Final Report

ESE 527

Yiru Wang, Wan-Yun Shen

Executive Summary

Since Covid-19 has caused a global pandemic and the patients are still increasing, the efficiency of hospitals has become a serious issue. We want to save as many lives as possible, but also assure that the patients who are at high risk can stay in the hospital to be taken care of at the same time, then the distribution of the medical resources become an important problem. This project aims to classify the patients and predict the time that future patients need to stay in the hospital and try to optimize the whole process.

This research project is a classification problem, we want to predict the length of stay for each patient and separate them into several classes including long-term and short-term. As we already know, most of the hospitals are overloaded with patients and there are a lot of people who cannot get medical care due to the unequal distribution of the medical sources. This may cause severe after effects or even take away lives. In this case, optimizing the efficiency of the hospital became the first priority. If we can accurately predict the length of stay for each patient, we are able to allocate each patient correctly and know how many more patients we can accept each day thus reserve the ward and other required sources. It would be a great benefit to society because the pandemic seems endless and there are variants emerging every once a while, which make the number of patients far from decreasing. Also, hospitals have to cure people other than covid-19 patients, so it is important to keep them from overwhelming. It is definitely also important to me because people around me might accidentally get covid or other diseases and have to go to the hospital, besides, I would like to make some contribution to the community to help reduce the risk of medical scarcity. Maximizing the efficiency of hospitals is important to me and also the whole society.

Data Description/Preprocessing

For the raw data, the training data contains 18 columns: the case ID that refers to each patients, code of hospital vary from type to region, available rooms in each hospital, department overlooking the cases, features of ward including bed grades, features of patient in level of severity of illness and region, visitors for patients, and other related variables such as age, deposit, and staying days. For the data cleansing, we first drop the data with missing values, then omit the columns that we think are not correlated with the staying days of the patients by common sense. For example, patientid and the grade of bed definitely have nothing related to the time period of a patient in staying. After which, the dataset is divided into long-term and short-term with the stay time over 60 days time period by analyzing the distribution of the parameters, and we create a new column of y with the long-term stay factored as 1, and short-term as 0.

Then the feature plot is drawn to understand the categorical parameters. We plot the histogram with x-axis of the factored stay time and with y-axis as the count of each class of the categorical variable. From Figure 1, the first plot is draw on the age groups, and we can see that people age between 31-50 are more likely to stay while going to the hospital; and for the second plot on the severity of illness, most people stay in hospital under the moderate level of illness in both long-term and short-term stay; for the last plot based on the

type of admission, most people enter the hospital and stay with trauma admission. Moreover, from the distribution and pattern of these graphs, we can see there is a large difference on the counting number for all three variables between short-term and long-term, which means people are less likely to stay in hospital after 60 days stay, and that is also the reason for this project to define short-term stay as under 60 days and long-term stay as 60 days and above.

