ICR - Identifying Age-Related Conditions

Zihao Jiang
z5352007@ad.unsw.edu.au

Lianqiang Zhao
z5396332@ad.unsw.edu.au

Supeng Wu
z5290563@ad.unsw.edu.au

Shiping Xu
z5315435@ad.unsw.edu.au

Yifan Yu
z5415202@ad.unsw.edu.au

August 4, 2023

1 Introduction

The link to the corresponding Kaggle competition for this report is: ICR - Identifying Age-Related Conditions:https://www.kaggle.com/competitions/icr-identify-age-related-conditions/overview

Due to the continuous advancement of medical technology and medical equipment, many health characteristics indicators have been created to judge the health status of each individual[1]. As these evaluation criteria increase, it has become more difficult for doctors or medical staff to comprehensively assess the relationship between measurements of specific characteristics and potential patient conditions. In addition, collecting enough information from patients is necessary, which requires a long process to make the final prediction. Therefore, machine learning methods are adapted by researchers to help identify these types of health characteristics and shorten the process as well as protect the patients' private information[2].

ICR-Identifying Age-Related Conditions on Kaggle (InVitro Cell Research 2023) is considered an excellent opportunity to apply machine learning in this context. The challenge aims to implement a group of classification models that efficiently detect conditions with measurements of anonymous characteristics, and Kaggle provided the data that needs to be encoded.

In this project, we decide to build several machine learning models to predict if a person has any of three medical conditions (Class 1) consisting of B, D, and G. Also, A is used to present that the person has none of three medical conditions (Class 0). Note that the scope of our problem is a binary classification task. We will use Precision, Recall, and F1-score as our project's evaluation metric, consistent with Kaggle's evaluation

metric.

2 Implementation

2.1 Basic Analysis of Data Set

This section is about where we started in analyzing the data, what algorithms we chose to process the data and how did we implement them. Through the first check of the data set, we have found that four datasets, train.csv, test.csv, greek.csv, sample_submission.csv respectively, are provided. Whereas the train.csv only takes 336.56kb, it represents this dataset in 58 columns, all in number except EJ, which is categorical. One important thing that we have noticed is the greeks.csv is not meaningful to the exploration of this dataset, since what included in greeks.csv are meaningless age-related conditions, which could be hard to make a connection to the train.csv. In addition to this, we also found that the data in test.csv are all 0, which makes it almost impossible to detect the abnormal situation.

2.2 Review of Suitable Learning Algorithms

Under the consideration of this competition is about computing the probability of binary classification, following other researchers' demonstration , we have chosen XGBoost, Neural Network, Random Forest, LightGBM and SVM as our alternative algorithms. we tried all of them in the model algorithm selection process and compared their performance. A description of each algorithm and their properties are summarized in Table 6 in the appendix.

2.3 Hyperparameter Adjustment

The hyperparameter adjustment of each model is very important in the experiment. In order to achieve the best performance of each model, we tested the parameters of each model separately. For the MLP model, in addition to the hyperparameters, we also adjusted the depth and width of the neural network to test the best performance under this task. The test parameters and process are shown in the table below. In model testing, we used separate test sets and cross-validation, precision, recall, and f1-score to test the performance of the mode[3].

Model	Hyperparameter	Effect	Test procedure
LightGBM+SVM	max_depth(LightGBM)	This parameter can adjust the depth of the decision tree, if the value is too large, it may cause the model to overfit, and if it is too small, it may underfit.	We tested the maximum depth of 1-5 respectively
	n_estimators(LightGBM)	This parameter will adjust the adaptive ability of the model, but will increase the training time	We tested n estimators of 100,200 and 30, respectively
	Kernel(SVM)	The kernel selection parameters of svm, which determine how to classify nonlinear problems, have a direct impact on the results	'RBF','sigmoid','poly','linear'
	C(SVM)	Regularized parameters are used to control the penalty for errors. Smaller values may cause the model to underfit, while larger values may cause the model to overfit	1,2
RandomForestRegressor	n_estimators	This parameter is to set the number of decision trees. The number of multiple decision trees can increase the performance kernel robustness of the model. An extreme number will lead to overfitting or underfitting of the model	We test when n_estimators equals 50, 100, 200, 300
XGBoost	n_estimators	The number of decision trees, the number of multiple trees can improve the performance of the model	100,500,1000,default value
	Learning rate	The learning rate and step size parameter, which is a parameter used to adjust the contribution of each tree	0.01、0.05、0.001、0.005、0.0001
Multi-Layer Perceptron(MLP)	Hidden layers neuronal width	For complex tasks, the width of neurons can enhance the model's ability to fit	For 4 deep MLP,each hidden layers width for 4,32,128,256
	Hidden layers neuronal deep	For complex tasks, adding depth allows the model to learn more complex feature representations and learn more abstract and advanced features	For each layers width equals 32,deep with 1,4,8
	Learning_rate	Each neuron backpropagation parameter adjusts the size	0.01,0.001,0.0001
	epochs	The number of times the model is trained, a larger epoch will make the model more accurate, but a larger epoch will result in overfitting, and a smaller epoch may cause the model parameters to be adjusted to the optimum	100,150,200

