Classification Of Imbalanced Data On Crotonylation Sites Using Lightgbm With Adasyn Oversampling

Favorisen R. Lumbanraja[1]*, Ardella Dean Awalia[2], Sri Karnila[3], Mohammad Reza Faisal[4], Aristoteles[5] and Akmal Junaidi[6]

[1*,2,5,6] Computer Science, Faculty of Mathematics and Natural Sciences, Lampung University, Bandar Lampung, Indonesia
[3] Doctoral Program, Faculty of Mathematics and Natural Sciences, Lampung University, Bandar Lampung, Indonesia
[4] Computer Science, Faculty of Mathematics and Natural Sciences, Lambung Mangkurat University, Banjarbaru, Indonesia
favorisen.lumbanraja@fmipa.unila.ac.id, ardella.da@gmail.com, 2237061008@students.unila.ac.id, reza.faisal@ulm.ac.id, akmal.junaidi@fmipa.unila.ac.id

Abstract. Histone crotonylation is one of newly identified post-translational modification. It may lead to changes in structure and function of proteins due to its power in gene expression regulation. Various kind of diseases have been found to be associated with this modification. That makes it indispensable to have reliable method that can accurately recognize this new type of acylation easily and quickly. This study was performed using LightGBM to compare the classification results between with and without the use of ADASYN in dealing with imbalanced problem between positive and negative class. The data was collected from the UniProt database consisting 1006 sequences in total. A combination of five feature extraction methods was used to transform raw sequence into feature vectors. We employed 5-fold cross-validation to determine the optimal hyperparameter combinations during model training. Subsequently, the best model was selected to evaluate the test data in model testing. The results suggest that classification using oversampled data yields a higher MCC than using original imbalanced data. However, classification by applying ADASYN did not give statistically significant difference.

Keywords: Classification, crotonylation, oversampling, ADASYN, LightGBM.

1 Introduction

Numerous studies have been done relating to post-translational modifications (PTMs). These biochemical alterations occur on single or multiple amino acid after a process called translation in protein synthesis [1]. PTMs play a significant role in the activity, function and regulation of protein structure. The identification of associated PTMs provides a fundamental basis for understanding biological processes, treating disease, and

S. Hadi et al. (eds.), *Proceedings of the 5th International Conference on Applied Sciences, Mathematics, and Informatics (ICASMI 2024)*, Advances in Physics Research 13,
https://doi.org/10.2991/978-94-6463-730-4_9

drug development. PTM sites can be identified using a variety of in lab methods, including radioactive labelling, chromatin immunoprecipitation (ChIP), mass spectrometry (MS), and liquid chromatography [2]. However, these laboratory techniques may have higher costs and fees.

Lysine crotonylation discovered by Tan et al. [3] is a post-translational modification that plays a crucial role in regulating gene expression and protein function [4]. It involves the addition of a crotonyl group to specific lysine residues in histones, leading to changes in their structure and function. Because of their role in the cell cycle, they have been found to be implicated in a number of diseases, e.g. acute kidney injury [5], [6], depression [7], HIV latency [8], and cancer [9]. Identifying and analyzing crotonylation sites in histones is essential for understanding the mechanisms underlying protein function and dysregulation.

Machine learning has been a rapidly evolving field within artificial intelligence and computer science. It involves the development and study of statistical algorithms that enable machines to learn and improve from data without being explicitly programmed. In recent years, machine learning techniques have been increasingly employed to predict PTM sites [10]–[14], leveraging advanced algorithms and feature extraction methods. In this case, their ability can be an alternative in identifying crotonylation sites at lower costs.

Liu et al. [10] proposed a method named LightGBM-CrotSite to predict histone crotonylation sites combining SMOTE oversampling and LightGBM classifier. Feature selection was performed to select the 833 features extracted using Elastic Net. The accuracy, MCC, and AUC of their method were 98.99%, 0.9798, and 0.9996, respectively. Random Forest was used by Qiu et al. [13] in identifying lysine crotonylation sites in histone proteins. They were utilizing ensemble classifier through voting system to handle imbalance data and it gave good enough MCC, 0.8260. Another study was done by Zhao et al. [15] to predict nonhistone crotonylation sites in Papaya species using Convolutional Neural Network (CNN). With a highly imbalanced data, 2,742 positive and 29,676 negative used for ten-fold cross-validation, 711 positive and 7,458 negative used for independent test, they achieved lower MCC. 0.339 and 0.335 for ten-fold cross-validation and independent test, respectively.