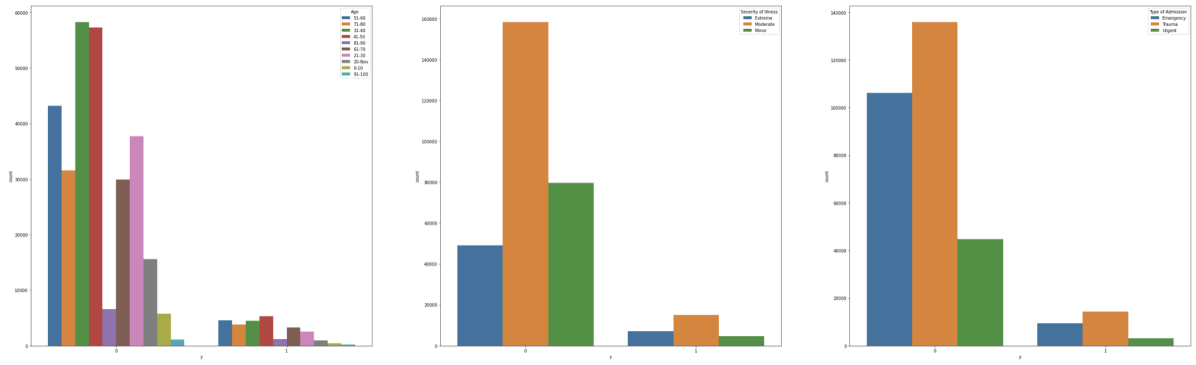

Figure 1: plot of categorical parameters

For further analysis, the original data are separated into training data and testing data by 70% and 30%. After which, we used one hot encoding to manage the categorical data. One hot encoding is a method that is often used to convert categorical data into numerical data without assigning them with order, which can improve accuracy of classifying our data. It encodes each class by creating a new column with a decision of 1 or 0 to redefine the original data as several new parameters. To implement one hot encoding, we import OneHotEncoder from the sklearn.preprocessing package to encode the non-numerical data. After that, we standardized the encoded data by using the StandardScaler package from sklearn to make our model more accurate by standardizing the data into a similar range for the further analysis.

As for outlier detection, the statistical techniques that we learnt from the lectures are dealing with the numerical dataset, which the outliers are detected with the extreme data points to the overall pattern of a specific dataset. While our data consist of the categorical data that put patients into groups such as the severity of the illness, and other meaningful variables like entry deposit, which is hard to define an outlier under such types of datasets. So the outlier detection is not applicable to our data, and dropping them will lead the analysis in a negative direction and our classification will no longer be accurate.

Model Approach

This project applied three statistical models in doing the classification, which are logistic regression, KNN, and random forest algorithm. The hyperparameter tuning with cross validation technique is implemented for all three models in choosing the best hyperparameter and testing for the highest accuracy. The performance of the models are measured by the confusion matrix with accuracy and AUC score for comparison, and the detailed method description and result are explained as below.

Logistic regression

Logistic regression is set to generate the categorical dependent variables. It differs from the linear regression by the Y. Here, the Y is discrete only on 0 and 1, and the relationship between X and Y is turned to the probability of successfully getting the value 1 on Y or not, which this probability is continuous and reasonable for the calculation. The logistic limits the

prediction value within [0,1], and the sigmoid function is applied to the linear regression function: $p = F(x) = \frac{e^x}{e^x+1} = \frac{1}{1+e^{-x}} \Rightarrow \frac{1}{1+e^{-f(x)}} = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$. Then the output variable Y follows the Bernoulli distribution, where: $P(Y = y_i) = p^{y_i}(1-p)^{1-y_i},\ 0 < p < 1,\ y_i = 0,\ 1.$ The loss function of this method is computed via the maximum likelihood estimation $L(\widehat{\beta}) = \prod_{i=1}^{n} h_\beta(x_i)^{y_i}(1-(h_\beta(x_i))^{1-y_i},\ where\ p = h_\beta(x_i) = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}}$, and the parameter β is calculated by maximizing this equation:

$argmax_\beta \sum_{i=1}^{n} [y_i log(h_\beta(x_i)) + (1-y) log(1-h_\beta(x_i))].$

