:  update  $\gamma$ ;
12: end while
13: return the imputed matrix  $\mathbf{O}_{new}$ 

```

where $\mathbf{y}_{true}$ is the vector of true value but is artificially missing, and $\mathbf{y}_{predicted}$ is the vector of the predicted value correspond to $\mathbf{y}_{true}$. NRMSE is more near to zero, the imputation is more accurate. Besides, Pearson correlation coefficient (PCC) is also able to quantify the consistence between the predicted and the raw data:

$$\text{PCC} = \frac{\text{cov}(\mathbf{y}_{predicted}, \mathbf{y}_{true})}{\text{std}(\mathbf{y}_{predicted})\text{std}(\mathbf{y}_{true})} \quad (15)$$

PCC is more near to 1, the imputation gets better accordance with actual data. These two metrics above reach a comprehensive comparison between xgbImpute and its competitors.

4 Experiments

4.1 Datasets

- Kinases expression of human colon and rectal cancer cell line (CRC): the authors used LC-MS/MS-based shotgun proteomics to measure the proteomes of 65 CRC lines and to quantify their expressed kinomes using Kinobeads ([13], [14]). This led to the identification of a total of 235 human kinases (median/cell line = 155, from Kinobeads experiments), see [15]. The missing rate of this dataset is about 34%.
- Proteome about the interstitial lung disease (ILD): This dataset has been previously described in detail by [16]. From each sample in 11 human lung tissue biopsies, the proteins were extracted with increasing stringency into four fractions by changing the detergent and buffer conditions. We select the result of first fraction and randomly sample 500 proteins without missing values.

Table 1. Competition Methods Description

Method	Available R package	Parameters
meanImpute	imputation	none
kNNImpute	imputation	k nearest neighbors
SVDImpute	imputation	number of components
BPCAImpu	pcaMethods	number of principal components
LLSImpute	pcaMethods	number of similar variables
RFImpute	missForest	numbers of trees and variables randomly sampled at each split

- Ovarian cancer proteome dataset (Ovarian): The authors performed quantitative proteomics on 25 ovarian tumor samples. Adapting label-free proteomic workflow they identified and quantified more than 9,000 proteins from low-mg archival samples in single-run measurements in the MaxQuant environment. More details see [17]. Randomly we select 400 proteins to construct the complete data matrix.

For the ILD and Ovarian dataset, we randomly introduced missing values to the complete dataset with missing rates of 10% to 50% stepped with 10%. Meanwhile, we saved the true value to evaluate the imputation accuracy later. For the CRC datasets, accounting to the original missing rate, we introduce missing values with missing rates from 35% to 50% stepped with 5%. Due to the missing mechanism is random, we performed 10 times imputation experiments for every missing rates.

4.2 Parameters

We compared xgbImpute with meanImpute, kNNImpute, SVDImpute, BPCAImpu, LLSImpute and RFImpute. The detail information of these methods was shown in Table 1. The parameters of k nearest neighbors in KNNImpute, the number of components in SVDImpute and the number of similar variables in LLSImpute are hyper parameters and we tried all possible parameters in $[1, m - 1]$ under the first missing rate of each dataset and selected the parameter in the best performance with the metrics of NRMSE.

The parameter of BPCAImpu is the number of principal components which was set to the numbers of rows subtracted one as recommended in [4]. The parameters of RFImpute include the numbers of trees and variables randomly sampled at each split for both of which we used the default values.

The parameters of xgbImpute are inherited from XGboost. When the base learner is regression tree, i.e. `booster= 'gbtree'`, the parameters mainly including:

- `nrounds` max number of boosting iterations.
- `eta` control the learning rate. It is used to prevent overfitting by making the boosting process more conservative.

8 xgbImpute

- **gamma** minimum loss reduction required to make a further partition on a leaf node of the tree.
- **max_depth** maximum depth of a tree.
- **min_child_weight** minimum sum of instance weight (hessian) needed in a child.
- **subsample** subsample ratio of the training instance.
- **colsample_bytree** subsample ratio of columns when constructing each tree.

When the base learner of XGboost is linear model, i.e. **booster** = 'gblinear', the parameters mainly including:

- **nrounds** max number of boosting iterations.
- **lambda** L2 regularization term on weights.
- **lambda_bias** L2 regularization term on bias.
- **alpha** L1 regularization term on weights.

In order to optimize parameters comprehensively, we consulted some references ([12], [18], [19]) and determined by grid search with cross validation. Ultimately, the results we used are : **nrounds** = 100, **eta** = 0.15, **gamma** = 0, **max_depth** = 6, **min_child_weight** = 1, **subsample** = 0.8, **colsample_bytree** = 0.8 and parameters for the 'gblinear' booster are: **nrounds** = 100, **lambda** = 10, **lambda_bias** = 0 and **alpha** = 0.

