Machines Learning Models
On Mortality Prediction and Length-of-stay Prediction

Abstract

Six Machine Learning / Deep Learning models from four categories are used to predict mortality and length-of-stay from the MIMIC-III database extraction, attaining 0.91 AUC and 1.44 RMSE by XGBoost and Multilayer Perceptron model. This report compares the conducted models (including individual models and ensembled models), and then it illustrates the features that may be biomedically important from the model feature selection. Finally, a website with these two optimal models built in have been opened, allowing the medical system to access the model prediction and obtain treatment insights by easily adjusting and inputting the bio-feature data.[1]

1. Introduction

Electronic Health Records (EHRs) can improve healthcare in many ways, such as delivering better patient treatments, improving hospital operations, and answering fundamental scientific questions. For example, the Hong Kong government has developed an electronic Health Record Sharing System (eHealth). It provides an electronic platform that could store lifelong electronic health records for all members of the public. It enables authorized healthcare-providing organizations in public and private sectors to access and share participating patients' electronic health records for healthcare purposes. Then government departments and individual healthcare providers, such as the Hospital Authority (HA), the Department of Health (DH), and individual private providers authorized by the patient, can access the patient's EHR on a need-to-know basis while providing healthcare. This system can enable timely diagnosis and treatment and reduce duplicate diagnostic tests.

The development of healthcare systems means that we have increasingly more available EHRs, but at the same time, processing and interpreting these EHRs manually is causing difficulties. Previous research implies that machine learning (ML) plays a role in such circumstances. It can efficiently use large amounts of data to make analyses and predictions. For example, it can detect diseases early, predict living states with high accuracy, and generate an accurate Length of Stay (LOS) prediction for better-allocating hospital resources. An accurate prediction can be beneficial as the occupancy rate of ICU beds is always high in Hong Kong.

Therefore, in this research, we aim to analyze the extraction from the public database MIMIC-III data through ML methods to build and train models that can accurately predict the mortality and LOS of patients in the intensive care unit (ICU)' mortality and LOS. Regarding deep learning (DL) as a further version of ML, we trained different types of both ML and DL models, including parametric regression, kernel methods, tree-based methods, and neural networks. After trying to train individual models, we also used the stacking technique to do a model ensemble to improve the predictions' accuracy. Finally, by comparing the results of different modeling methods, we also interpreted the models and tried to provide some prior ideas for

[1] Website: Mortality probability prediction

medical research from the perspective of big data.

2. Literature Review

2.1 Importance of using Machine Learning on EHR

While enjoying the availability of data from EHRs, the large number of EHRs also creates difficulties in using, processing, and interpreting them. Gunčar et al. (2018) described the process of manually interpreting the blood test EHRs as even the most professional hematologist may ignore trends, deviations, and interactions between the growing number of blood parameters measured by contemporary laboratories.

In contrast, according to Chan (2019), ML algorithms can readily manage hundreds of features (parameters) and can exploit the relationships between these features, which makes the medical profession especially attractive for ML applications.

2.2 Previous Machine Learning examples

ML can detect diseases early and predict living states with high accuracy. Gunčar et al. (2018) applied three models to predict the haematologic disease: Support Vector Machine, Naïve Bayesian classifier, and XGBoost. Within them, the XGBoost model obtained an accuracy of more than 0.86. Logistic regression can also be used to conduct mortality prediction (Taslimitehrani et al., 2016). Ding et al. (2021) used Artificial Neural Networks Model to predict in-hospital mortality in acute pancreatitis in MIMIC-III early and achieved 0.796 AUC.

An accurate Length of Stay (LOS) prediction is essential for hospitals allocating resources. Adilabad et al. (2022) obtained an accuracy of 94.16% using the random forest model when predicting the LOS in the ICU, which is much higher than the traditional human brain prediction.

2.3 Previous Novel Work and Our Strategy

2.3.1 Prior Biological knowledge for LOS prediction

Harutyunyan et al. (2019) chose 17 clinical features that are easily assessed throughout the population and have important biological implications for predicting the LOS for adult ICU patients. Straathof (2020) selected 11 of these 17 features to predict the LOS of Neonatal intensive care unit (NICU) patients. The six features that were not selected were either due to their low biological significance in the NICU or insufficient data in that research. By considering the biological meaning and the data sufficiency, three more features were added, and one feature was removed. The set of a total of 13 features (Table 1) would be kept for task 2 modeling.

Feature Name	Illustration
pulmonary capillary wedge pressure (PCWP)	Increased PCWP can reflect increased left atrial pressure, such as acute pulmonary edema; when PCWP is lower than normal, it reflects hypovolemia.
diastolic blood pressure	The force of blood against the artery walls while the heart relaxes.
systolic blood pressure	The force of blood against the artery walls while the heart contracts.
fraction inspired oxygen	the percentage or oxygen in the air that a person inhale.
oxygen saturation	the extent to which hemoglobin in the blood is saturated with oxygen.
glascow coma scale total (GCS)	A medical method to evaluate the degree of coma of a patient. GCS has three aspects: eye-opening response, language response and body movement. The sum of the scores of the three aspects is the coma index.
height	Patient's height
heart rate	Patient's heart rate
glucose	Blood sugar concentration
weight	Patient's weight
respiratory rate	Patient's respiratory rate
temperature	Patient's body temperature
pH (of the blood)	Patient's blood pH

Table 1. A set of 13 features kept for task 2 prediction

2.3.2 Web-based Application

After training models to predict haematologic disease, Gunčar et al. (2018) also developed a web-based application that enables the easy input of blood parameters and produces a prediction and visualization of haematologic disease results[2], which increases the practical significance of model utilization.

In our project, we also deploy the best two models for the two tasks on a website. When the hospital system imports the corresponding patient bio-feature data into the website, the website will display the prediction results. Another important reason for constructing the website is that, through the website, medical personnel can predict how a patient's mortality or LOS would change if a particular biological feature changes. For example, they would learn how the dependent variable changes if the patient's blood glucose concentration is increased while keeping other signs constant simply by changing the input of blood glucose concentration on the website. Therefore, the website can give medical staff treatment suggestions based on the model. Such applications also provide new ideas for utilizing other EHRs.

[2] It's available on www.smartbloodanalytics.com

3. Data Processing

3.1 Data Set and Extraction

MIMIC-Extract uses the publicly accessible MIMIC-III database's raw EHR data for patients in critical care, then creates data frames that can be used directly in standard ML workflows. In MIMIC-Extract, there are 104 bio-features in total, which are related to blood, heart, lungs, and some basic physical measures (Table 2).

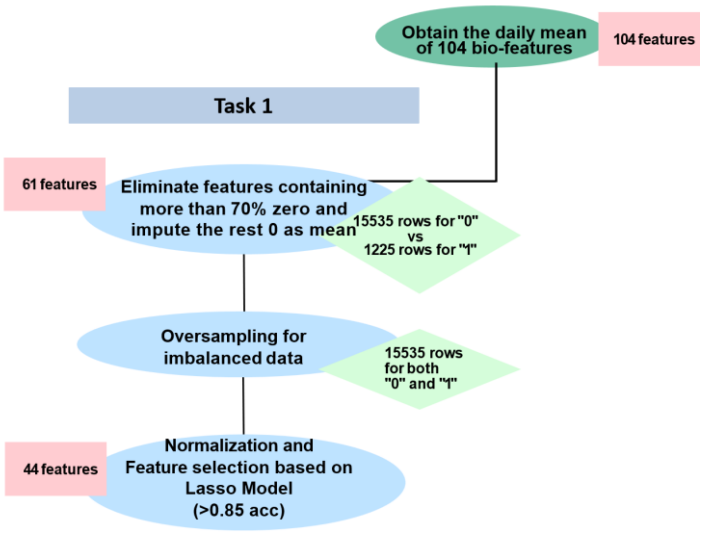
Blood	Heart	Lung	Other basic characteristics
albumin pleural albumin urine red blood cell count ascites red blood cell count csf platelets total protein bilirubin Cholesterol...	cardiac index cardiac output fick cardiac output thermodilution heart rate oxygen saturation partial pressure of carbon dioxide partial pressure of oxygen ...	respiratory rate respiratory rate set tidal volume set tidal volume spontaneous positive end-expiratory pressure positive end-expiratory pressure set pulmonary artery pressure mean ...	temperature alkaline phosphate anion gap basophils calcium urine chloride Co2 ...