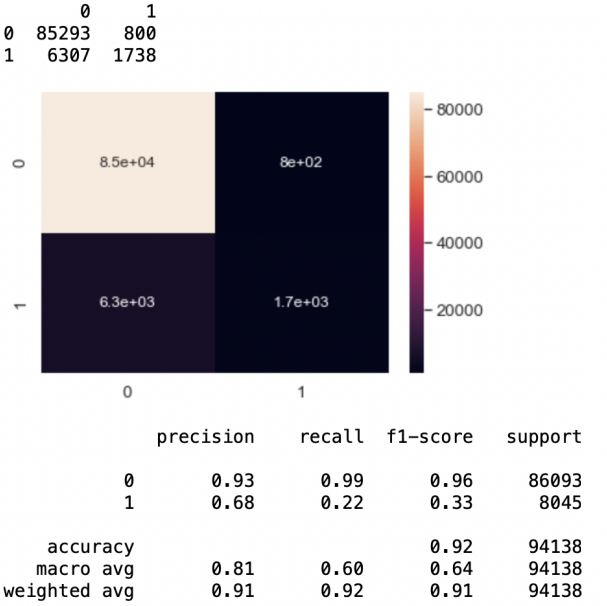

Figure 2: confusion matrix and accuracy of logistic regression

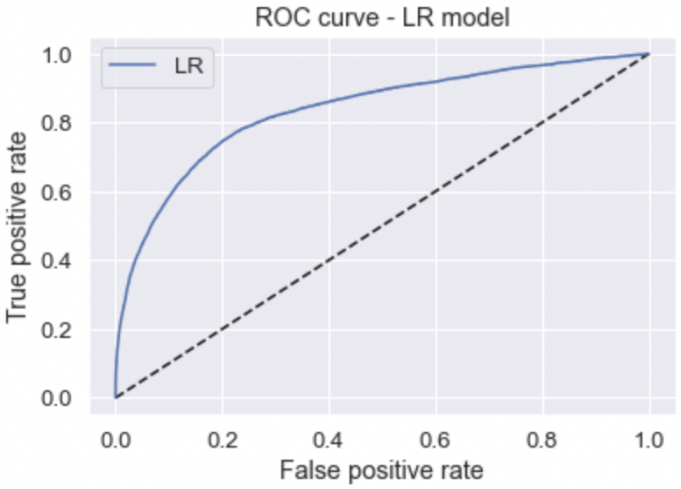

Figure 3: ROC curve of logistic regression

For this project, the LogisticRegression function in python is applied to generate this algorithm. In order to predict Y, the probability is predicted using the 'predict' function, it is then mapped into data with 0 or 1 and the data is set to be 1 if the probability is greater than threshold 0.5, which the function f(x) is a linear function under this condition. Next, the Lasso and Ridge regression is applied with l1 and l2 in selecting the parameters and choosing the best parameter set, in which, the result shows that the best score is achieved with the penalty of l2 and best parameter set of 0.01. Having the generated probability with the best parameter, the confusion matrix is computed with the testing set (Figure 2). After fixed encoding problem of categorical data and applied standardization, the accuracy of this prediction algorithm is improved to 92.5% calculated from the confusion matrix with equation $\frac{correct\ prediction\ on\ Y=1}{total\ predictions} = \frac{86121+1710}{86121+1710+782+5525} = 0.925$. Moreover, the ROC curve is plotted with the performance of all the thresholds (Figure 3), and the AUC score is calculated for measuring the performance of the model. AUC score is the area under the ROC curve, and a higher score indicates a better performance. For this logistic model, the AUC score is 0.836, which shows that logistic regression is a suitable method in predicting this dataset.

KNN

KNN is a supervised machine learning algorithm that can be used to solve both classification and regression problems. The steps of implementing a KNN algorithm are:
1. Load the data.
2. Choose the value of K, which is a parameter that refers to the number of nearest neighbors to include in the process.
3. For each example in the data, calculate the distance between the query example and the current example from the data, then add the distance and the index of the example to an ordered collection. The euclidean distance calculation is usually used to calculate the distance.
4. Sort the ordered collection of distances and indices from smallest to largest by the distances.
5. Pick the first K entries from the sorted collection
6. Get the labels of the selected K entries
7. Return the mode of the K labels
In this project, python is used to implement this algorithm by using the sklearn package–KNeighborsClassifier.

