1. Feature extraction

a)

i. Icutype is currently being treated as a continuous variable with a single integer value ranging from 1 to 4. It should be treated as a categorical variable. Due to it being treated like a continuous variable when using a linear classifier, it may view 4 as "greater than" the other numbers, which is not the case for this feature. We can use one-hot encoding and treat each ICU type as a binary variable.

ii. With this representation, it would not fully capture possible fluctuations over time. With just the average, it makes it seem constant when it may not be constant.

b)

i. This approach assumes that the values are missing randomly, and that they are independent from other variables. It also assumes that the mean is a good representation of the missing variables. One way to handle this would be seeing if the feature has a lot of missing values. If it does, then we could drop the feature entirely. Another way would be to predict what the missing value may be by using regression models trained on the observed data.

c)

i. This is helpful because this way, all features operate on the same scale while still preserving the original distribution for each feature.

d)

i. There are 40 features, and here are the averages along w/ the names

2.1 Hyper parameter selection for L2-Regularized logistic Regression

b) This is beneficial because this ensures that each fold maintains a similar class ratio, the model learns better decision boundaries, and it reduces the variance

c)

performance measures	Best C	CV performance
Accuracy	100	0.8655
precision	1	0.6114
sensitivity	1000	0.2342
specificity	0.001/0.01/0.1	1.000
F1-score	1000	0.3301
AUROC	1	0.8013
AUPRC	1	0.4545

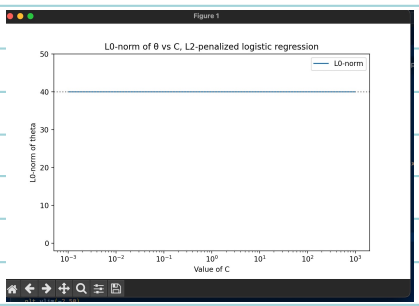
For accuracy, the performance peaks when C=100.
For precision, it starts at 0, then rises significantly when C=1.
For sensitivity, it starts off low, then it peaks when C=1000
For specificity, the performance is at the absolute peak (1.000) when C is small
For F1-score, it peaks when C=1000
For AUROC and AUPRC, it peaks when C=1.
I would rely heavily on AUROC b/c it best balances sensitivity and specificity, and so for C I would go with C=1.

d)

chosen C=1

performance measures	Test performance
Accuracy	0.87
precision	0.6842
sensitivity	0.1806
specificity	0.9860
F1-score	0.2857
AUROC	0.8391
AUPRC	0.4754

e)



The # of non-zero entries for L2 regularization remains at 40 throughout all of the values of C b/c L2 does not make any coefficients =0, but it makes them close to 0. All features still contribute.

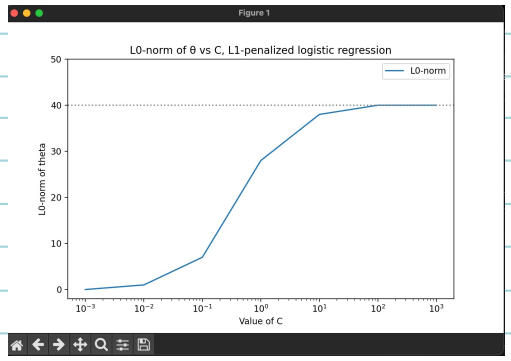
f)

⊕ coeff	feature name
2.659	mean-BUN
2.1854	age
1.6816	mean_Bilirubin
1.519	mean_lactate

⊖ coeff	feature name
-2.6373	mean-GCS
-1.647	mean_temp
-1.334	mean_NA
-1.1414	Height

2.2 Hyperparameter selection for L1 regularized Logistic Regression

a) The best C was C=1.000, CV= 0.8016, and Test performance = 0.4754

b) 

When C is small, there are a lot of coefficients that are equal to 0, but as C gets bigger, there are less that equal 0, so more features get retained. L2 does not force coefficients to be 0 like L1 does, which explains the trends in both graphs. L1 selects features that are more important, so those classifier coefficients may be larger since there are less.

3. Challenge

For feature extraction and preprocessing, I decided to use the same methods from hw3_main.py. I called generate_feature_vector, impute_missing_values_challenge, and normalize_feature_matrix_challenge.

For the hyperparameters, I did a logistic regression with solver = "liblinear" so I could use L1 and L2. I also set the class_weight to 'balanced' b/c there are less patients that are predicted to be in the death class, so it helps with the imbalance. Then I tried to find the best C from the log space 10^{-3} - 10^{3}, and I set the penalties to L1 and L2. Then I did RandomizedSearch CV and a 5-fold cross-validation. I also used AUROC to help me find the best C from those experiments.
Then after I found the best parameters, I ran those on the 10,000 labeled examples, and that gave me the final classifier. Then I evaluated the model on the training set using the metrics we learned in class.

confusion matrix: 7157 1303
597 943

accuracy: 0.81 Recall: 0.6123 F1 score: 0.4982
Precision: 0.4199 Specificity: 0.8460 AUROC: 0.8174

Finally, I got the probabilities using clf.predict_proba(X_heldout)[:,1], converted them into labels {-1,1}, and uploaded them to challenge.csv.