This study aims to build a model to classify histone crotonylation sites using machine learning methods, with a focus on different oversampling technique, ADASYN, to address the imbalance class in the crotonylation dataset. It is hoped that this study can contribute to the growing field of machine learning in biology and provide valuable insight into the histone crotonylation.

2 Materials and Methods

2.1 Dataset

Data was obtained from the previous study [10] consisting 159 crotonylated sequences and 847 noncrotonylated sequences. Crotonylated sequences refer to the positive class. Therefore, we can say that the ratio between positive and negative class is nearly 1:5 as shown in Fig. 1. The length of each sequence is 31 amino acids, where 15 amino acids

being upstream or downstream, and 1 amino acid in the middle of the sequence is the residue being crotonylated (or noncrotonylated). These sequences were collected from UniProt database with the keywords of histone, human, and mouse.

Fig. 2 shows two sequences of each class that used in this study as an example. Positive class represented by the sequence with K-residue in the middle of the sequence colored by red, while the sequence with the blue K-residue in the middle represent negative class. The workflow of this study can be seen in Fig. 3.

First, we collected data from previous study [10]. Second, we combined each class and separated it as two subset, train data and test data, to avoid information leakage from each other. Feature extraction was done for every subset. Third, to find best model with best hyperparameters, k-fold cross validation (k=5) was used for training process. In this process, the k-1/k fold was treated as train set and the 1/k fold was treated as test set. Oversample was performed just in train set, not in the test set. Then, the best model obtained from training process would be used to test the test data and used accuracy, specificity, sensitivity, MCC, and AUC as evaluation metrics.

2.2 Methods

Feature Extraction. Feature extraction is a technique used to transform raw data into features that can represent the data [16]. In case of protein sequences data, those sequences are extracted by one or more protein descriptor. It is used to obtain information of chemical bond and relevant characteristic of a protein sequence [17]. There are two types of protein descriptors based on the length of features obtained. First, static length protein descriptors generate the same number of features regardless the length of protein sequence. Second, dynamic length protein descriptors generate features which depend on sequence length or parameter used in the descriptor.

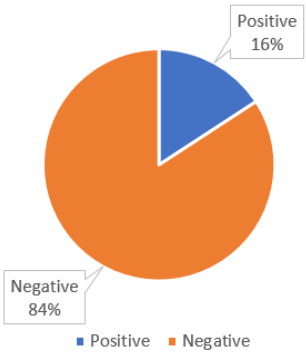

Fig. 1. Ratio between Positive and Negative Class.

>Sequence_1
SKRATQKTRAMMARTKQTARKSTGGKAPRKQ

>Sequence_2
MMARTKQTARKSTGGKAPRKQLATKAARKSA

Fig. 2. Example of Positive and Negative Sequence.

In this study, we use the same protein descriptors as previous study [10] to extract features from raw sequences, i.e. binary encoding (BE) [18], position-weight amino acid (PWAA) [11], encoding based on grouped weight (EBGW) [19], k-nearest neighbors (KNN) [12], and pseudo-position specific scoring matrix (PsePSSM) [20]. According to the type of protein descriptor used, BE is considered as dynamic length, whereas the rest are static length protein descriptor.

1. BE: BE is used as a categorical encoding technique where it represents different categories using binary code, zero and one. Each amino acid in the sequence was encoded according to 'ACDEFGHIKLMNPQRSTVWYX' order. As an example, amino acid 'K' was encoded as '00000001000000000000'. Consequently, every sequence was transformed into 651 feature vector.
2. *PWAA*: PWAA extracts amino acid position information in a protein sequence [11]. Position information of amino acid in a sequence with the length 2L + 1 (L denotes upstream or downstream) can be obtained using Equation (1). Then, 20 feature vectors were obtained.