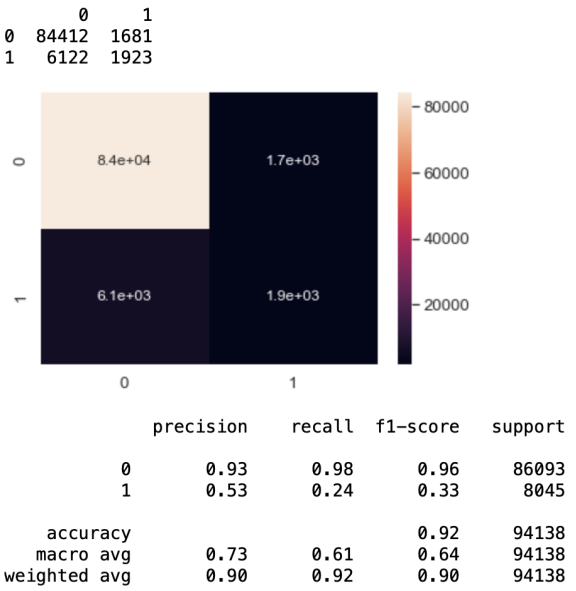

Figure 4: confusion matrix and accuracy of KNN

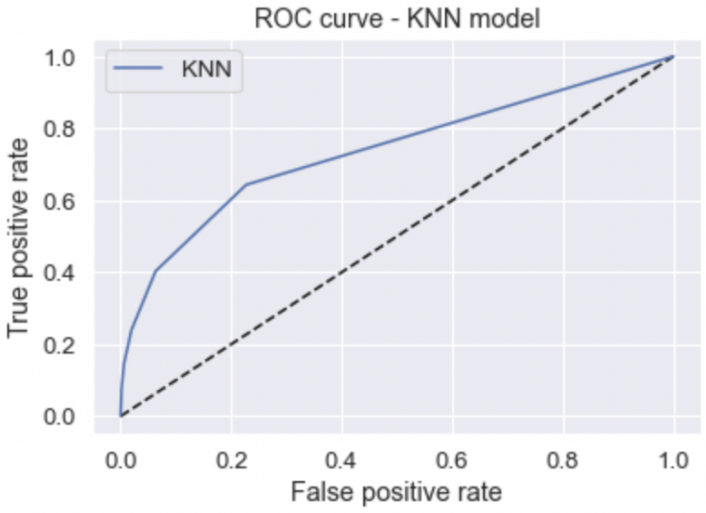

Figure 5: ROC curve of KNN

After the hyperparameter tuning, we get the best parameter set of 5 nearest neighbors with accuracy of 91.7%, and a confusion matrix (Figure 4) is plotted to better show the result. The ratio of the length of stay in data was 10.7(long-term divided by short-term), and the ratio of the training result was 10.7, which is almost the same. Then the ROC curve is then graphed and the AUC score is calculated as 0.738, which shows that the KNN is a relatively accurate classification model for this project.

Random Forest

Random forest is a supervised machine learning method that is widely used in classification and regression problems. It is built on the basis of the decision trees. Decision tree is a tree-like structured model with each node representing a test and each branch is a test result, and this method will apply the feature selection by calculating the entropy and choose the parameter as the splitting feature by the result order to do the classification. The random forest method includes a sampling idea. Suppose there are N observations in the dataset, and m features of the data. Random forest first takes n observations randomly from the

dataset and picks k features randomly from the total m features to calculate the best splitting mode of the decision tree, then it repeats with the previous process to get a large number of decision trees to form the random forest model. Random forest algorithm reaches the classification result by taking the majority vote of classified result from the decision trees or taking average in the regression case.