Table 2. Feature Samples in different categories

This dataset includes the hourly average of each patient, and each bio-feature, during 24 hours. We utilized the mean hourly average for each patient and each bio-feature. Namely, we got 104 columns. The reason why other statistics are not used is that: since more than 80% of the hourly averages are not detected, which implies a huge amount of missing values. Therefore, it is not suitable to use its temporal properties for modeling. Furthermore, since more than half of the samples' bio-feature records do not contain more than two data points, therefore the data trend does not exist. Utilizing minimum and maximum will lose much information compared with the raw data. Finally, the mean of the hourly average was calculated and used.

3.2 Data Pre-processing for Mortality Prediction

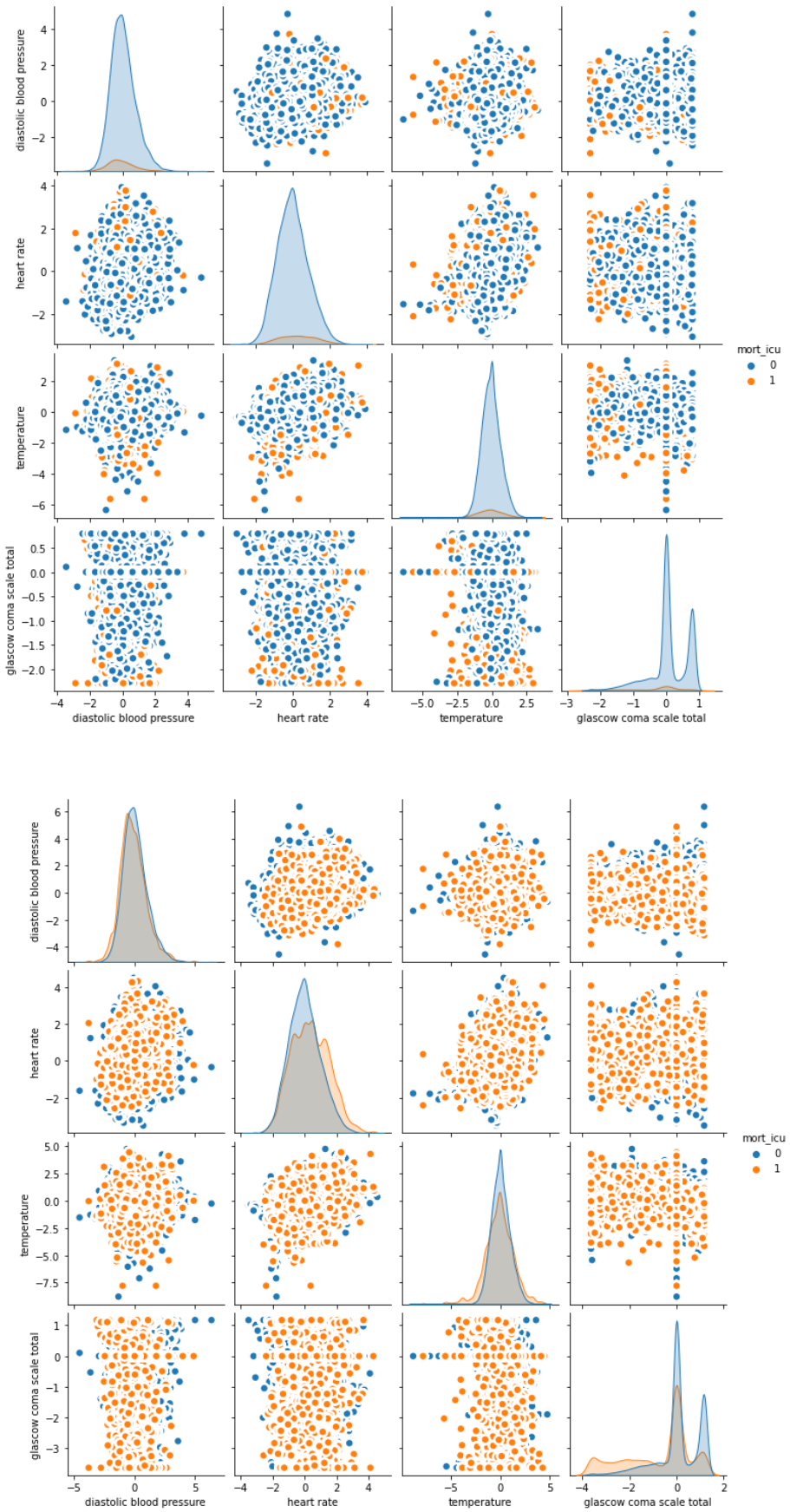
Further data preprocessing is different for Mortality and LOS prediction. Pic 1 shows the thorough flow of pre-processing for predicting mortality.



Pic 1. Data Pre-processing for Mortality Prediction

For Mortality prediction, we remove the feature with more than 70% zero value. feature size shrinks from 104 to 61 in this step. For the rest of the 61 features, we impute the calculated mean to substitute the zeros.

