RESEARCH

Paper 6 in applied machine learning

Aakash Nepal and Cuong Gia Pham

Full list of author information is available at the end of the article
*Equal contributor

Abstract

Goal of the project: This project aimed to choose two datasets from the "ML and clinical epigenetics" paper[1], perform feature extraction using PyMethylProcess[2], train eXtreme Gradient Boosting (XGBoost) and linear Support Vector Machine (SVM-Lin) classifiers, evaluate their performance using specified measures, and conduct a feature importance analysis for precision medicine.

Main results of the project: With the help of PyMethylProcess, the data has been transformed into a form, which helps researchers easier and faster in applying machine learning models in their projects. But the result with the help of PyMethylProcess wasn't as good as the result from the original. However, since we separated samples into two groups only, the result is more or less still containing confounders.

Personal key learning :
1. Pham Gia Cuong: Multi-modal deep learning model, understanding DNA methylation data preprocessing
2. Aakash Nepal: was introduced to data augmentation,nohup, and learned more about epigenomics, XGBoost and SVM.

Estimation working hours:
1. Pham Gia Cuong: 7 hours per week
2. Aakash Nepal: 7 hours per week.

Project evaluation: 2.3 out of 5

Number of words : Approx. 2500(without abstract, captions, references)

1 Goal of the project

Clinical epigenetics in precision medicine, notably in oncology, strives to enhance diagnosis, prognosis, and therapy selection by utilizing epigenetic information. Machine Learning (ML) techniques may be used to construct prediction models for categorizing cancer subtypes and predicting treatment results by evaluating changes in gene expression patterns induced by epigenetic alterations. This method improves cancer diagnostic accuracy, permits individualized treatment methods based on epigenetic profiles, and aids in the discovery of new therapeutic targets. Precision medicine can enhance patient outcomes in cancer care by integrating clinical epigenetics to increase patient categorization, and therapy effectiveness, and decrease unwanted effects. The goal of this project was to choose two different datasets from the "ML and clinical epigenetics" paper[1], perform feature extraction using a framework, train two ML classifiers, evaluate them using the measures listed on the paper, and do the feature important analysis. For this, two ML classifiers, namely eXtreme Gradient Boosting(XGBoost) and linear support vector machine(SVM-Lin) were trained on those extracted features from the PyMethylProcess framework to classify the samples from the chosen datasets(GSE97362[3] and GSE112047[4]). The models were evaluated using the performance measures described in Table 1 of the ”ML and clinical epigenetics” paper[1]. Furthermore, a feature importance analysis was also performed to understand the biological relevance of the most important features found by using those machine learning methods which might add some meaning to precision medicine.

2 Data

For this project we used two data from Gene Expression Omnibus(GEO) database which were also found in Table 2 of the ”ML and clinical epigenetics” paper[1]. One GEO dataset with accession number GSE97362 was about CHARGE and Kabuki syndromes study which was conducted by Butcher et al.[5]. This data was gained through methylation profiling using a genome-tiling array. The bisulphite converted DNA samples was hybridized to the Illumina Infinium Human Methylation450 Beadchip platform. The dataset consists of 125 healthy control samples and 109 samples of CHARGE and Kabuki syndromes with CHD7 loss of function mutations or KMT2D loss of function mutations making it a total of 234 samples. Another dataset with GEO accession number GSE112047 that we used in our project was about genome-wide DNA methylation analysis of tumor and adjacent normal prostate tissues. Following genomic DNA extraction and bisulfite conversion, the Illumina Infinium methylation 450k bead chip array was used for analysis. It contained 16 healthy controls and 31 tumor samples making it a total of 47 samples. These data were originally preprocessed in R 3.4.2 with the Illumina normalization approach and background correction.

3 Methods

3.1 DeepPerVar

Based on last week’s report, we successfully deployed only DeepPerVar[6] and conducted tests on a small test set. DeepPerVar is a multi-modal deep learning framework used for predicting genome-wide quantitative epigenetic signals. Additionally, it assesses the functional impact of noncoding variants on an individual level by quantifying the allelic difference in predictions. We started this tool by exploring the idea of determining feature importance. To analyze the importance of different features, we examined the weights of each subnet and layer. Through this analysis, we observed that the sequence subnet had the highest weight. This finding appeared evident, prompting us to further investigate the sequence subnet. Our aim was to identify potential patterns or significant positions within this subnet. To achieve this, we used the perturbing input method. This technique involved systematically removing nucleotides from each sequence and observing how the output changed. The degree of change in the output provided insight into the relative importance of the specific feature. Our hope was to uncover key patterns or positions that significantly influenced the prediction outcomes. However, we didn’t find any significant difference with a small dataset. Besides that, we encountered also some difficulties in searching for a new dataset. DeepPerVar predicts the modification of histone or methylation based on the normalized count of peaks, which are gathered from the alignment files. Hence, we stopped investigating further for this tool and switched to PyMethylProcess.