$$Ci= 1L\ (L+1)j=-LLxi,jj+\ jL \tag{1}$$

3. *EBGW*: EBGW was proposed by Zhang et al. [19] to extract the physicochemical properties characteristic of protein. Protein sequence consist of 20 amino acids through a series of biochemical reactions. Hence, different amino acid must have different physicochemical properties. Amino acids can be divided into four categories as follow:
   Acidic amino acid C1 = {A, F, G, I, L, M, P, V, W},
   Basic amino acid C2 = {Q, N, S, T, Y, C},
   Neutral and Polar amino acid C3 = {D, E},
   Neutral and Hydrophobic amino acid C4 = {H, K, R}

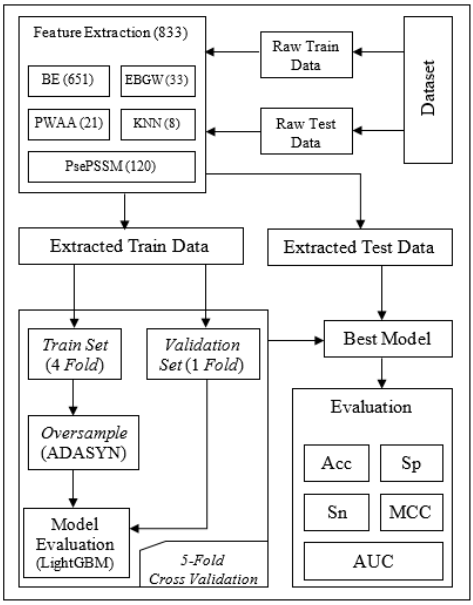

Fig. 3. Research Workflow.

The four types of amino acid are combined to obtain three new group represented in Equation (2)-(4). Given protein sequence $P = p1p2 \ldots p$n, to be transformed into three characteristics.

$$H_1(P_j) = \begin{cases} 1 \text{ if } p_j \in \{C_1, C_2\} \\ 0 \text{ if } p_j \in \{C_3, C_4\} \end{cases} \tag{2}$$

$$H_2(P_j) = \begin{cases} 1 \text{ if } p_j \in \{C_1, C_3\} \\ 0 \text{ if } p_j \in \{C_2, C_4\} \end{cases}, \tag{3}$$

$$H_3(P_j) = \begin{cases} 1 \text{ if } p_j \in \{C_1, C_4\} \\ 0 \text{ if } p_j \in \{C_2, C_3\} \end{cases}, \tag{4}$$

Protein sequences were mapped based on these group, thus, each sequence was reduced into three binary features. Using subsequences L = 11, EBGW generated 33 feature vectors.

4. *KNN*: This protein descriptor extracts protein sequence feature based on local sequence similarity using BLOSUM62 [12]. First, calculate the distance between test site and all known sites. If there are two local protein sequences with the length of N, $S_1 = (S_1(1), S_1(2), \ldots, S_1(N))$ dan $S_2 = (S_2(1), S_2(2), \ldots, S_2(N))$, the distance between those protein can be expressed in Equation (5).

$$\text{Dist}(S_1,S_2)=1-\frac{\sum_{j=1}^{L}\text{sim}(S_1(j),S_2(j))}{N} \quad (5)$$

Similarity score matrix is defined in Equation (6). Max{matrix} denotes the maximum value in similarity matrix and min{matrix} denotes the minimum value in similarity matrix. The feature vector obtain depends on *k* used. In this study, we choose 2, 4, 8, 16, 32, 64, 128, and 256. Hence, there would be 8 feature vectors obtained.

$$sim(S_1(j),S_2(j)) = \frac{Matrix(S_1(j),S_2(j))-\min\{matrix\}}{\max\{Matrix\}-\min\{Matrix\}} \quad (6)$$

5. PsePSSM: PsePSSM was proposed to extract evolutionary information of a protein sequence [20]. In this study, we use PSI-BLAST to obtain PSSM matrix. Then, transformed it into 120 PsePSMM features with $\xi = 5$.

Data Separation. In this study, we separate data into two subsets, train data and test data with three schemes. Data separation was done by considering the proportion of each class for better class representation in each separated data. Train data were used to do hyperparameter optimization and find the best model in training process using 5-fold cross validation. The best model was then used to test the test data in testing process. Following are three schemes to separate the data.