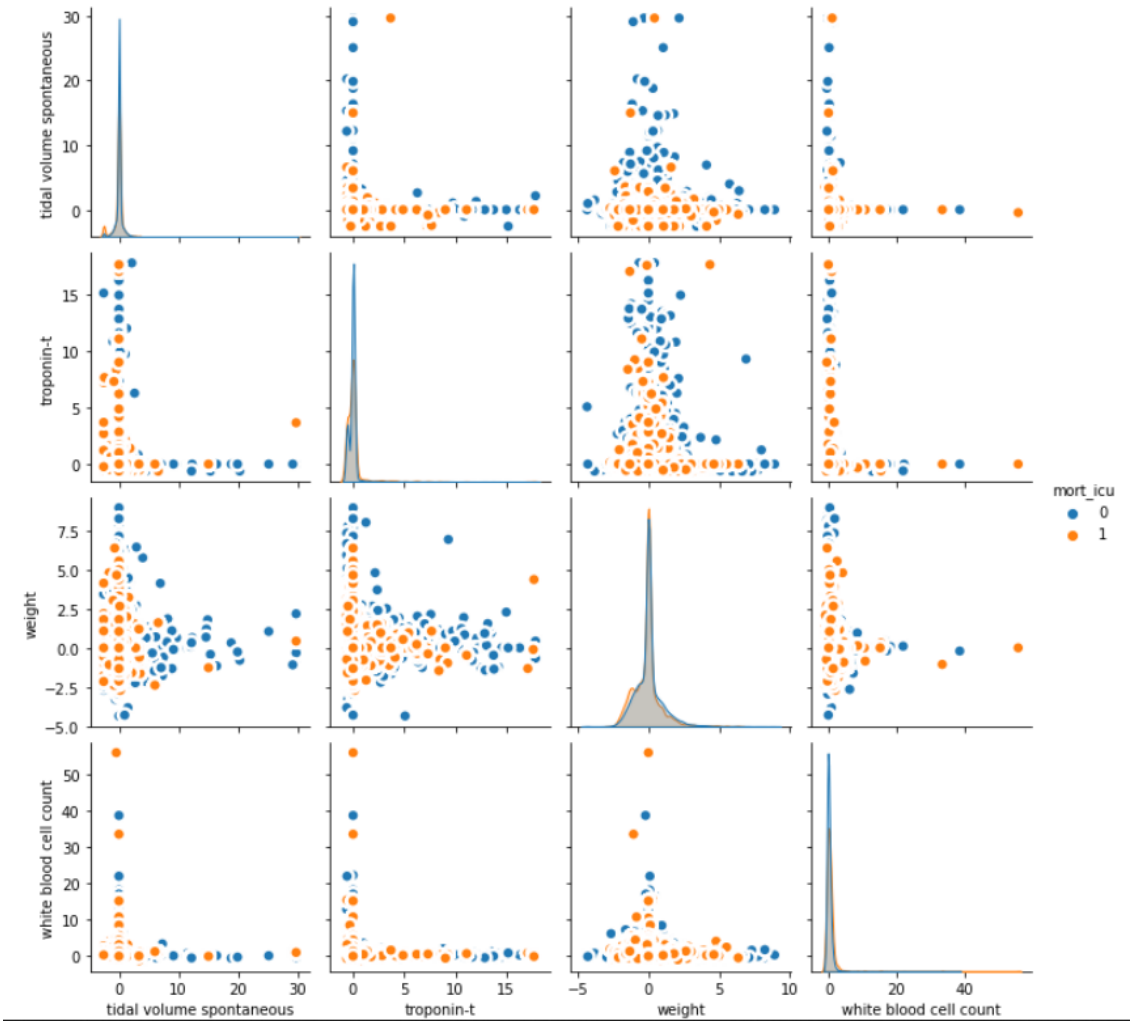
We observed that in train and valid dependent variables, '0' shows much more frequently than '1' (with 15535 and 1225 rows, respectively). Then, we used oversampling in imbalanced learning to generate more samples with an output of '1', hence decreasing the model's bias. The following pair plots can show how oversampling helps to generate balanced data.



Pic 2. Pair plot comparison between features before (upper) and after (lower) oversampling

The upper and lower graphs respectively represent the feature interactions before and after oversampling. Among them, the small picture on the diagonal indicates the distribution of the variable on mortality. From the above figure, we can see that the blue line is always much higher than the orange line. In the scatter plot next to it, we can also intuitively see that the blue points are far more than or even cover the orange points, which both imply that the data is very unbalanced. When the machine model sees more points in the binary classification, it will tend to directly predict many results into this category. This can indeed improve some accuracy rates, but it loses the meaning of modeling prediction.

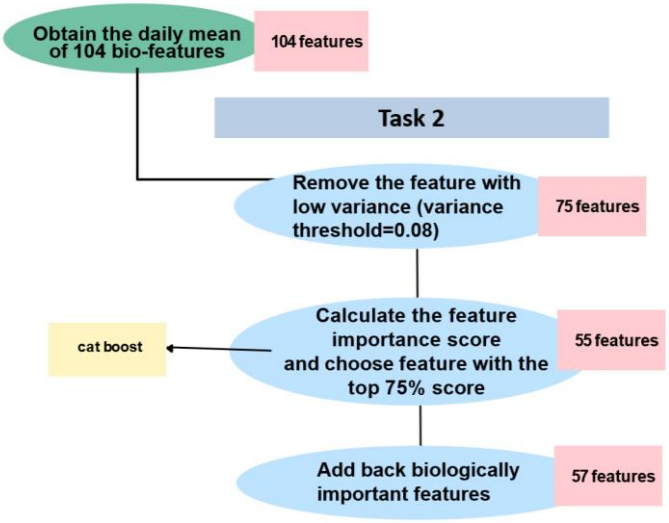
Finally, we further selected the feature by using the lasso model since the lasso model had a satisfying fitting to the task I data (around 85% accuracy). By comparing pic 3 (features excluded by lasso selection) and the lower graph of pic1, we can see that lasso selection excludes some features with more extreme outliers. This may be because these features are not concentrated in one area, so their impact on the results is not concentrated. Another reason might be that their predictions are quite different from the predictions of most other features, which interferes with the prediction of the model. Therefore, the model removes them, leaving some variables with a more concentrated distribution and fewer extreme outliers.



Pic 3. Pair plot of features removed by lasso feature selection

In the above steps, we simplify our feature to 44 for the Mortality prediction, and we will use the selected feature subset to fit Elastic net regression, SVM, XGBoost, and MLP, which will be explained in chapter 4.

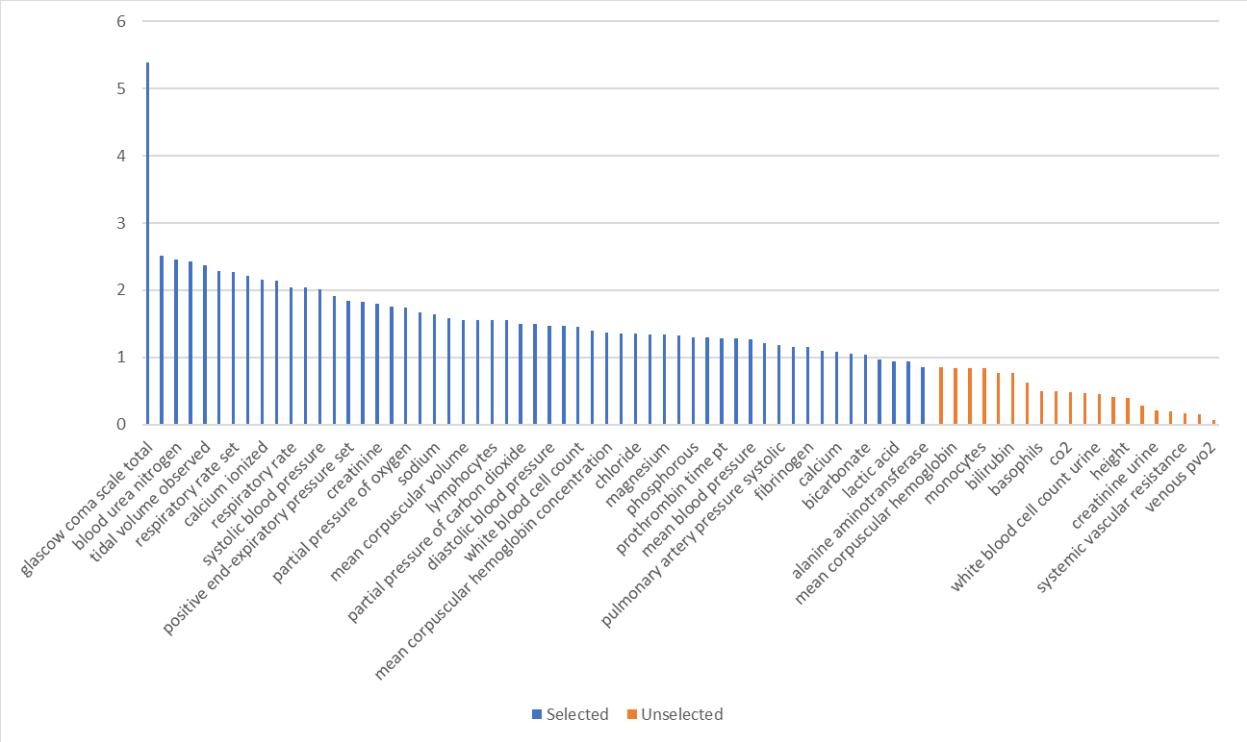
3.3 Data Pre-processing for loss Prediction



Pic 4. Data Pre-processing for LOS Prediction

For LOS prediction, we first remove the feature with low variance. If a feature has too small a variance across all samples, it will not contribute much to the model's predictions. To ensure the remaining features are biomedically important to the LOS prediction, we set the variance threshold at 0.08. In this step, we had 74 features left.