Figure 1: The hyperparameter adjustment of each model

2.4 Choice of Model Evaluation Metric

The evaluation metric for this Kaggle challenge is the Precision, Recall, and F-score.

Precision (positive predictive value) is a relationship between relevant elements and the selected features. The metric enables a classification model to calculate the proportion of true positive predictions to the total number of true positive and false positive predictions. Precision can be expressed using the following formula:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

Recall (sensitivity or completeness) is the fraction of selected elements among the relevant factors that measure the proportion of positive instances correctly predicted out of all the actual positive examples in the dataset. Written as a formula:

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

F-score (F-measure) is a different evaluation metric used in machine learning to assess a model's predictive capability, which combines a model's precision and recall scores, making it a popular choice in contemporary literature for model evaluation[4]. Written as a formula:

$$F{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

2.5 Algorithm selection using cross validation

We aimed to pinpoint the most effective learning algorithm for the dataset. Here, a learning algorithm refers to the combination of a learning method (such as Neural network, Random Forest or support vector machines) and the approach for choosing the best hyperparameters for each learning method (for example, utilizing a cross-validated randomized or grid search).

Cross-validation is a statistical technique used in machine learning and data analytics to assess the performance of a predictive model with less bias than traditional methods[5]. This technique is particularly useful in situations where the dataset is limited in size, as it allows for more robust analysis by utilizing all the available data.

In cross-validation, the dataset is divided into 'k' subsets or "folds". The model is then trained on 'k-1' of these subsets and tested on the remaining one. This process is repeated 'k' times, with each subset being used as the testing set once. At the end, an average performance metric is computed from the individual results

obtained from each round. This helps in reducing the risk of overfitting, as it provides a more generalized performance measure.

The most common type of cross-validation is k-fold cross-validation, where 'k' is usually set at 5 or 10. Another popular variation is leave-one-out cross-validation (LOOCV), where the model is tested on a single observation, and trained on the rest[6]. These methods allow for more robust and generalizable models by testing their performance across different sections of the data. As showed in Figure 3.

For instance, varying folds of the cross-validation loop may lead to distinct sets of optimal hyperparameters being fed to the outer loop for a given learning method. If the consequent outer cross-validation performance remains constant (meaning, different optimal hyperparameter sets for the same learning method yield similar AUCs), it indicates that our algorithm is stable and generalizes well across different datasets[7].

After identifying the best learning method and its optimal hyperparameter set, we trained the chosen model on the entire training dataset, as more data typically enhances model performance. Lastly, we assessed the performance of the model against the test set.