1. *Scheme 1*: The data is separated into 70% data as train data (704 sequences with 111 positive and 593 negative) and 30% data as test data (302 sequences with 48 positive and 254 negative).
2. *Scheme 2*: The data is separated into 80% data as train data (804 sequences with 127 positive and 677 negative) and 20% data as test data (202 sequences with 32 positive and 170 negative).
3. *Scheme 3*: The data is split into 90% data as train data (905 sequences with 143 positive and 762 negative) and 10% data as test data (101 sequences with 16 positive and 85 negative).

ADASYN. He et al. [21] proposed Adaptive Synthetic (ADASYN) with the objective to reduce bias and learn adaptively. This method is based on the idea of oversampling minority data according to their density distribution. It means that there are more synthetic data generated for minority class samples that are more difficult to learn than minority class samples that are easier to learn. ADASYN algorithm can be expressed as follows.

1. Calculate the degree of class imbalance.
2. If the degree of class imbalance is more than maximum value that can be tolerated, continue the process. Otherwise, the algorithm stop.
3. Calculate the number of synthetic data that have to be generated.

4. For each minority sample, find k nearest neighbors using Euclidean distance. After that, find the ratio between the number of majority sample and all sample in the neighbourhood of minority sample. Then, normalize the ratio.
5. Find the number of synthetic data that have to be generater for each minority sample. See [21] for more details. We selected k (n_neighbors) to 5 (default), 7, and 9 for k nearest neighbor.

LightGBM. LightGBM was proposed by Ke et al. [22] based on Gradient Boosting Decision Tree (GBDT) algorithm with an addition of two methods, i.e. gradient-based one side sampling (GOSS) and exclusive feature bundling (EFB). GOSS is a technique to create new subset based on larger gradient. On the other hand, EFB is a technique to bundle mutual exclusive features. The objective was reducing memory consumption as the dimension increase and speed up the computation. The illustration of how gradient boosting work is shown in Fig. 4.

LightGBM is an ensemble method using decision tree as the base classifier, this algorithm can be used for classification and regression task. It combines weak learner sequentially with minimizing error from previous learner. First, it makes initial prediction for each sample. Next, the first tree is build to minimize the error from the initial prediction. The second tree is then build to minimize the error from the first tree, and so on. Finally, the last tree is used to determine final prediction.

There are some hyperparameters can be tuned in LightGBM. In this study, two hyperparameters was used, i.e. learning rate and maximum depth. It can be said that learning rate is a value that determines how much contribution of each tree to final prediction. Small learning rate value will need longer time, but better at predict the final result. Meanwhile, maximum depth is a value that defines maximum depth level of a tree. Increasing the maximum depth of the tree can result in a more complex model, which may be more prone to overfitting. Due to the limitation of time and resources, we selected learning rate to 0.05, 0.1 (default), and 0.15 (0.05 lower and upper from default value), as well as max depth to 4, 6, 10, and 12 (randomly).

Feature Importance in LightGBM can be used to see which features are used and affect the results the most. It helps to decide which protein descriptor that is useful in this study.

Evaluation. Evaluation metrics are required for measuring or evaluating model performance. Those metrics need the class prediction of each sample. Confusion Matrix shows how well model predicts sample of each class. Confusion Matrix for binary classification task is shown in Table I.

Where, True Positive (TP) defines the number of positive samples correctly classified. True Negative (TN) defines the number of negative samples correctly classified. False Positive (FP) defines the number of negative samples incorrectly classified. False Negative (FN) defines the number of positive samples incorrectly classified. Evaluation metrics used in this study are expressed in (7)-(10).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{7}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{8}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{9}$$

$$\text{MCC} = \frac{TP \; x \; TN - FP \; x \; FN}{\sqrt{(TP+FN) \; x \; (TN+FP) \; x \; (TP+FP) \; x \; (TN+FN)}} \tag{10}$$

The models were implemented in Windows 10 Pro version 22H2 with Intel Core i3-6006U processor and 8 GB of RAM.

3 Results and Discussion

The raw data were transformed into feature vectors by combining five protein descriptors, resulting in a total of 833 features.

3.1 Classification Test Results without ADASYN

Best model of each scheme from training process without oversampling would be tested on test data to check whether these models can work well in unseen data. The result can be seen in Table II. Best model in Scheme 1 used learning_rate=0.2 and max_depth=6. Best model in Scheme 2 used learning_rate=0.1 and max_depth=12. Best model in Scheme 3 used learning_rate=0.15 and max_depth=10.