To further process the data, we calculated the feature importance score based on CatBoost (after the model ensemble mentioned in 4.1.7), and we extracted the feature with the top 70% score as the feature subset (Pic 5). The reason why we don’t use lasso selection as in the mortality task is that for this regression problem, lasso selection should be based on linear regression, which only fits around 15% of our training data. The model does not fit well implies the features selected cannot be reasonable.



Pic 5. Feature Selection by CatBoost

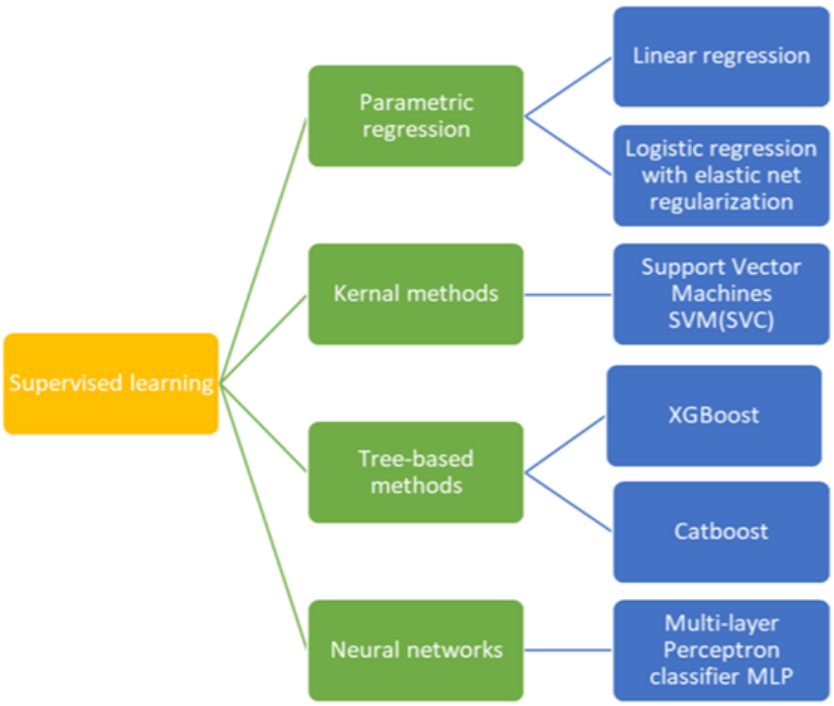
From this feature selection, we found that **'The Glasgow coma scale total', 'blood urea nitrogen, 'tidal volume observed', 'respiratory rate set', and 'calcium lionized** are viewed as the five most important bio-features regarding predicting the LOS.

As mentioned in 2.3.1 and Table 1, 13 features have important biomedical meaning in estimating the LOS. We found that two of the 13 important variables were not included in the set selected by Catboost, which are "pulmonary capillary wedge pressure" and "height", so we added them to improve the interpretability of the model. Hence, there are 57 features left for multilayer perceptron, cat boost, elastic-net regression, and logistic regression. In chapter 4, we would see how these models predict LOS based on these features.

4. Methodology

4.1 Baseline Models

In this study, we adopt six models from four commonly used supervised ML algorithms: parametric regression, Kernel methods, Tree-based methods, and Neutral networks to find the most suitable model to achieve our two goals.



Pic 6. Model used categorization

In this part, we'll briefly introduce the six models we've selected as follows:

4.1.1 Multiple Linear Regression

Linear regression analysis is used to predict the dependent variable based on one or more independent variables. The multiple linear regression can be defined with the formula (Tranmer & Elliot, $Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_i x_i + \varepsilon_i$

Although Linear regression analysis is a relatively simple model which is easy to interpret, it can provide enormous flexibility, which can be applied to various circumstances.

Elastic net regression is a regularized regression linearly combining both L_1 and L_2regularizations. The estimates from such a regression model can be defined by:

$$\min_{\beta}(y - X\beta)^T (y - X\beta) + \lambda\left(\alpha||\beta||_{l1} + \frac{(1-\alpha)||\beta||_{l_2}^2}{2}\right)$$

It is useful when multiple features are correlated with each other.

4.1.2 Logistic Regression

The logistic model has been the most used model for binary regression since about 1970 (Cramer, 2002). It is widely used in medical fields to predict the risk of developing a given disease based on the observed characteristics of the patient.

The logistic model takes the form $p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$ to ensure the estimation of p(X) lies between 0 and 1. Here we used the logistic regression with elastic net regularization, a regularized regression method that linearly combines the L_1 and L_2 penalties.

4.1.3 Support Vector Machines (SVM)

Support vector machines (SVMs) are proposed by Boser, Guyon, and Vapnik (1992). The objective of SVMs is to find a hyper-plane *(w,b)* with maximized quantity, $\gamma = \min_i \; y^i\{\langle w, \varphi(x^i)\rangle - b\}$where $\langle w, \varphi(x^i)\rangle$ indicates an inner product, y represents margin and the quantity $\langle w, \varphi(x^i)\rangle - b$ correlates with the distance between the data point x^i and the decision boundary (Furey et al., 2000).

SVMs provide significant accuracy as the prediction error is minimized by finding a hyperplane with maximized quantity, so it is effective in high dimensional spaces. Moreover, complex relationships between data points can be obtained without performing difficult transformations due to the kernel trick used in the SVMs model.

4.1.4 XGBoost

XGBoost is a scalable ML technique for gradient tree boosting. Chen and Guestrin (2016) made their improvements in the regularised objective of XGBoost by the modified formula as follows,

$L\,(t) = \Sigma i{=}1\; n\; l\,(yi\,,\, \hat{y}\,(t{-}1) + ft\,(xi\,) + \Omega(ft\,))$

where $\hat{y}_i^{(t-1)}$ indicates the prediction of the i-th instance at the t-th iteration and l represents the differentiable convex loss function which computes the difference between the prediction and the target variable Yi. We choose the XGBoost model to represent the tree-based method because that it always gives more importance to functional space when reducing the cost of a model while Random Forest tries to give more preferences to hyperparameters to optimize the model.

4.1.5 Multi-layer Perceptron classifier (MLP)

Multi-layer Perceptron classifier (MLP) is a fully connected feedforward artificial neural network (ANN) class. It can distinguish data that is not linearly separable.

At first, we considered using recurrent neural network (RNN), a particular type of artificial neural network adapted to work for time series data or data that involves sequences to complete the task. However, we found that because of the sparsity of the data sets we study. There is not enough data to support RNN to generate good performance. Therefore, we also decided to integrate the 24-hour data into a mean, then the model was also reduced into MLP.