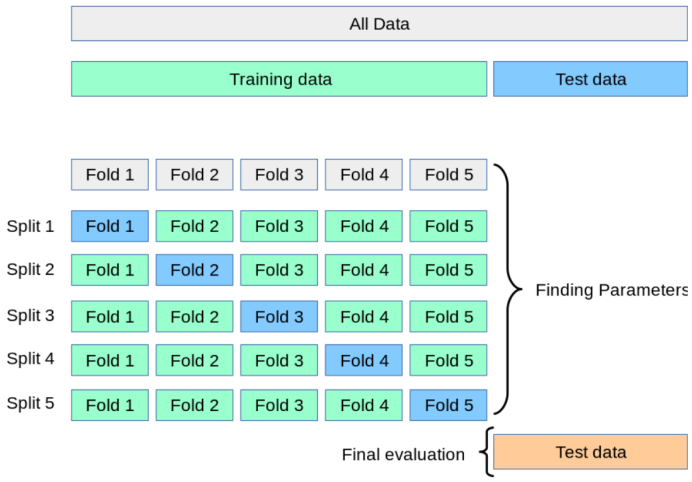


Figure 2: k-fold cross-validation

3 Experimentation

3.1 Exploratory data analysis

Kaggle provided a training and test set containing 617 and 400 observations, respectively. In addition, a collection of Greek alphabet variables (supplemental metadata) was supplied solely for the training set. Given the relatively modest size of the training dataset and the presence of missing values, we have opted for the data processing approach of filling in missing values to enhance the completeness of the training data. This decision facilitates seamless cross-validation during our analysis, as the filled data enables us to utilize the entire dataset without significant information loss due to missing values. Also, we have chosen to use Kaggle's undisclosed test set as our internal evaluation test set, ensuring consistency with leaderboard scores.

Although the training data set comprises over fifty anonymized health characteristics linked to three age-related conditions, a significant point is that the correlation between each data feature exerts varying degrees of impact on the prediction results. Our research strategy involves adopting data preprocessing techniques (such as one hot encoding or normalization) to enhance the dataset's quality. We aim to optimize the information available to our model (such as SVM or neural network model) through systematic data processing, leading to improved predictive performance and more robust predictions.

3.2 parameter performance of each model

Table 1: Marco F1-score

method	**precision**	**recall**	**F1-score**	**training accuracy**	**cross val accuracy**	**support**
lgbm	N/A	1.0	1.0	1	0.94	N/A
svm	N/A	0.676	0.798	0.94	0.89	N/A
MLP prediction	0.88	0.81	0.84	N/A	N/A	88
MLP cross-validation	0.99	0.99	0.99	N/A	0.99	548
XGBOOST	0.85	0.70	0.77	1	0.93	N/A
Randomforest	0.93	0.5	0.65	0.92	0.92	N/A

The primary objective of this outer loop was to find the best algorithm that demonstrates robust generalization on diverse data and yields consistent AUC scores. To evaluate this, we examined the average AUC from the 5-fold cross-validation. Ideally, we aimed to achieve a high average AUC, indicating effective generalization during the model fitting and hyperparameter selection process in the inner cross-validation loop.

Our research results indicate that the mlp algorithm is the optimal choice among all tested algorithms. It consistently obtained the highest AUC and, in most cases, showed the lowest standard deviation, suggesting

its stability and resistance to overfitting or noisy data. However, the performance of other models was still quite good, albeit with slightly lower AUCs.

Based on these findings, we decided to adopt the mlp algorithm, along with its associated hyperparameter selection process, for the main data analysis process. Subsequently, a comprehensive exposition encompassing the background, model elucidation, and meticulous data analysis pertaining to the MLP (Multi-Layer Perceptron) methodology will be presented.

3.3 Multi-Layer Perceptron

For the Multilayer Perceptron (MLP), a type of neural network, we constructed four MLP structures[8]. Each MLP uses the ReLU activation function $f(x) = \max(0, x)$ for each hidden layer, which avoids gradient saturation and can prevent problems such as vanishing gradients. The output layer uses the sigmoid activation function $f(x) = 1/(1 + \exp(-x))$. As an S-shaped curve function, it can well represent the probability of a sample. We also tried to add a dropout rate in the model to prevent each neuron from being overly dependent on other neurons.

- **MLP1**:
  - Input Layer: Each input data has 57 dimensions.
  - Hidden Layer: 1 hidden layer, each layer has 56 neurons, using ReLU activation function.
  - Output Layer: 1 output layer, 1 neuron, using sigmoid activation function.
- **MLP2**:
  - Input Layer: Each input data has 57 dimensions.
  - Hidden Layer: 1 hidden layer, each layer has 256 neurons, using ReLU activation function.
  - Output Layer: 1 output layer, 1 neuron, using sigmoid activation function.
- **MLP3**:
  - Input Layer: Each input data has 57 dimensions.
  - Hidden Layer: 2 hidden layers, each layer has 128 neurons, using ReLU activation function.
  - Output Layer: 1 output layer, 1 neuron, using sigmoid activation function.
- **MLP4**:
  - Input Layer: Each input data has 57 dimensions.

– Hidden Layer: 3 hidden layers, the first layer adds dropout rate of 0.25, 512 neurons. The last two layers have 256 neurons, using ReLU activation function.