3.2 PyMethylProcess

To get DNA methylation data, biological samples(e.g., blood and cells) are collected, DNA is extracted, bisulfite conversion is conducted, and different methylation tests, such as PCR or sequencing, are used to determine the methylation status of the DNA. PyMethylProcess is a Python package that executes different R scripts for processing and analyzing DNA methylation data. PyMethylProcess offers various functionalities to preprocess and normalize DNA methylation data, perform quality control, detect differentially methylated regions (DMRs) , and conduct downstream analysis. Besides that, PyMethylProcess supports also many types of DNA methylation data, including both array-based (e.g., gained using Illumina Infinium arrays) and sequencing-based (e.g., gained using bisulfite sequencing) data. Furthermore, PyMethylProcess provides methods for quality control, such as filtering out low-quality probes or samples based on various criteria. It also offers statistical methods to identify DMRs between different groups or conditions. From the last report, we struggled with installing PyMethylProcess on the system's environment because of conflict between versions of packages. However, with the help of Apptainer[7], we were able to fully deploy and tested it on our GEO datasets. First, PyMethylProcess will read and analyse the idat and phenotype files downloaded from GEO database by using minfi[8] or enmix[9] DNA methylation analysis pipeline written in R script. The output from the pipeline is the beta values matrix, which provides a comprehensive view of DNA methylation patterns across multiple samples and genomic locations. For this project, we used command line interface of PyMethylProcess to download the GEO datasets, and then to run the preprocessing pipeline with default settings, the non-autosomal CpGs were removed and missingness was reported. After that, we used it for the imputation and finally, the necessary features were selected with the help of pymethyl-preprocess command using mean absolute deviation which gave us the final preprocessed pickle file. This pickle file was loaded into the jupyter notebook interface with the help of MethylationArray from PyMethylProcess. At last, the relevant columns were selected and merged from pheno and beta data.

3.3 Data augmentation and Feature selection

Data augmentation is a method to artificially extend the dataset by creating more samples which might help to reduce overfitting. For the GSE112047 dataset with a low sample size of 47, we applied data augmentation techniques for generating additional numerical data such as scaling, adding noise, and shuffling to the original data and then concatenated them together with the original data to increase the sample size to 235 samples. it consisted now of 155 tumor and 80 control samples. Another key step in improving the model performance, reducing overfitting, and improving generalization is the feature selection. For our dataset GSE97362, we classified syndromes into the diseased class and controls into the healthy class. After that, we performed feature selection using LASSO (Least Absolute Shrinkage and Selection Operator) which is a type of regularization method used in linear regression to select the features and handle multicollinearity. It has one hyperparameter α and it supports sparsity in coefficient estimates by adding a penalty component to the ordinary least squares objective function which decreases the less relevant features towards zero. When doing feature selection using LASSO, we used a α value of 0.005 as regularization intensity. It was then fitted to the data, and the resultant coefficients were computed using the coef_ function. Because non-zero coefficients reflect selected features, their associated indices were gathered. The columns of the selected features were obtained from the data using these indices. After data augmentation and feature selection accordingly, both of the datasets were split, allocating 70% of the data for training and 30% for testing.

3.4 Machine learning models

From the train and test sets, we now can apply it to the machine learning model. In this project, we employed two machine learning models, including XGBoost and SVM-Lin.

XGBoost classifier is a scalable machine-learning system for tree boosting. XGBoost iteratively builds an ensemble of decision trees by correcting the mistakes made by previous trees. It combines weak learners, also called decision stumps, and evaluates their performance using a loss function. The loss function measures the discrepancy between the predicted and actual labels. By adding the loss function from the previous tree multiplied by a learning rate to create the next tree. This iterative approach results in an ensemble of boosting trees. After having an ensemble boosting trees model, the prediction is done by calculating the average output from all trees. For the dataset GSE97362, we used cross-validation to find the best optimal learning rate of 0.01 keeping other hyperparameters as default for building the XGBoost model. Due to a warning in Python, we choose Root Mean Squared Error as a metric for the model. At last, we used XGBoost's inbuilt feature_importances_ function for feature importance analysis.