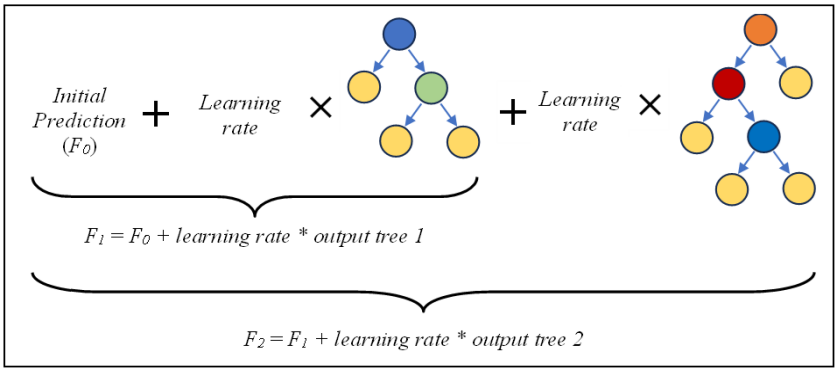

Fig. 4. Gradient Boosting Illustration.

Table 1. Confusion Matrix

		Actual	
		Positive	Negative
Predicted	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Table 2. Testing Result Without ADASYN

Scheme	Acc	Sp	Sn	MCC	AUC
1	96.03%	98.03%	85.42%	0.8491	0.9611
2	95.05%	97.06%	84.25%	0.8143	0.9680
3	95.05%	97.65%	81.25%	0.8101	0.9860

Table 3. Testing Result With ADASYN

	k	Acc	Sp	Sn	MCC	AUC
Scheme 1	5	96.03%	98.43%	83.33%	0.8473	0.9686
	7	95.36%	97.24%	85.42%	0.8266	0.9690
	9	95.70%	97.64%	85.42%	0.8377	0.9674
Scheme 2	5	95.05%	96.47%	87.50%	0.8195	0.9419
	7	95.05%	96.47%	87.50%	0.8195	0.9640
	9	94.55%	95.88%	87.50%	0.8044	0,9496
Scheme 3	5	96.04%	97.65%	87.50%	0.8515	0.9868
	7	95.05%	96.47%	87.50%	0.8195	0.9371
	9	96.04%	97.65%	87.50%	0.8515	0.9890

Table 4. Statistical Paired T-Test

			Original – ADASYN (*n_neighbors=5)*	Original – ADASYN (*n_neighbors=7*)	Original – ADASYN (*n_neighbors=9*)
Paired Differences	*Mean*		-0.3300	0.2208	-0.0546
	Std dev		0.5716	0.3824	0.8143
	Std err Mean		0.0033	0.2208	0.4701
	95% Confidence Interval	*L*	-1.7501	-0.7291	-2.077
		U	1.0900	1.1706	1.9681
	t		-1	1	-0.12
	df		2	2	2
	Sig. (2-tailed)		0.4226	0.4226	0.9181

Table 5. Comparisons With Previous Study

	Acc	Sp	Sn	MCC	AUC
LightGBM-CrotSite	98.99%	99.11%	98.86%	0.9798	0.9996
This Study	96.04%	97.65%	87.50%	0.8515	0.9890

3.2 Classification Test Results with ADASYN

Best model of each scheme from training process with oversampling would also be tested on test data to check whether these models can work well in unseen data. Table III shows the testing result from each scenario. Best model obtained from Scheme 1 was using k=5 with learning_rate=0.15 and max_depth=4. Best model in Scheme 2 was using k=7 with learning_rate=0.15 and max_depth=10. And best model in Scheme 3 was using k=9 with learning_rate=0.15 and max_depth=12.

3.3 Feature Importances

Feature Importances can be obtained from model that was used in training process. Top 20 of 833 features are presented to demonstrate their significant contribution to the final prediction. Fig. 5, Fig. 6, and Fig. 7 shows the Feature Importances of the model that achieved the best classification result for Scheme 1, Scheme 2, and Scheme 3, respectively. It can be seen that feature 'knn_2' was the most used feature in all schemes.

3.4 Statistical Test

Paired t-test was conducted as a statistical test to compare the data before and after treatment. In this study, we did a two-tailed paired t-test on the test results between accuracy without ADASYN oversampling and accuracy with ADASYN for each k

(n_neighbors) shown in Table IV. There was no significant difference in the accuracy after applying ADASYN compared to the accuracy before applying ADASYN for the three tests, indicated by $p > .05$.