4.1.6 boost and Other Models for Model Ensemble

Model	Illustration
AdaBoost	It is based on the idea of boosting. A strong classifier is obtained through the linear combination of multiple weak classifiers. During training, it focuses on misclassified samples, and the weight of the weak classifier with high accuracy is large. (Wikipedia contributors, 2022)

Gradient Boost	It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees.
Random Forest	It is an ensemble learning method for classification and regression that operates by constructing a multitude of decision trees at training time.
LightGBM	LightGBM is a distributed and efficient framework for implementing the GBDT algorithm. It uses the leaf-wise splitting method to generate a decision tree, finds feature segmentation points through a histogram-based algorithm, and supports parallel learning. (Wikipedia contributors, 2022)
CatBoost	The biggest feature of CatBoost is that it can directly process category features without any pre-processing to convert categories to numbers.
KNN	K-nearest neighbours algorithm (KNN) is a non-parametric supervised learning method, the input consists of the *k* closest training examples in a data set. The output depends on whether *k*-NN is used for classification or regression. (Wikipedia contributors, 2022)

Table 3. Models for Model Ensemble

These models are commonly used and can complement each other, but lack of literature to prove that they have a significant effect on our task. Therefore, we used a model resemble containing these six models, along with XGBoost, which can algorithmically make the tree-based model training better, to train the data. We also used Bayesian optimization to generate parameters aiming to get the best outputs of the seven models. Finally, we chose CatBoost to predict on the test set.

4.2 Choices on Each Task

4.2.1. Task 1 Mortality Prediction -- Classification Problem

For task 1, predicting the mortality of patients is a classification problem. The outputs are categorized into two categories, 0 and 1 only. We chose four models: Logistic regression, Support Vector Machines (SVM), XGBoost, and Multi-layer Perceptron classifier (MLP).

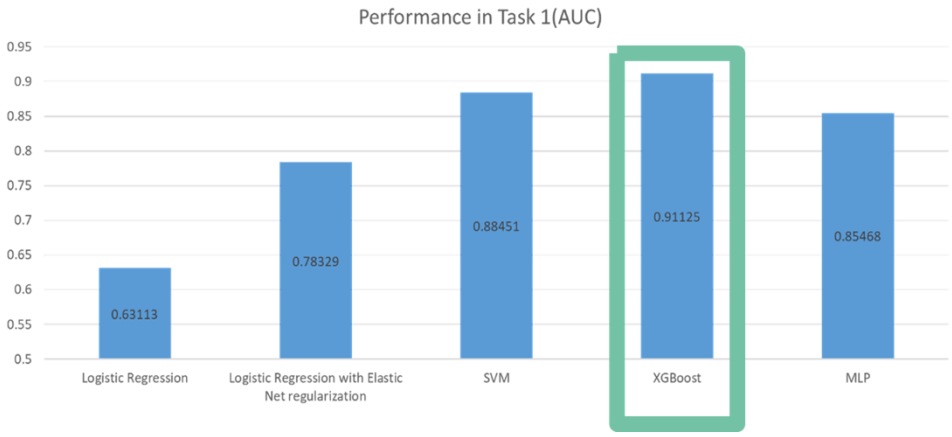
4.2.2. Task 2 Length of Stay (LOS)Prediction – Regression Problem

For task 2, predicting the length of stay is a regression problem. Through the application of several models, we chose four models that perform better: Multiple Linear Regression, Elastic Net Regression, CatBoost (after the model ensemble mentioned in 4.1.7), and Multi-layer Perceptron classifier (MLP).

5. Experiments and Results

5.1 Analysis on Task1: Mortality Prediction

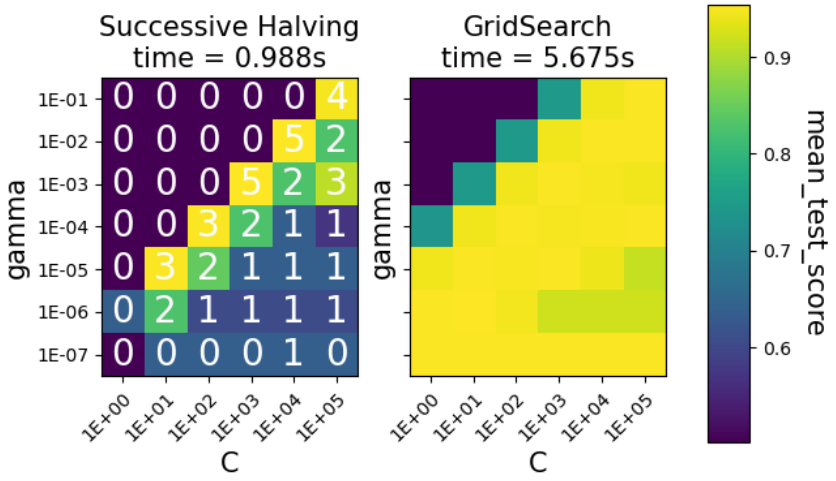
Among the five models adopted, the best model to predict mortality is XGBoost. To restate the data cleaning method used in the XGBoost with the best performance, we only used the mean values taken at the end of each day, standardized the captured data, and imbalanced the values about the prediction dataset with unbalanced ‘0’ and ‘1’ values.



Pic 7. Performance Comparison Between the Five Adopted Models

5.1.1 Hyperparameters tunning

In total, six hyperparameters were adjusted by the halving grid searching method with cross-validation. According to the documentation of Scikit-learn (Scikit-learn Developers, n.d.), the halving grid searching method may optimize hyperparameters as well as the normal grid searching method. The principal comparison of the two methods is shown in pic 8. To compare the performances of the optimization results, we adopted the area under the curve ("AUC").



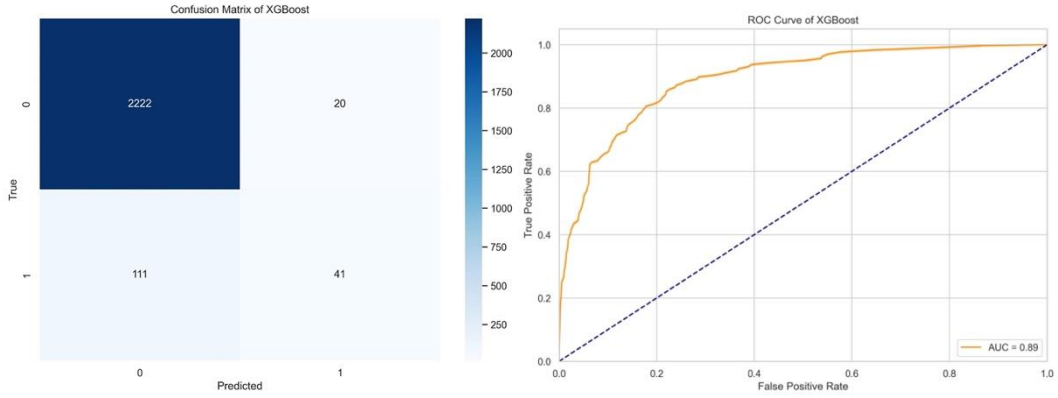
Pic 8. Principal comparison of halving grid search and normal grid search

The optimization of hyperparameters started at the number of estimators using the built-in method of XGBoost. It helps to confirm the minimum number of estimators needed for training with a specific training dataset. Maximum depth and minimum child weight, which are optimized afterward, can help the model more complex and more fit to the dataset

inputted, and indicate the minimum sum of instance weight needed in a child respectively. Gamma was found independently for minimal loss reduction after partitioning on the leaf nodes of the tree. We then tried to grid search the subsample ratios of training instances and columns when building each tree to find the value that gives the best AUC value. To further avoid overfitting, both L1 and L2 regularisation was adopted. Eventually, we tried to optimize the learning rate.