– Output Layer: 1 output layer, 1 neuron, using sigmoid activation function.

The structures of the four MLPs are shown in the figure below .

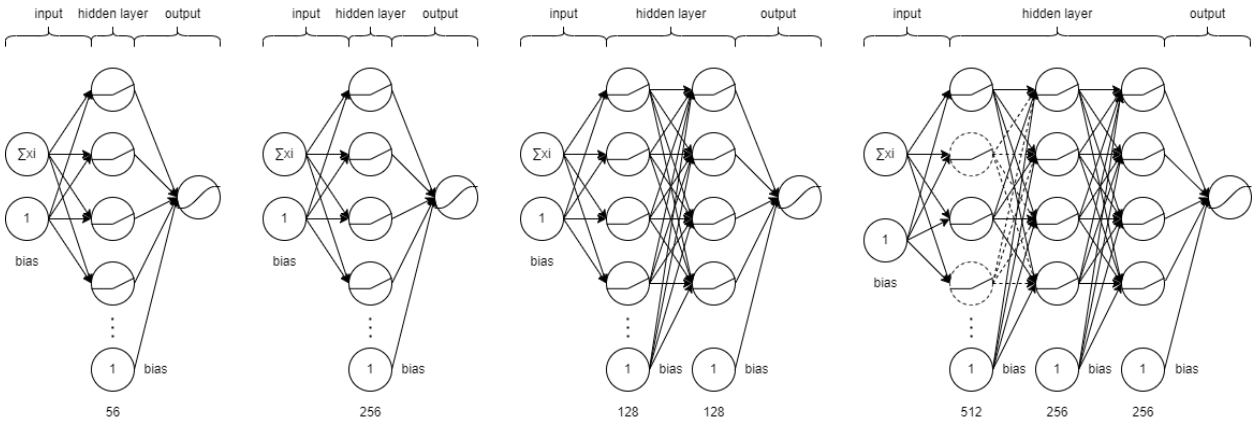


Figure 3: structure of MLP

To prevent overfitting, we set EarlyStopping as callbacks and set the minimum change value to 0.01. The result curves of the four models are shown in the figure below .