Another machine learning algorithm for the classification task that we used is SVM-Lin. The main goal of it is to find a hyperplane (from a set of hyperplanes) with a maximal margin in N-dimensional space that distinctly classifies the dataset, where N is the number of features. SVM-Lin has only one regularization parameter which is a hyperparameter (in Python C for "Cost"). The trade-off between maximizing the margin and keeping the minimum of error using the regularization parameter C, SVM-Lin finds a good compromise between overfitting and underfitting. If C is set too high, the model may become too sensitive to noise or irrelevant information, resulting in poor generalization. On the other hand, if C is set too low, the model may underfit and fail to capture essential patterns in the data. For this project, we used cross-validation to find the optimal C value of 0.0005 for our dataset GSE112047, and to cope with class imbalance, we set class_weight= "balanced" for SVM-Lin in order to penalize mistakes on the minority class by an amount equal to how under-represented it is. At last, we used the inbuild coef_ function of SVM-Lin for feature importance analysis.

3.5 Model evaluation

We used accuracy, and area under the ROC curve (AUC-ROC) as criteria for evaluation. The accuracy metric as described by equation (1) evaluates a classification model's overall accuracy by measuring the ratio of truly classified samples to the total number of samples.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.
The AUC-ROC is a summary measure of the model's performance that shows the likelihood that the model would score a randomly chosen positive instance higher than a randomly chosen negative instance. AUC-ROC values range from 0 to 1, with higher values suggesting stronger discrimination and prediction capability of the model.
Besides that, to be able to have a better overview of the performance, we also gathered the sensitivity, specificity, precision, recall, and F1-score value of each model. Precision is a crucial evaluation criterion. Precision is the ratio of true positives predicted properly by the model and it is calculated as shown in equation (2).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

Similarly, Recall (also known as sensitivity) is a measure that indicates how well a model can recognize positive results. As outlined in equation (3), recall is obtained by dividing the number of TP by the total number of participants.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

On the other hand, specificity measures how effectively the model avoids incorrectly categorizing negative cases as positive. As shown in equation (4), specificity is obtained by dividing the number of TN by the total number of TN and FP.

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{4}$$

where TN is the number of true negatives and FP is the number of false positives. As described by equation (5), the F1-score is a metric that considers both precision and recall. Only when recall and precision both have a value of 1, does the F1 Score become 1. Only when both recall and precision is strong can the F1 score rise.

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

4 Results and Discussion

4.1 Model evaluation

4.1.1 GSE97362 dataset

For this dataset, we applied XGBoost to distinguish two cohorts, which are the control group and the Kabuki-CHARGE syndrome group. From Figure 1, we can see that the AUC value is 0.89 which shows that the model can somewhat well discriminate between the normal group and Kabuki-CHARGE group. Besides that, Table 1 presents the performance metrics of XGBoost and SVM from the original paper[3].

Performance Metric	XGBoost	SVM
Accuracy	0.8169	0.996
Sensitivity	0.9143	1.0
Specificity	0.7222	1.0

Table 1 **Performance Metrics between XGBoost and SVM from original paper**

The metrics evaluated include accuracy, sensitivity, and specificity. The XGBoost model achieved an accuracy of 0.8169, indicating that approximately 81.69% of the predictions made by the model were correct. In comparison, the SVM model exhibited a higher accuracy of 0.996, suggesting a significantly higher level of accuracy in its predictions. The sensitivity of the XGBoost model demonstrated a value of 0.9143, indicating that it correctly identified approximately 91.43% of the actual positive cases. On the other hand, the SVM model achieved a sensitivity of 1.0, implying that it correctly identified all the positive cases in the dataset. In terms of specificity, the XGBoost model achieved a value of 0.7222, indicating that it correctly identified approximately 72.22% of the actual negative cases. Conversely, the SVM model exhibited a specificity of 1.0, implying that it correctly identified all the negative cases in the dataset. The result from the original paper[3] seems to outperform the XGBoost with the sensitivity and specificity are 1. Besides that, our model also gives 0.83 for the value of the F1 score in normal prediction and 0.80 for syndrome prediction.