5 Results

5.1 normalized root mean squared error

In this study, for each dataset, we conducted experiments ten times for each imputation method and each missing rate and calculated the average normalized root mean squared error(the xgbImpute including two methods based on the base learner: gbtreeImpute and gblinearImpute). As shown in Fig. 2, we can observe that NRMSE increases with the increasing of missing rates for all imputation methods approximately. This is reasonable because a larger missing rate comes with the loss of more information. Besides the average NRMSE, we also performed a unpaired Wilcoxon test of the NRMSE of the compared methods versus the NRMSE of gblinearImpute which have the best performance in most datasets at each missing rate.

Combining the average NRMSE result and significance result of CRC dataset in Table 2, we can see that the RFImpute have the minimum average NRMSE at each missing rate. Meanwhile, none of the significance between RFImpute and gblinearImpute results are significant as Table 2 shows. That is to say, the performance of RFImpute and gblinearImpute for the CRC dataset are close to each other. In addition, the gbtreeImpute have the moderate performance compared to other methods. In the case of 45%, 50%, 55% missing rates, the LLSImpute algorithm produced huge NRMSE. To better display of other valid value, we don't show these outliers.

Fig. 2. NRMSE vs. different missing rate on the three datasets.

For the ILD dataset, result shown in Fig. 2 (ILD) and Table 3, we can observe that the gblinearImpute have the best performance at each missing rate. Oppositely, the gbtreeImpute is not work better than other methods. Considering the significance result, we found kNNImpute, LLSImpute and RFImpute have no significance at missing rate 10% to 30% compared to gblinearImpute. But when the missing rate is increasing to 40% and 50%, the gblinearImpute is better than all other competitors significantly. Also, the kNNImpute and LLSImpute produce outliers of imputation result, and the RFImpute have a enormous NRMSE at missing rate 40%.

For the Ovarian dataset, result shown in Fig.2 Ovarian and Table 4, we can see that the gblinearImpute is better than other methods significantly at each missing rate except the RFImpute at missing rate of 10% and 40%. Meanwhile, the gbtreeImpute has poor performance at missing rate 40% to 50% compared to

10 xgbImpute

Fig. 3. Pearson correlation coefficient vs. different missing rate on the three datasets.

other methods. In the case of 40%, 55% missing rates, the knnImpute algorithm produce huge NRMSE compared to other methods which are not shown.

Looking at the metric of NRMSE, the gblinearImpute with specify parameters is mostly better than other competitors at each missing rate. Considering the robust, the gblinearImpute is also consistently outperforms its competitors. For the specify parameters gbtlineImpute, it doesn't show its advantages.

5.2 Pearson correlation coefficient

To show the effectiveness of different imputation methods in preserving the original data structure, we also calculated the PCC based on the true value and the predicted value. Fig.3 presents corresponding experimental results for different methods about the PCC values. Combining Fig.3 with Fig.2, we can observe that the average PCC trend is opposite to the average NRMSE trend, i.e. PCC decreases with the increasing of missing rates for all imputation methods approximately. In particular, we found that only in the case of 35% missing rate of CRC dataset, the performance of meanImpute and kNNImpute are inconsistent under the metrics of NRMSE and PCC. That is to say, in the view of NRMSE, the meanImpute is superior, but in the view of PCC, it is inferior. Except this case, we don't find that the NRMSE is outperformance but the PCC is underperformance. Thus, looking at the metric of PCC, the conclusion is basically the same with the metric of NRMSE, i.e. for CRC dataset, the gblinearImpute and rfImpute is outperformance, for ILD and Ovarian dataset, the gblinearImpute is more accuracy and robust than other competitors.

6 Conclusion

Missing value imputation is a challenging for proteomics research. With the development of many kinds of imputation methods, it is more and more difficult to improve the accuracy and robustness of imputation method. In this paper, we applied the state-of-art machine learning algorithm XGboost to realize the imputation of label-free mass-spectrometry based proteomics data. Theoretically, when training the XGboost model every time for each missing column we can use cross validation to get the best parameters, but this will lead to much

time-consuming. Alternatively, we empirically determined the parameters for the two kind base learners of XGboost. Our preliminary experimental results show on three real label-free mass-spectrometry based proteomics datasets that the allocated parameters of XGboost with linear base learner for imputation is competitive to or outperforms six established imputation methods like the random forest based imputation.

For subsequent analysis, we plan to analysis every sample predict error other than the whole error to find the cause of relative big mistake such as the distribution of missing values. Furthermore, we will explore the XGboost based imputation can be applied to other fields that also suffer from the problem of missing values such as DNA microarray or metabolic data.