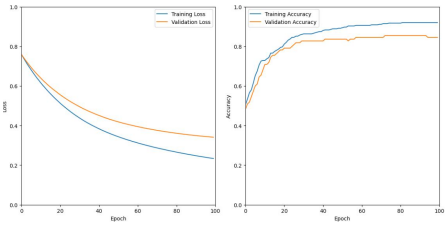


Figure 4: MLP1

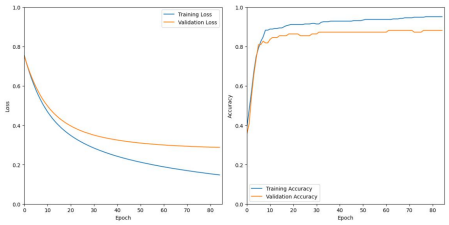


Figure 5: MLP2

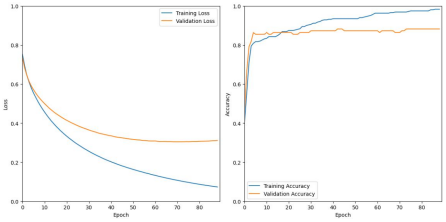


Figure 6: MLP3

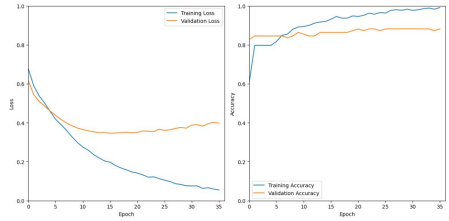


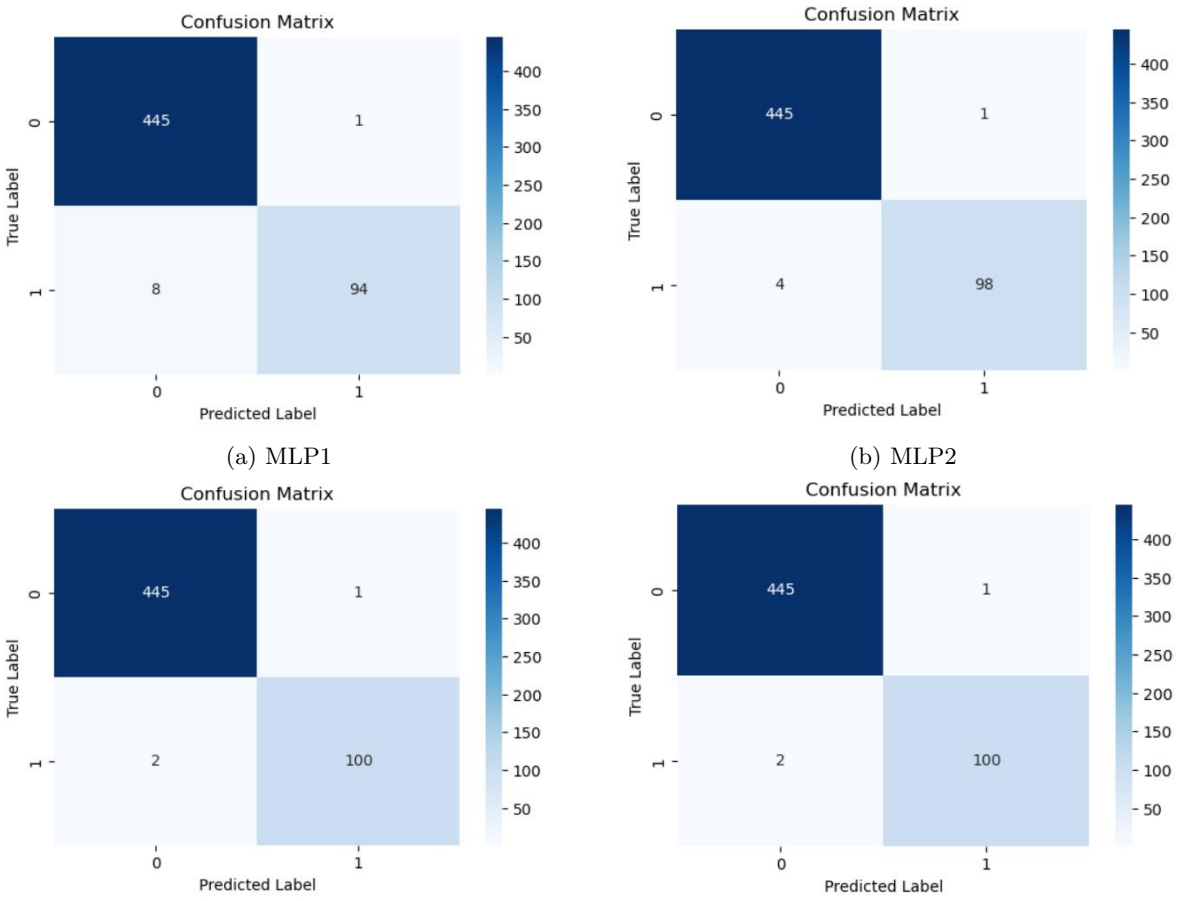
Figure 7: MLP4

Figure 8: result curves

We also used the method of cross-validation, set to fold 20 times, and calculated the Precision, Recall, F1-score, accuracy and confusion matrix of each of the four models. The results are shown in the table below , and the confusion matrix of the model is shown in the figure below .

MLP model		Precision	Recall	F1-score	support
MLP1	0	0.98	1.00	0.99	446
	1	0.99	0.92	0.95	102
	accuracy			0.98	548
	Macro avg	0.99	0.96	0.97	548
	Weighted avg	0.98	0.98	0.98	548
MLP2	0	0.99	1.00	0.99	446
	1	0.99	0.96	0.98	102
	accuracy			0.99	548
	Macro avg	0.99	0.98	0.98	548
	Weighted avg	0.99	0.99	0.99	548
MLP3	0	1.00	1.00	1.00	446
	1	0.99	0.98	0.99	102
	accuracy			0.99	548
	Macro avg	0.99	0.99	0.99	548
	Weighted avg	0.99	0.99	0.99	548
MLP4	0	1.00	1.00	1.00	446
	1	0.99	0.98	0.99	102
	accuracy			0.99	548
	Macro avg	0.99	0.99	0.99	548
	Weighted avg	0.99	0.99	0.99	548

Figure 9: mlp performance



(a) MLP1

(b) MLP2

(c) MLP3

(d) MLP4

Figure 10: confusion matrix

3.4 model evaluation results

After selecting the best learning method and optimal set of hyperparameters for each label, we then fit our final models and evaluated their performance on the test set.