3.5 Comparison with Previous Study

Several methods have been proposed to classify histone and nonhistone crotonylation sites. We compared this study to previous study that used the same dataset. Table V shows the comparison of this method and LightGBM-CrotSite. It is obvious that our method has not been able to outperform the previous method.

4 Conclusion

The purpose of this study was building a model using LightGBM classification by applying oversampling technique, ADASYN, to address the imbalance issue between positive class and negative class in the dataset. The data was moderately imbalanced with 159 positive data and 847 negative data. Dataset split into train data and test data based on three schemes. Five protein descriptor was used in feature extraction process. BE extracted 651 features, PWAA extracted 21 features, EBGW extracted 33 features, KNN extracted 8 features, and PsePSSM extracted 120 features. The extracted features were combined as one dataset with a total 833 features. ADASYN was used to oversample the minority (positive class), so it could balance the majority (negative class). Note that, ADASYN was only on train data. Hyperparameters used in LightGBM were learning rate and maximum depth. In training process, 5-fold cross validation was performed to do some hyperparameter optimization. The best model from each scenario in training process then used to test the test data.

Although, the application of ADASYN to balance the minority class have a better result than the imbalanced class, there was no statistical difference of its accuracy between each k. It indicates that the application of ADASYN did not have enough impact on improving classification results. Choosing the combination of sampling methods and classifier to use is a crucial step before starting research. It may depend on the distribution of the data, because no one size fits all.

For future research, we recommend to use another sampling technique to address the imbalanced and use more hyperparameters to be tuned, e.g. maximum number of leaves, because LightGBM use leaf-wise method to grow a tree. Another hyperparameter like number of estimators used can also be considered. Build more trees to improve the previous tree so that the final prediction will be closer to the true value.

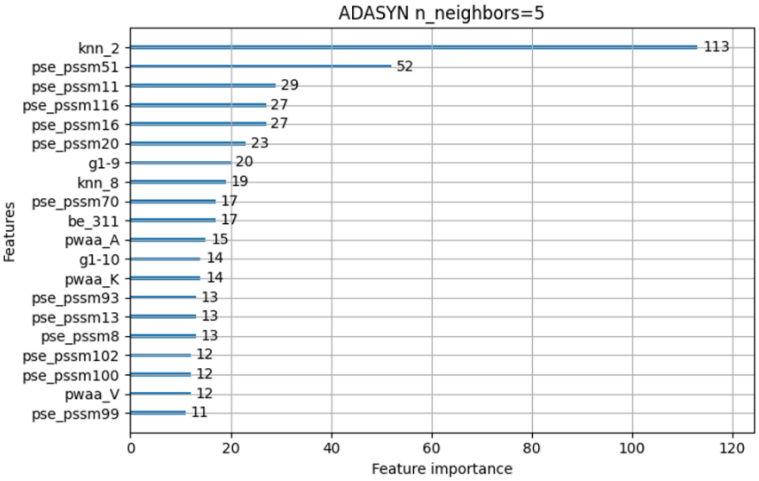

Fig. 5. Feature Importances of Best Model in Scheme 1.

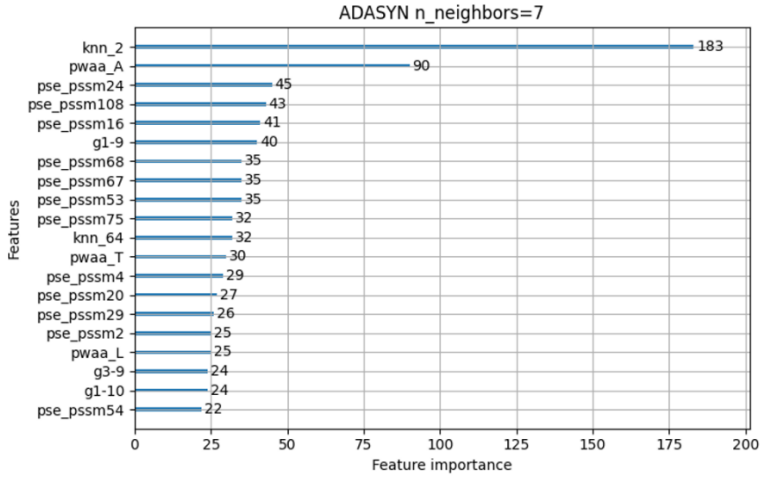

Fig. 6. Feature Importances of Best Model in Scheme 2.