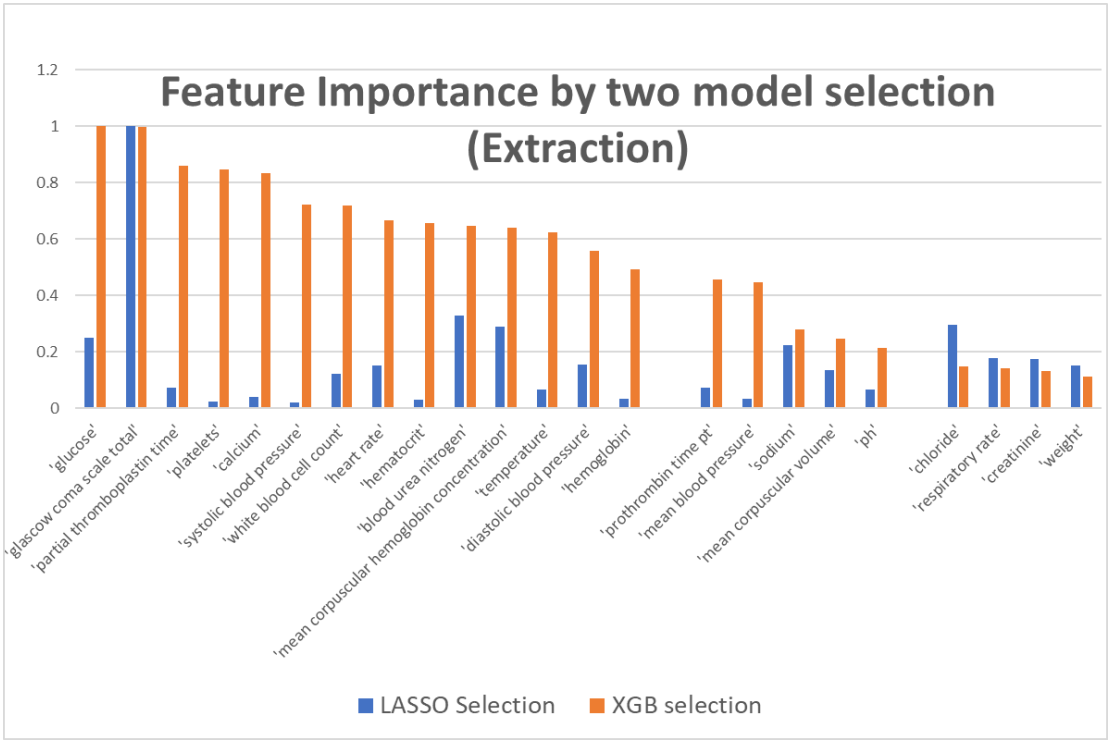
5.1.2 Results and Discussion

The results of the optimized XGBoost model performance can be viewed below:
The picture on the left shows the confusion matrix of XGBoost, and the picture on the right shows the roc map generated by a valid set generate. Here AUC reaches 89%, in the test data set, AUC can reach more than 91%.



Pic 9. XGBoost Results

After the XGBoost model is trained, we use the model to sort the importance of variables (the orange bar in Pic 9, shows only the relatively important ones). After standardization, comparing the importance ranking of lasso selection (blue bar in Pic 9), we found that the two models have different emphases on feature importance. However, features that both models value deserve our attention. They include: 'The Glasgow coma scale total', 'glucose', 'blood urea nitrogen, 'mean corpuscular hemoglobin concentration', and 'diastolic blood pressure. These features might have significantly solid meaning for predicting mortality and hopefully can provide some insights for biomedical research.



Pic 10. Feature importance by two model selection for mortality prediction

5.2 Analysis on Task2: LOS Prediction

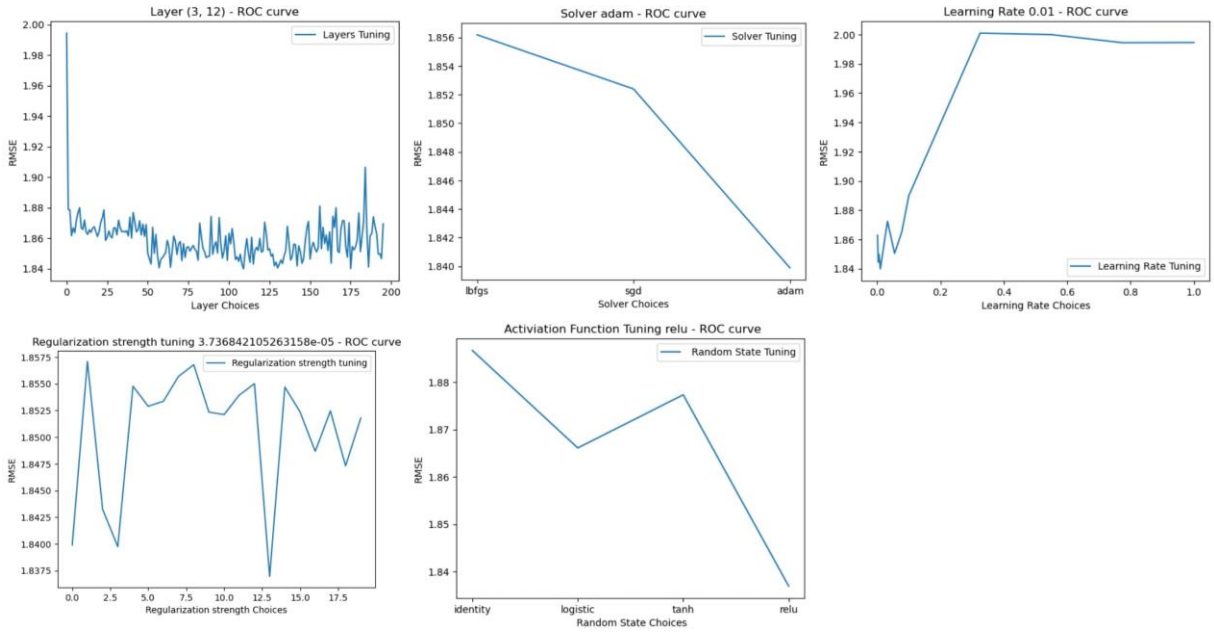
Multilayer Perceptron (MLP), as a primary-level technique of deep learning, shares a new way to process the data, generating a considerable result comparing classical ML models.

In this paper, it is proposed to use the setting of depth-4 MLP. Since the dataset MIMIC-III is not a complicated one, using a deeper network or more advanced techniques such as CNN architectures may result in the overfitting problem. To balance the performance of the model, the number of hidden layers is designed to be 2. During the processing, the workflow shall be (1) using MLPregressor in the Sklearn library to Find the best set of parameters (2) building a network using Pytorch with a more customized setting; and (3) adding an external module supplementing the model in the hope of enhancing performance. During the whole process, we adopt RMSE to evaluate the performance of our models.

5.2.1 Hyperparameter tuning

There are multiple trials with fluctuating results during the process. With various initial parameter settings, it is envisaged that the model would find the best configuration for obtaining a global minimum. To avoid redundant results misleading audiences, it is only the best scenario that is discussed. In the parameter-tuning, we first seek the best set of neurons in each hidden layer. Usually, having a lot of neurons causes the overfitting issue. Several attempts have been made to set many neurons, such as 100, 500, or 1000. It is to be predicted that the RMSE increases more rapidly than in models with fewer neurons. Thus, we fixed the range