In terms of why NN behaved better than the others, we came up with two points that could probably resulting in its trustable performance. Firstly, neural network has multiple layers in structure, which equip it with stronger ability to learn and represent complicated model, whereas others, such as SVM or random forest, might not as they be designed to separate linear data for example [9]. Secondly, compared with manually feature extraction, neural network is able to learn feature automatically through backward propagation, which enable neural network to be more effective when dealing with data high dimension and complicated structure[10].

Figure 6 provides a summary of the final models in four hyperparameters for each category. From overall observation, MLP3 and MLP4 behaves slightly better than the others. Additionally, from figure 5, we can tell that the loss of MLP4 could be a little overfitting and unstable, whereas it is smoother for MLP3 [11].

4 Discussion

4.1 Model Comparison

In this project, we experimented with five models, namely LightGBM, SVM, MLP, XGBoost, and Random Forest. We selected MLP, which performed best in this project, as our final model.

SVM is a powerful classifier, especially suitable for dealing with unbalanced and high-dimensional data through a Kernel function. However, it requires more training time when dealing with large-scale data, especially for complicated Kernel functions.

LightGBM is an ensemble learning algorithm that constructs a strong classifier by assembling multiple weak classifiers, most commonly decision trees. LightGBM performs exceptionally well when dealing with large-scale data and complex tasks, although it is limited in handling unbalanced data[12].

XGBoost belongs to the Gradient Boosting Decision Tree category, similar to LightGBM. However, it employs a more precise greedy algorithm to generate trees, resulting in higher computational costs.

MLP is a model based on Neural Networks, which enables it to deal with sophisticated non-linear problems and adapt to various types of data. But typically, MLP requires relatively long training data, especially when deep layers are needed, and pre-processes data using standardization and normalization.

Their performance is summarized in Table 1.

4.2 Metrics

The F1 score is a measure of accuracy that takes the harmonic mean of precision and recall. It is a value between 0 and 1, with 1 being the best value. The F1 score is particularly useful when you want to balance precision and recall, i.e., the identification of true positives while considering the misclassification of negatives.

Cross-validation is a technique used to evaluate model performance. The dataset is typically divided into k parts, and then trained on $k-1$ parts and tested on the remaining part. This process is repeated k times, with each part serving as the test set once. The purpose of cross-validation is to reduce the model's reliance on a particular train-test split of the dataset, providing a more accurate estimation of how the model performs on unseen data[13].

Accuracy is a performance metric for classification models. It calculates the ratio of correctly classified observations to the total number of observations. While it's a good measure in many situations, it can be misleading in cases of class imbalance.

Recall, or the true positive rate, is a measure of how many actual positives a model has found. In some applications, finding all the positives may be more important than finding positives precisely.

Cross-validation doesn't directly compare with the other metrics mentioned, as it's a technique for evaluating model performance, not a performance metric itself. However, the strength of cross-validation lies in providing a more robust and accurate estimation of model performance. By repeatedly evaluating the model on multiple train-test splits, it minimizes the variance in the evaluation, more accurately reflecting how the model is expected to generalize to unseen data[14]. This is an advantage over using a single train-test split, which may introduce noise and bias due to random sampling.

In summary, cross-validation is considered superior because it provides a way to reduce evaluation error and bias, more accurately reflecting the expected performance of the model. Based on the comparison between the chosen metrics, cross-validation was selected to assess the model's suitability. And MLP behaves the best performance hence the final model in this project.

5 Conclusion

In this project, we tested several learning algorithms as part of ICR-Identifying Age-Related Conditions to classify whether a person has any of three medical conditions.

As for algorithms, our project contains six learning methods, including Multilayer Perceptron (MLP), Random Forest (RF), Decision Tree (DT), eXtreme Gradient Boosting (XGBoost), Light Gradient Boosting Machine (LGBM), and Support Vector Machine (SVM). The six learning algorithms adopted are comparable, with MLP performing best in predicting the patient's medical conditions. The evaluation metrics of MLP show that the Precision, Recall, and F1-score all reached 0.99, which would have placed our team in a

relatively high-ranking position. Cross-validation was selected to assess the models' suitability by comparing the chosen metrics. It enables our project to get a more accurate estimation of different model performances with multiple train-test splits, which can solve the issue of random samples.