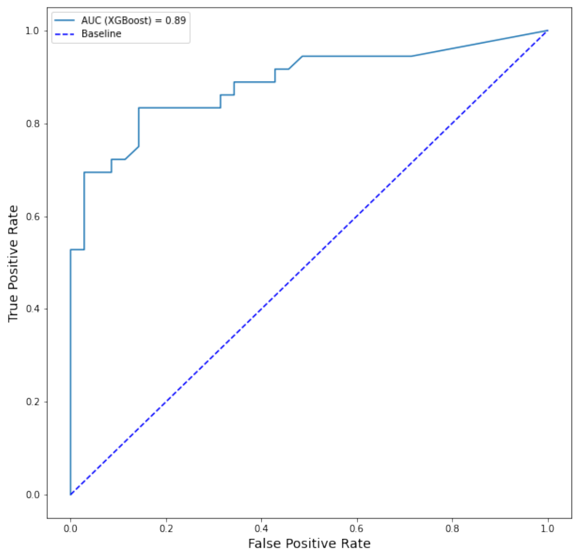

Figure 1 ROC curve of XGBoost on test set of GSE97362

4.1.2 GSE112047 dataset

For this dataset, we applied a SVM-Lin to classify the control group and the cancer group. Figure 2 shows us the ROC curve and the AUC value of running a SVM-Lin on the test dataset of GSE112047. The SVM-Lin model achieved an AUC value of 0.94, which shows that the SVM-Lin model is good at differentiating between the two groups. In comparison with the original paper[4], in which they applied the

Performance Metric	SVM-Lin	Lasso
Accuracy	0.8873	0.97
Sensitivity	0.8	0.96
Specificity	0.9347	0.98

Table 2 Performance Metrics between SVM-Lin and LASSO from original paper

LASSO as a machine learning model, our AUC value seems a little less than the AUC from the original paper (0.98).
Table 2 shows us also the metrics evaluated including accuracy, sensitivity, and specificity. The SVM-Lin model achieved an accuracy of 0.8873, indicating that approximately 88.73% of the model's predictions aligned with the actual outcomes. Conversely, the LASSO model demonstrated a higher accuracy of 0.97, suggesting a superior predictive performance. In terms of sensitivity, the SVM-Lin model correctly identified around 80% of the actual positive cases, as reflected by its sensitivity score of 0.8. Conversely, the LASSO model exhibited a sensitivity of 0.96, indicating its ability to capture a larger proportion of positive cases, approximately 96% in this case. The specificity of SVM-Lin achieved a score of 0.9347, indicating that it accurately identified roughly 93.47% of the true negative cases. In contrast, the LASSO model achieved a higher specificity score of 0.98, indicating a greater ability to correctly identify negative cases, approximately 98% in this scenario. Since the sample size is small and we oversampled the data, the result is less reliable. The F1 score for this model is 0.83 for normal prediction and 0.91 for cancer prediction. Meanwhile, the original paper[4] didn't provide any information about the F1 value.

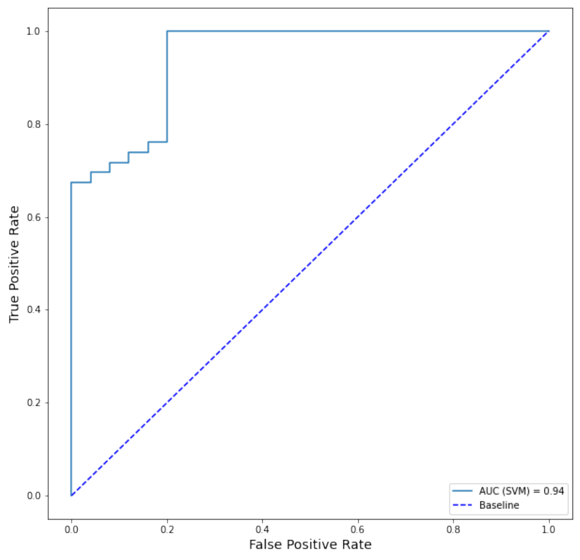

Figure 2 ROC curve of SVM-Lin on test set of GSE112047

4.2 Feature importance

Table 3 shows the 8 most important CpG probes that were found by feature importance analysis. Among the eight CpG probes discovered by XGBoost in GSE97362, we discovered one probe, cg08425810, from the AGAP2 region that is associated with both CHARGE and Kabuki syndromes, which was also reported in the paper of dataset[3]. Some CpG probes like cg17881200 and cg17187762 were found in the shelf (the region that is adjacent to CpG islands), however, no differentially methylated CpG probes were found that were described in the paper[3]. From those found by SVM-Lin, two of the CpG probes, namely cg24396624 associated with the