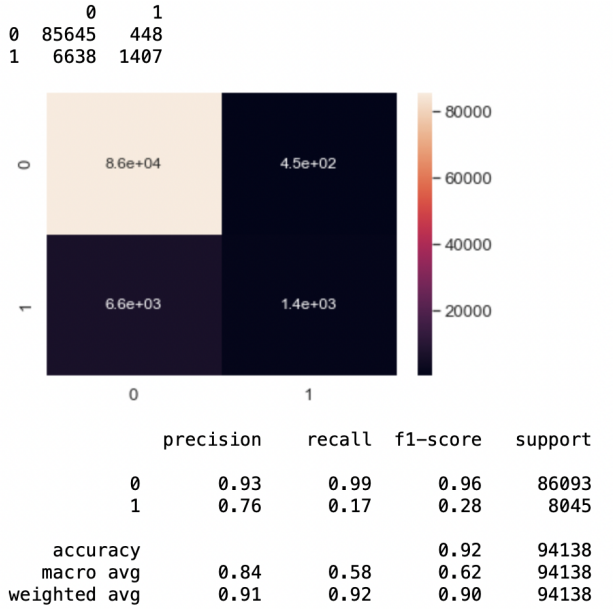

Figure 6: confusion matrix and accuracy of random forest

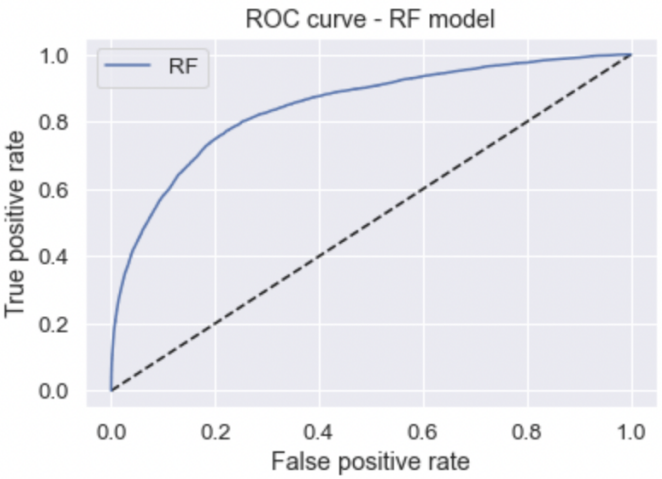

Figure 7: ROC curve of KNN

For this project, the random forest method is first applied in the gridsearch to test for the best estimators, then the result shows the best model with 100 estimators and depth of 10 for the trees. Then a confusion matrix is computed and plotted to show the result of classification (Figure 6). From the result, the accuracy is 92.5%; from the ROC curve, the AUC score is calculated as 0.844, and since random forest is a more flexible method that is widely used for large datasets, it is a better classification result than the logistic regression method. Compared with the KNN, it is less time consuming since the classify approach is straightforward and accurate and it does not need to compute all the euclidean distance and make calculations. Therefore, the random forest method currently returns the best classification result for this dataset.

Results and insights

Comparing these three models, the random forest has the best result, which has accuracy of 92.47% and AUC score of 0.844 as mentioned above. This result shows that random forest is an efficient model for estimating the hospital stay time. The result is reasonable for random forest is mostly applied for large datasets and it is more straightforward and flexible than the other two models, which is also least time-consuming. The random forest classifier performs better with more categorical data than numeric, and logistic regression is more good at dealing with the numerical data, which explains why the random forest has a better performance in this project; when applying KNN, it needs to scan the whole dataset and calculate the distance for each data points to do the prediction, and for this project, the large dataset makes it both time-consuming and with low accuracy.

Other than the statistical interpretation, this result means that we are able to successfully predict the patient's hospital stay time with a high accuracy using a random forest model. Combined with the data distribution and frequency that discussed above, this progress can

help hospitals better allocate its resources and save more people's lives with less sources wasting. Especially under a pressing pandemic situation, this technique will be efficient in helping hospitals make wiser decisions, which is also the motivation for this project after witnessing people dying because of the resources shortage of covid pandemic, and this project can make its contribution in preventing the similar tragedies.

Conclusion

As we mentioned in our summary, we wanted to classify the patients and predict the time that future patients need to stay in the hospital and further maximize the efficiency of hospitals. In order to achieve our goal, we applied three classification models including logistic regression, knn, and random forest to train and test the dataset that is collected from real hospital cases. From our results, we can see that it is possible to classify and predict the length of stay of the patients with the random forest model, which means that we can optimize the efficiency of hospitals by predicting the patients' length of stay while they enter the hospital. If the length of stay is predicted precisely, the resources can be allocated reasonably and save sources for the potential events to prevent the hospitals from overloading thus saving more lives. During the pandemic, the mortality rate increases because lots of the original patients' medical resources are occupied by the covid infected patients and lead to their mortality. With our prediction, we will be able to save more lives and time and also optimize the whole process. We can easily classify the patient into long-term staying or short-term staying groups, and make different treatments for each group. Nowadays, in a world where technology is improving and artificial intelligence is arising, we should use these new methods to solve existing problems. Since the pandemic seems to continue, our prediction of the hospital stay length will be a significant contribution to society. For improvement and applications in reality, more comprehensive data is needed and more features are required including the time series data and survival analysis could be considered to make the model more practical and complete.

Appendix

Github link for the project: **https://github.com/wanyun617/ESE-527-Project.git**