From the whole project, our models achieved their goal of accurately predicting potential patient conditions. In addition, it is rather convenient for us to superimpose or modify the model because of the given feature extraction, parameter fine-tuning, etc. Considering these advantages, MLP was constructed with four layers (MLP1, MLP2, MLP3, MLP4) and adjusted the parameter settings of the hidden layer, leading to our final expected result.

6 Future Work

For feature engineering, time-series analysis could be implemented to create additional features, allowing the model to learn more in-depth advanced features. In terms of the model, ensemble learning could be utilized, combining models and assigning specific weights to them. These weights could then be adjusted during the learning process to enhance the model's generalization ability. For the MLP model, more complex neural networks could be explored to understand the impact and role of each neuron in the hidden layers on the overall results.

References

[1] Liang Tan et al. "Toward real-time and efficient cardiovascular monitoring for COVID-19 patients by 5G-enabled wearable medical devices: A deep learning approach". In: *Neural Computing and Applications* (2021), pp. 1–14.

[2] Mohammad Shehab et al. "Machine learning in medical applications: A review of state-of-the-art methods". In: *Computers in Biology and Medicine* 145 (2022), p. 105458.

[3] Muhammad Adnan et al. "Utilizing grid search cross-validation with adaptive boosting for augmenting performance of machine learning models". In: *PeerJ Computer Science* 8 (2022), e803.

[4] Dominik Müller, Iñaki Soto-Rey, and Frank Kramer. "Towards a guideline for evaluation metrics in medical image segmentation". In: *BMC Research Notes* 15.1 (2022), pp. 1–8.

[5] Patrick Schratz et al. "Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data". In: *Ecological Modelling* 406 (2019), pp. 109–120.

[6] Daniel Berrar et al. *Cross-Validation.* 2019.

[7] Jacques Wainer and Gavin Cawley. "Nested cross-validation when selecting classifiers is overzealous for most practical applications". In: *Expert Systems with Applications* 182 (2021), p. 115222.

[8] Jinghua Zhang et al. "Applications of artificial neural networks in microorganism image analysis: a comprehensive review from conventional multilayer perceptron to popular convolutional neural network and potential visual transformer". In: *Artificial Intelligence Review* 56.2 (2023), pp. 1013–1070.

[9] A. S. More and D. P. Rana. "Review of random forest classification techniques to resolve data imbalance". In: *2017 1st International Conference on Intelligent Systems and Information Management (ICISIM)*. IEEE. 2017, pp. 72–78. DOI: 10.1109/ICISIM.2017.8122151.

[10] Laith Alzubaidi et al. "DFU_QUTNet: diabetic foot ulcer classification using novel deep convolutional neural network". In: *Multimedia Tools and Applications* 79.21-22 (2020), pp. 15655–15677.

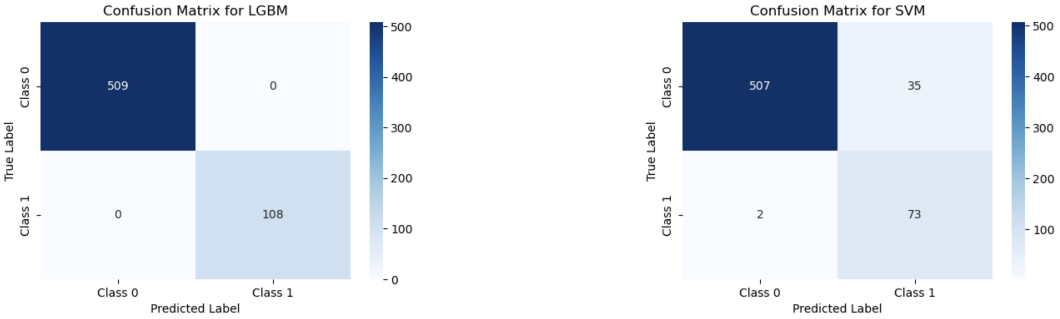
[11] L. Jiang et al. "Prediction of coronary heart disease in gout patients using machine learning models". In: *Math Biosci Eng* 20.3 (2023), pp. 4574–4591. DOI: 10.3934/mbe.2023212. eprint: 2022.12.27.