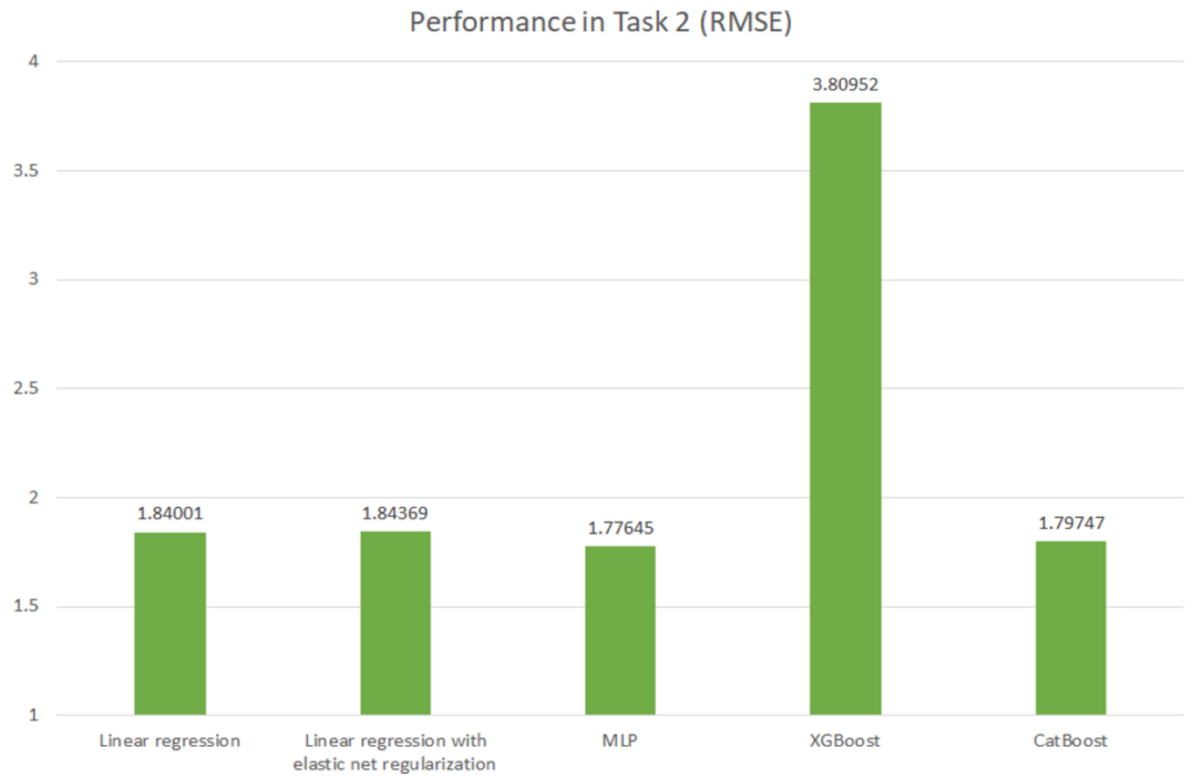
from 1 to 20 in the final trial. It turns out that (3,12) is a great combination of the number of neurons. For the supported solvers, Adam gets drastically different results from the other two. Adam is a fast-converging technique for stochastic objective function optimization using first-order gradients and adaptive estimations of lower-order moments. Thus, it has the advantages of computational efficiency and the capability to generate a remarkable performance. There is a list of learning rate and ridge regularization strengths we pre-selected for the tuning process. These values are decided from prior knowledge – For learning rates, we have 15 values from 0.001 to 1; For regularisation strengths, we have 20 choices from 1e-5 to 0.00005. The best result we obtained is 0.01 and 3.74e-05 for learning rate and regularisation strengths respectively. Finally, Relu is the option for the activation function. It is widely used in many of other advanced deep neural networks due to its outstanding performance and efficiency. The pic below shows the parameter tuning process.



Pic 11. Parameter tuning process

5.2.2 Results and Comparison

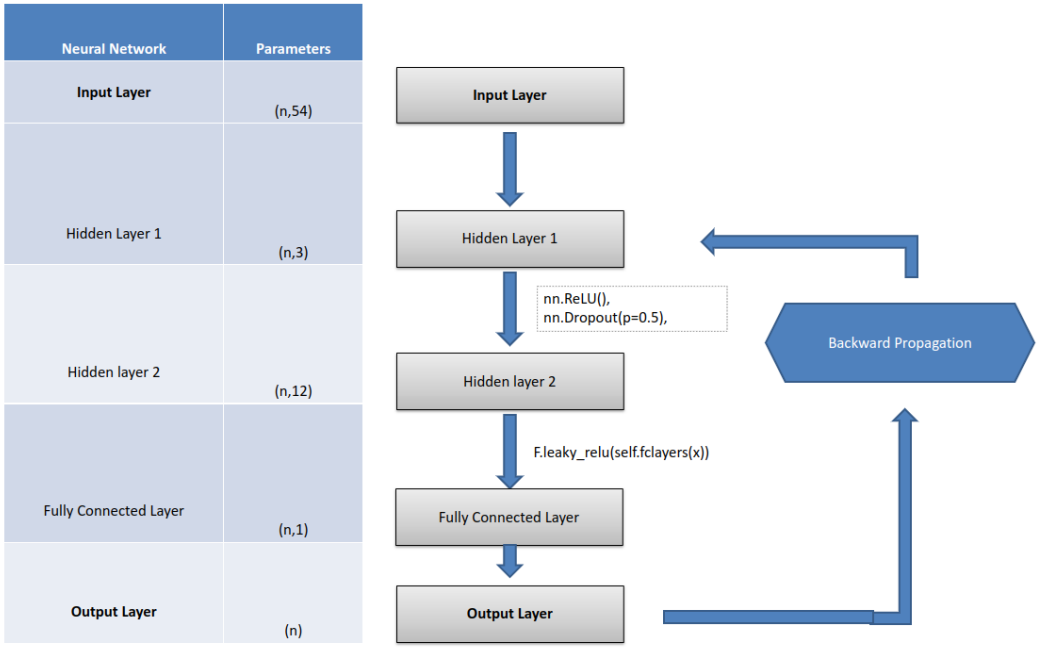
In this task, we also adopt other ML models, such as Catboost, XGBoost, and linear regression. In the evaluation of the Kaggle leaderboard, we get an RMSE of 1.77645 when Using the MLP models, which is the highest among all models. It is 2.03 smaller than the RMSE that gain by XGBoost, which shows the superior capability in LOS regression of Multilayer perceptron. The result and more detailed comparison are shown in the chart and table below.



	MLP	Linear Regression	ElasticNet	CatBoost	XGBoost
Value	1.77645	1.84001	1.84369	1.79747	3.80952
Difference	0	0.06356	0.06724	0.02102	2.03307
Percentage(%)	0	+3.577922261	+3.785076979	+1.183258746	+114.4456641

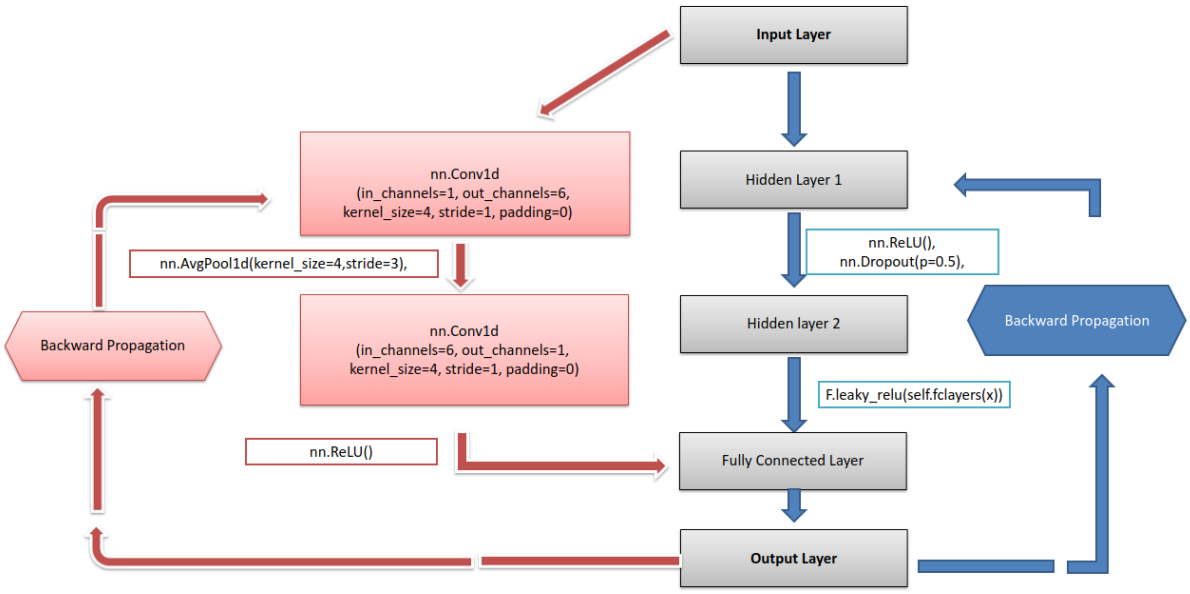
Pic 12. Performance from the five models in task 2