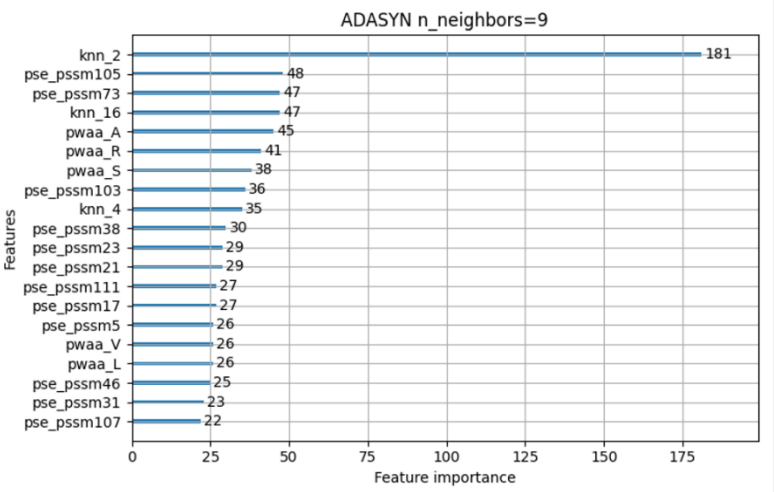

Fig. 7. Feature Importances of Best Model in Scheme 3.

References

1. Carter, M., Shieh, J.: Biochemical assays and intracellular signaling. In: Guide to Research Techniques in Neuroscience, pp. 311–343. Elsevier, Amsterdam (2015). https://doi.org/10.1016/B978-0-12-800511-8.00015-0
2. Dou, L., Yang, F., Xu, L., Zou, Q.: A comprehensive review of the imbalance classification of protein post-translational modifications. Briefings in Bioinformatics **22**(5) (2021). https://doi.org/10.1093/bib/bbab089
3. Tan, M., Luo, H., Lee, S., Jin, F., Yang, J.S., Montellier, E., Buchou, T., Cheng, Z., Rousseaux, S., Rajagopal, N., Lu, Z., Ye, Z., Zhu, Q., Wysocka, J., Ye, Y., Khochbin, S., Ren, B., Zhao, Y.: Identification of 67 histone marks and histone lysine crotonylation as a new type of histone modification. Cell **146**(6), 1016–1028 (2011). https://doi.org/10.1016/j.cell.2011.08.008
4. Wei, W., Mao, A., Tang, B., Zeng, Q., Gao, S., Liu, X., Lu, L., Li, W., Du, J.X., Li, J., Wong, J., Liao, L.: Large-scale identification of protein crotonylation reveals its role in multiple cellular functions. Journal of Proteome Research **16**(4), 1743–1752 (2017). https://doi.org/10.1021/acs.jproteome.7b00012
5. Zhao, Y., Hao, S., Wu, W., Li, Y., Hou, K., Liu, Y., Cui, W., Xu, X., Wang, H.: Lysine crotonylation: an emerging player in DNA damage response. Biomolecules **12**(10), 1428 (2022). https://doi.org/10.3390/biom12101428
6. Ruiz-Andres, O., Sanchez-Niño, M.D., Cannata-Ortiz, P., Ruiz-Ortega, M., Egido, J., Ortiz, A., Sanz, A.B.: Histone lysine crotonylation during acute kidney injury in mice. Disease Models & Mechanism **9**(6), 633–645 (2016). https://doi.org/10.1242/dmm.024455
7. Liu, Y., Li, M., Fan, M., Song, Y., Yu, H., Zhi, X., Xiao, K., Lai, S., Zhang, J., Jin, X., Shang, Y., Liang, J., Huang, Z.: Chromodomain Y-like protein–mediated histone crotonylation regulates stress-induced depressive behaviors. Biological Psychiatry **85**(8), 635–649 (2019). https://doi.org/10.1016/j.biopsych.2018.11.025

