In-Hospital Mortality Machine Learning Model

Andrew R. Plant
Department of Computer Science
Texas A&M University
College Station, TX 77840
plantand000@tamu.edu

Abstract

In this project, we aimed to create a machine learning model to predict in-hospital mortality of ICU patients based on the first 48 hours of their admission. Our dataset consisted of almost 2000 patients, and we focused heavily on preprocessing the data to make it easily trainable. We used a Support Vector Machine with probability and conducted hyperparameter tuning using gridsearch to find the best C, kernel, and gamma. Additionally, we utilized PCA whitening to improve the data quality. Our model achieved an AUROC score of 0.89515 on the test dataset, which was the highest achievable score. The results indicated that our model is effective in predicting in-hospital mortality, and its accuracy did not decrease when tested on new data. Overall, our findings suggest that the implemented method can be used to predict in-hospital mortality with high accuracy, which can significantly improve patient outcomes.

1 Introduction

The prediction of in-hospital mortality is a critical task in the field of healthcare, and it can aid physicians in decision-making processes to improve patient outcomes. In this project, we aimed to create a machine learning model to predict in-hospital mortality of ICU patients based on the first 48 hours of their admission. We utilized a dataset consisting of nearly 2000 patients, and we focused on preprocessing the data to make it easily trainable. Our approach involved the use of a Support Vector Machine with probability, and we conducted hyperparameter tuning using gridsearch to find the best C, kernel, and gamma. Additionally, we utilized PCA whitening to improve the data quality. The results of our method showed an AUROC score of 0.89515 on the test dataset, which was the highest achievable score. These findings suggest that our model can be effective in predicting in-hospital mortality with high accuracy, thus improving patient outcomes. We did not use other resources to solve this problem but when compared with similar models, we achieved a relatively high AUROC score in comparison to our peers.

2 Methodology

The approach to predicted in-hospital mortality requires several important steps. First, the data is loaded from CSV files, preprocessed, and split into training and testing sets. These sets are then used to train a support vector machine (SVM) using a grid search to optimize the hyperparameters. The trained model is then used to predict the probability of in-hospital morality for the test set and accuracy and area under the receiver operating curve is calculated to ensure the model is not over-fit and transfers well to unseen data. The methodology has a heavy focus on data preprocessing so that the model can train on easily interpretable data thus leading to a higher accuracy.

Preprint. Under review.

2.1 Data Preprocessing

The unprocessed training X data contains 791937 lines of data points to train from while the output is the in-hospital mortality rate for a given patient. This means that there are multiple rows of data for a given patient making it impossible to train a SVM without some form of data preprocessing. Our method of preprocessing reshapes the data such that each patient gets one individual row with all metrics existing on that row to be trained on by the SVM model. The preprocess function takes in pandas dataframe and drops the numerical first column as the data will be aligned by patientunitstay. The first step in data preprocessing is to alter the demographic data (e.g. age, ethnicity, gender) and put it in a trainable format. The non-null values for the rows of the demographic columns are fetched, if the rows have null data for a given patient, the data is replaced with average (for male/female height and weight only). We then convert any instances of age > 89 to 90 so that it can be a trainable integer value similar to all other age entries. The categorical columns of gender and ethnicity are one-hot encoded and concatenated with the rest of the data. That concludes the demographic data preprocessing.

The next section of preprocessing deals with the cellattribute celllabel, and offset columns. We remove instances of 'hands' and 'feet' from cellattribute as they are not neccessary for training and change the offset value to be a numeric column and group these by patientunitstayid. Next, we loop through all the rests to find the highest offest which is the most recent test taken for a given patient. We then convert the most recent cell data using one-hot encoding again. We the merge the transformed cell data with the demographic data on the patient id.

The third step involves processing all nursing data (e.g. nursingchartcelltypevalname, nursingchartvalue) which we convert the caterical attribute header using one-hot encoding and transform the data into a singluar list to be appended to a row of the final dataset.

The final set is to take all numerical columns and use a StandardScaler to fit-transform the values and recombine with the categorical values. This ensures that features with different scales are given equal wieght by the support vector machine when training creating a much better model.

Table 1: Sample unprocessed x-table

admissionheight	admissionweight	age	cellattributevalue	celllabel
157.5		87		
157.5	58.5	> 89		
177.8	70.3	78		

Table 2: Sample preprocessed x-table

pH	Respiratory Rate	O2 Saturation	Heart Rate	Non-Invasive BP Systolic
False	True	True	True	True
True	True	True	True	True
True	True	True	True	True

2.2 Model Design

In this project, the chosen model is a Support Vector Machine classifier with a radial basis function kernel. This choice was made based on the fact that SVMs have been shown to perform well on classification tasks, particularly in cases where the number of features is relatively large compared to the number of samples. The RBF kernel is a popular choice for SVM classification, as it can model complex decision boundaries in the data. In addition, Principal Component Analysis (PCA) was considered as a preprocessing step, but was not used in the final model as it slowed the training process and did not yield a higher AUROC or accuracy score.

2.3 Model Training

The model uses the built in fit function which, in this scenario, does an internal 5-fold cross validation on the train X data. This was paired with a grid search to tune the hyperparameters of the model.

2.4 Hyperparamter Tuning

The hyperparameters of the SVM classifier were tuned using a GridSearchCV method, with the following parameter grid: 'C': [0.01, 0.1, 1, 2, 3, 10, 11], 'kernel': ['rbf', 'poly', 'linear'], 'gamma': [0.01, 0.011, 0.012, 'scale', 'auto']. The parameter C controls the trade-off between maximizing the margin and minimizing the classification error, while the parameter gamma determines the influence of each training sample on the decision boundary. The hyperparameters were chosen to be within a reasonable range based on previous projects and experimentation.

3 Results

Our project placed 10th with an AUROC score of 0.89515 which is 0.04664 points behind the leading submission and 0.04612 points ahead of the baseline Ryan's Submission which was the metric to beat for this competition. This AUROC was generated using the evaluation data set provided. When locally tested our model had an AUROC of 0.864558707643814 and an accuracy 0.9354838709677419 after training and rigorous hyperparamter tuning. This lead us to believe that the model would extend well to the evaluation data set and any other data set of similar form.

4 Conclusion

We believe our model did well because we made careful design decisions for data preprocessing and model selection. We accepted the training time overhead of a very large grid search which allowed us to tune our hyperparameters very well which most importantly lead to the use of the RBF kernel.

If we wanted to improve the results of our method, we could explore other model architectures such as deep learning neural networks to see if they can better explain the complex patterns present in the data. Additionally, we could use other hyperparameter tuning methods or modify our preprocessing such that no data is removed whatsoever. Overall, we believe that our approach was effective and has great potiential for further optimization.