[12] Prerna Singh. "Systematic review of data-centric approaches in artificial intelligence and machine learning". In: *Data Science and Management* (2023).

[13] Stephen Bates, Trevor Hastie, and Robert Tibshirani. "Cross-validation: what does it estimate and how well does it do it?" In: *Journal of the American Statistical Association* (2023), pp. 1–12.

[14] Han-Shin Jo et al. "Path loss prediction based on machine learning techniques: Principal component analysis, artificial neural network, and Gaussian process". In: *Sensors* 20.7 (2020), p. 1927.

Appendix



(a) Confusion Martix for LGBM

(b) Confusion Martix for SVM

Figure 11: Confusion Martix

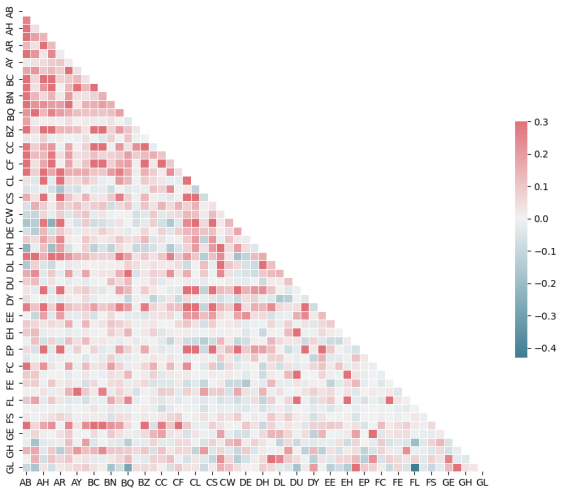


Figure 12: mlp correlation martix

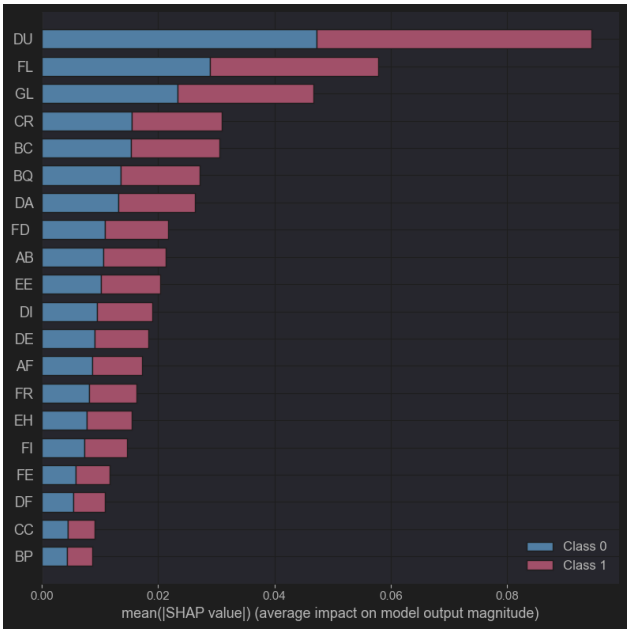


Figure 13: output magnitude

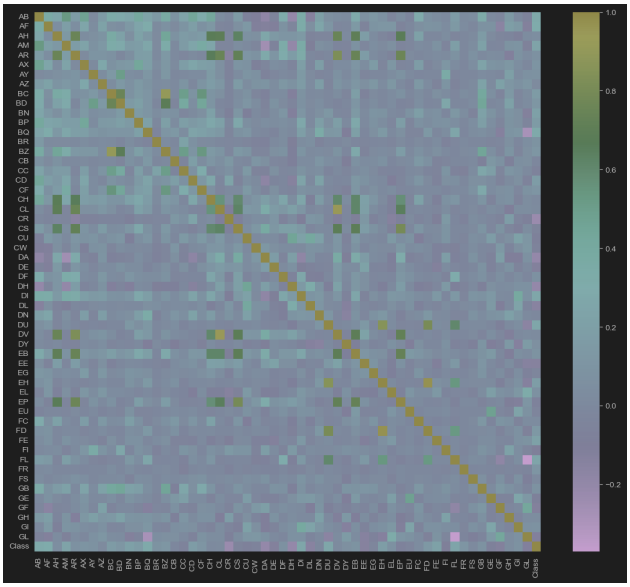


Figure 14: randomforest correlation martix