8. Jiang, G., Nguyen, D., Archin, N.M., Yukl, S.A., Méndez-Lagares, G., Tang, Y., Elsheikh, M.M., Thompson III, G.R., Hartigan-O'Connor, D.J., Margolis, D.M., Wong, J.K., Dandekar, S.: HIV latency is reversed by ACSS2-driven histone crotonylation. The Journal of Clinical Investigation **128**(3), 1190–1198 (2018). https://doi.org/10.1172/JCI98071
9. Wan, J., Liu, H., Ming, L.: Lysine crotonylation is involved in hepatocellular carcinoma progression. Biomedicine & Pharmacotherapy **111**, 976–982 (2019). https://doi.org/10.1016/j.biopha.2018.12.148
10. Liu, Y., Yu, Z., Chen, C., Han, Y., Yu, B.: Prediction of protein crotonylation sites through LightGBM classifier based on SMOTE and elastic net. Analytical Biochemistry **609**, 113903 (2020). https://doi.org/10.1016/j.ab.2020.113903
11. Shi, S. P., Qiu, J. D., Sun, X. Y., Suo, S. B., Huang, S. Y., Liang, R. P.: A method to distinguish between lysine acetylation and lysine methylation from protein sequences. Journal of Theoretical Biology **310**, 223–230 (2012). https://doi.org/10.1016/j.jtbi.2012.06.030
12. Gao, J., Thelen, J.J., Dunker, A.K., Xu, D.: Musite, a tool for global prediction of general and kinase-specific phosphorylation sites. Molecular & Cellular Proteomics **9**(12), 2586–2600 (2010). https://doi.org/10.1074/mcp.M110.001388
13. Qiu, W. R., Sun, B. Q., Xiao, X., Xu, Z. C., Jia, J. H., Chou, K. C.: iKcr-PseEns: Identify lysine crotonylation sites in histone proteins with pseudo components and ensemble classifier. Genomics **110**(5), 239–246 (2018). https://doi.org/10.1016/j.ygeno.2017.10.008
14. Qiu, W. R., Sun, B. Q., Tang, H., Huang, J., Lin, H.: Identify and analyze crotonylation sites in histone by using support vector machines. Artificial Intelligence in Medicine **83**, 75–81 (2017). https://doi.org/10.1016/j.artmed.2017.02.007
15. Zhao, Y., He, N., Chen, Z., Li, L.: Identification of protein lysine crotonylation sites by a deep learning framework with convolutional neural networks. IEEE Access **8**, 14244–14252 (2020). https://doi.org/10.1109/ACCESS.2020.2966592
16. Salau, A.O., Jain, S.: Feature extraction: A survey of the types, techniques, applications. In: 2019 International Conference on Signal Processing and Communication (ICSC), pp. 158–164. IEEE, New Delhi (2019). https://doi.org/10.1109/ICSC45622.2019.8938371
17. Emonts, J., Buyel, J.F.: An overview of descriptors to capture protein properties – Tools and perspectives in the context of QSAR modeling. Computational and Structural Biotechnology Journal **21**, 3234–3247 (2023). https://doi.org/10.1016/j.csbj.2023.05.022
18. Sohrawordi, M., Hasan, M.A.M., Hasan, M.A.M.: LyFor: Prediction of lysine formylation sites from sequence-based features using support vector machine. In: 2020 IEEE Region 10 Symposium (TENSYMP), pp. 250–253. IEEE, Dhaka (2020). https://doi.org/10.1109/TENSYMP50017.2020.9230689
19. Zhang, Z.H., Wang, Z.H., Zhang, Z.R., Wang, Y.X.: A novel method for apoptosis protein subcellular localization prediction combining encoding based on grouped weight and support vector machine. FEBS Letter **580**(26), 6169–6174 (2006). https://doi.org/10.1016/j.febslet.2006.10.017
20. Chou, K.C., Shen, H.B.: MemType-2L A web server for predicting membrane proteins and their types by incorporating evolution information through Pse-PSSM. Biochemical and Biophysical Research Communications **360**(2), 339–345 (2007). https://doi.org/10.1016/j.bbrc.2007.06.027
21. He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), pp. 1322–1328. IEEE, Hong Kong (2008). https://doi.org/10.1109/IJCNN.2008.4633969

22. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T. Y.: LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems 30 (2017)