Order	GSE97362(XGBoost)	GSE112047(SVM-Lin)
1	cg17881200	cg25013753
2	cg12128839	cg06634576
3	cg08425810	cg03221390
4	cg09798888	cg10117599
5	cg19092981	cg25203245
6	cg25587069	cg24396624
7	cg19022697	cg25339566
8	cg09549073	cg22773555

Table 3 The 8 most important CpG probes identified by XGBoost and SVM-Lin model

RARB gene and cg25339566 associated with the TCTEX1D1 gene, were found to be differentially methylated in prostate cancer which were also reported in the supplementary data of the datasets paper[4]. Furthermore, it was found that if the gene RARB is methylated, it might be an indicator of the initiation of prostate cancer[10] and TCTEX1D1 seems to be found highly expressed in The Cancer Genome Atlas Program data of prostate cancer[11]. Moreover, the development of diagnostic methods may be aided by further analysis and verification of these CpG probes, which may offer important insights into the causes and mechanisms of diseases.

5 Contributions

1 Pham Gia Cuong: Overall 50% report and tried DeepPerVar, ran the XGBoost, and deploying tools.
2 Nepal Aakash: Overall 50% report and did the preprocessing with pymethylprocess, ran SVM plus extracted the feature importances.

References

[1] Sebastian Rauschert et al. “Machine learning and clinical epigenetics: a review of challenges for diagnosis and classification”. In: *Clinical Epigenetics* 12 (1 2020), p. 51. DOI: 10.1186/s13148-020-00842-4. URL: https://doi.org/10.1186/s13148-020-00842-4.

[2] Joshua J Levy et al. “PyMethylProcess-convenient high-throughput preprocessing workflow for DNA methylation data”. In: *Bioinformatics* 35.24 (2019), pp. 5379–5381. DOI: 10.1093/bioinformatics/btz594.

[3] Erfan Aref-Eshghi et al. “Genomic DNA methylation signatures enable concurrent diagnosis and clinical genetic variant classification in neurodevelopmental syndromes”. In: *American Journal of Human Genetics* 102 (1 2018), pp. 156–174. DOI: 10.1016/j.ajhg.2017.11.007.

[4] Erfan Aref-Eshghi et al. “Genomic DNA methylation-derived algorithm enables accurate detection of malignant prostate tissues”. In: *Frontiers in Oncology* 8 (2018). DOI: 10.3389/fonc.2018.00015.

[5] Darci T. Butcher et al. "CHARGE and Kabuki Syndromes: Gene-Specific DNA Methylation Signatures Identify Epigenetic Mechanisms Linking These Clinically Overlapping Conditions". In: *The American Journal of Human Genetics* 100.5 (May 2017), pp. 773–788. DOI: 10.1016/j.ajhg.2017.04.004.

[6] Ye Wang and Li Chen. "DeepPerVar: a multi-modal deep learning framework for functional interpretation of genetic variants in personal genome". In: *Bioinformatics* 38.24 (Dec. 2022), pp. 5340–5351. DOI: 10.1093/bioinformatics/btac696.

[7] Dave Dykstra. "Apptainer Without Setuid". In: *ArXiv* abs/2208.12106 (2022).

[8] Martin J. Aryee et al. "Minfi: a flexible and comprehensive Bioconductor package for the analysis of Infinium DNA methylation microarrays". In: *Bioinformatics* 30.10 (2014), pp. 1363–1369. DOI: 10.1093/bioinformatics/btu049.

[9] Zongli Xu et al. "ENmix: a novel background correction method for Illumina HumanMethylation450 BeadChip". In: *Nucleic Acids Research* 44.3 (2016), e20. DOI: 10.1093/nar/gkv907.

[10] B Rybicki et al. "Methylation of Retinoic Acid Receptor, Beta (RARB) Gene Increases Risk for Prostate Cancer in African-American Men". In: *Cancer Epidemiology, Biomarkers Prevention* 20.4 (Mar. 2011), pp. 718–718. ISSN: 1055-9965. DOI: 10.1158/1055-9965.EPI-11-0091. eprint: https://aacrjournals.org/cebp/article-pdf/20/4/718/2273148/718.pdf. URL: https://doi.org/10.1158/1055-9965.EPI-11-0091.

[11] Ma'ayan Laboratory. *TCGA Signatures of Differentially Expressed Genes for Tumors*. Accessed: 1 July 2023. Year. URL: https://maayanlab.cloud/Harmonizome/gene_set/Prostate+adenocarcinoma_PRAD_TCGA-HC-7738-11A-01R-2118-07/TCGA+Signatures+of+Differentially+Expressed+Genes+for+Tumors.