References

1. Cosmin Lazar, Laurent Gatto, Myriam Ferro, Christophe Bruley, and Thomas Burger. Accounting for the multiple natures of missing values in label-free quantitative proteomics data sets to compare imputation strategies. *Journal of proteome research*, 15(4):1116–1125, 2016.
2. Bobbie-Jo M Webb-Robertson, Holli K Wiberg, Melissa M Matzke, Joseph N Brown, Jing Wang, Jason E McDermott, Richard D Smith, Karin D Rodland, Thomas O Metz, Joel G Pounds, et al. Review, evaluation, and discussion of the challenges of missing value imputation for mass spectrometry-based label-free global proteomics. *Journal of proteome research*, 14(5):1993–2001, 2015.
3. Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.
4. Shigeyuki Oba, Masa-aki Sato, Ichiro Takemasa, Morito Monden, Ken-ichi Matsubara, and Shin Ishii. A bayesian missing value estimation method for gene expression profile data. *Bioinformatics*, 19(16):2088–2096, 2003.
5. Trond Hellem Bø, Bjarte Dysvik, and Inge Jonassen. Lsimpute: accurate estimation of missing values in microarray data with least squares methods. *Nucleic acids research*, 32(3):e34–e34, 2004.
6. Hyunsoo Kim, Gene H Golub, and Haesun Park. Missing value estimation for dna microarray gene expression data: local least squares imputation. *Bioinformatics*, 21(2):187–198, 2005.
7. Tapio Schneider. Analysis of incomplete climate data: Estimation of mean values and covariance matrices and imputation of missing values. *Journal of climate*, 14(5):853–871, 2001.
8. Aiguo Wang, Ye Chen, Ning An, Jing Yang, Lian Li, and Lili Jiang. Microarray missing value imputation: A regularized local learning method. *IEEE/ACM transactions on computational biology and bioinformatics*, 16(3):980–993, 2018.
9. Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.
10. Runmin Wei, Jingye Wang, Mingming Su, Erik Jia, Shaoqiu Chen, Tianlu Chen, and Yan Ni. Missing value imputation approach for mass spectrometry-based metabolomics data. *Scientific reports*, 8(1):1–10, 2018.
11. Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

12. Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.
13. Marcus Bantscheff, Dirk Eberhard, Yann Abraham, Sonja Bastuck, Markus Boesche, Scott Hobson, Toby Mathieson, Jessica Perrin, Manfred Raida, Christina Rau, et al. Quantitative chemical proteomics reveals mechanisms of action of clinical abl kinase inhibitors. *Nature biotechnology*, 25(9):1035–1044, 2007.
14. Guillaume Médard, Fiona Pachl, Benjamin Ruprecht, Susan Klaeger, Stephanie Heinzlmeir, Dominic Helm, Huichao Qiao, Xin Ku, Mathias Wilhelm, Thomas Kuehne, et al. Optimized chemical proteomics assay for kinase inhibitor profiling. *Journal of proteome research*, 14(3):1574–1586, 2015.
15. Martin Frejno, Riccardo Zenezini Chiozzi, Mathias Wilhelm, Heiner Koch, Runsheng Zheng, Susan Klaeger, Benjamin Ruprecht, Chen Meng, Karl Kramer, Anna Jarzab, et al. Pharmacoproteomic characterisation of human colon and rectal cancer. *Molecular systems biology*, 13(11):951, 2017.
16. Herbert B Schiller, Christoph H Mayr, Gabriela Leuschner, Maximilian Strunz, Claudia Staab-Weijnitz, Stefan Preisendörfer, Beate Eckes, Pia Moinsadeh, Thomas Krieg, David A Schwartz, et al. Deep proteome profiling reveals common prevalence of mzb1-positive plasma b cells in human lung and skin fibrosis. *American journal of respiratory and critical care medicine*, 196(10):1298–1310, 2017.
17. Fabian Coscia, Ernst Lengyel, Jaikumar Duraiswamy, Bradley Ashcroft, Michal Bassani-Sternberg, Michael Wierer, Alyssa Johnson, Kristen Wroblewski, Anthony Montag, S Diane Yamada, et al. Multi-level proteomics identifies ct45 as a chemosensitivity mediator and immunotherapy target in ovarian cancer. *Cell*, 175(1):159–170, 2018.
18. Jialin Yu, Shaoping Shi, Fang Zhang, Guodong Chen, and Man Cao. Predgly: predicting lysine glycation sites for homo sapiens based on xgboost feature optimization. *Bioinformatics*, 35(16):2749–2756, 2019.
19. Reda Rawi, Raghvendra Mall, Khalid Kunji, Chen-Hsiang Shen, Peter D Kwong, and Gwo-Yu Chuang. Parsnip: sequence-based protein solubility prediction using gradient boosting machine. *Bioinformatics*, 34(7):1092–1098, 2018.