When we build a network using Pytorch with a more customized setting, it is hoped that we could design a more complicated model that could generate better performance. We use the best setting we gain in the former step except that we add batch normalization, a drop-out technique, and leaky Relu as the final activation function. Batch normalization is proposed for a faster and more stable output by re-centering and re-scaling the inputs for the layer. Drop-out techniques are to avoid the overfitting problem caused by the advanced architecture of MLP. Leaky Relu, as an improvement of Relu activation, also provides a good insight when we deal with negative output, incorporating some information from negative values. As a result, we yield a lower RMSE when using this new model. The RMSE on the validation dataset is 1.44, which is a huge improvement compared to the MLP using Scikit-learn. A rough visualization and a detailed workflow have been attached in pic 13.



Pic 13. Rough Visualization and Detailed Workflow

Further improvement is adding an external module to enrich the capability of the model. We added a CNN architecture module to the Pytorch MLP. The module contains 2 convolutional layers with an average pooling layer to reduce computational cost. The output from the CNN module and output from the MLP hidden layers will be aggregated and passed through a fully connected layer. Finally, we get the output from the previous layer. The architecture is shown below.



Pic 14. MLP architecture

However, the new model fails to outperform the second model. The RMSE on the validation dataset is only 1.99 which is the worst among the 3 proposed MLPs. The table shows the comparison of all 3 proposed MLP models. Hence, it is proposed to use the pure MLP model which is built using Pytorch with setting: num_layers = 3, learning_rate = 0.001, batch_size = 32, num_epochs = 500, l2_lamb = 3.736842105263158e-05, Relu activation function, drop-out techniques.

	MLP (Sklearn)	MLP (Pytorch ver1)	MLP (Pytorch with CNN module)
RMSE	1.78	1.44	1.99
Ranking	2	1	3

Table 4. MLP performances

6.Limitation

In the model studying, our hyperparameter tuning is insufficient for each model, and the accuracy for each model might not be very precise. For different models, we use the same training set, which may lead to bias.

Moreover, our study uses statistical methods to analyze biological studies. However, as we need more biological data insights, the feature selection might need to be more convincing in biology. If we have a deeper understanding of bio-features, it will aid us in having better feature selection.

Trade-offs between “meaningful data” and “data that can generate high accuracy” is also a topic to consider. As discussed in chapter 3, many approaches to pre-process the data, which aim at making it more “meaningful” were adopted in this research. However, the model results were not as good as expected, at least on the test set that we have. We have tried more meaningless data sets (for example, putting meaningless data marker masks into training), and sometimes its performance was better, but we have discarded such obvious meaningless discussions. We believe that meaningful data still generally leads to better results, and in other words, models trained on meaningful data sets will have lower variance, even if the results now have a higher bias. The important remaining problem is that we need to think more deeply about to what extent we should control the subjective “meaning” of the data, and how much objective space should be given to the machine to train the model.

7. Conclusion

For mortality and Length of stay prediction, we attempted several classification and regression models to find the best fit model. We applied Elastic-net regularization, SVM, XGBoost, RNN, and MLP to solve the first question. After comparing the validation accuracy and modifying the hyperparameter, we believed XGBoost had the best performance in predicting mortality, with 0.89 AUC in valid Y prediction and 0.91125 in testing AUC. We found ‘The Glasgow coma scale total’, ‘glucose’, ‘blood urea nitrogen, ‘mean corpuscular hemoglobin concentration’, and ‘diastolic blood pressure is valued by both Lasso feature selection and

XGBoost feature selection, which might have important biological relevance for predicting mortality.

To predict LOS, we used elastic-net regression, logistic regression, multilayer perceptron (MLP), and cat boost (after model ensemble) to fit the trained model. After comparing the RMSE, we believe MLP is the best model for LOS estimation with 1.44 in fit valid data. From the feature selection done by CatBoost (after model ensemble), we found that ‘Glasgow coma scale total’, ‘blood urea nitrogen, ‘tidal volume observed’, ‘respiratory rate set’, and ‘calcium lionized are viewed as the five most important bio-features regarding predicting the LOS.

Finally, a website with these two optimal models built in have been opened, allowing the medical system to access the model prediction and obtain treatment insights by easily adjusting and inputting the bio-feature data.

Reference

Alabbad, D. A., Almuhaideb, A. M., Alsunaidi, S. J., Alqudaihi, K. S., Alamoudi, F. A., Alhobaishi, M. K., Alaqeel, N. A., & Alshahrani, M. S. (2022). The machine learning model for predicting the length of stay in the intensive care unit for Covid-19 patients in the eastern province of Saudi Arabia. *Informatics in Medicine Unlocked*, *30*, 100937. https://doi.org/10.1016/j.imu.2022.100937

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin. classifiers. *Proceedings of the Fifth Annual Workshop on Computational Learning Theory - COLT '92*. https://doi.org/10.1145/130385.130401

Chan, K. K.-Chung. (2019). ICU Beds Utilization Pattern. 2019 Hospital Authority Convention. Retrieved 2022, from https://haconvention2019.dryfta.com/abstract-archive/abstract/public/1351/icu-beds-utilization-pattern

Chen, T., & Guestrin, C. (2016). XGBoost. *Proceedings of the 22nd ACM SIGKDD. International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939785

Cramer, J. S. (2002). The origins of logistic regression (PDF) (Technical report). Vol. 119. Tinbergen Institute. pp. 167–178. doi:10.2139/ssrn.360300.

Gunčar, G., Kukar, M., Notar, M., Brvar, M., Černelč, P., Notar, M., & Notar, M. (2018). An. application of machine learning to hematological diagnosis. *Scientific Reports*, *8*(1). https://doi.org/10.1038/s41598-017-18564-8

Scikit-learn Developers. (n.d.). *Comparison between grid search and successive halving*. https://scikit-learn.org/stable/auto_examples/model_selection/plot_successive_halving_heatmap.html

Straathof, B. T. (2020). A Deep Learning Approach to Predicting the Length of Stay of Newborns in the Neonatal Intensive Care Unit.

Taslimitehrani, V., Dong, G., Pereira, N. L., Panahiazar, M., & Pathak, J. (2016). Developing EHR-driven heart failure risk prediction models using CPXR(Log) with the probabilistic loss function. *Journal of Biomedical Informatics*, *60*, 260–269. https://doi.org/10.1016/j.jbi.2016.01.009

Tranmer, M., & Elliot, M. (2008). *Multiple linear regression*. Cathie Marsh Centre for

Census and Survey Research, 5(5), 1–5.

Wikipedia contributors. (2022, November 10). *LightGBM*. Wikipedia. https://en.wikipedia.org/wiki/LightGBM

Wikipedia contributors. (2022, October 9). *CatBoost*. Wikipedia. https://en.wikipedia.org/wiki/CatBoost