In-hospital Mortality Prediction using MIMIC-III Data

Xiaokun Zhang

Abstract

As data science technology advances in recent years, health care data has become one of the key objectives for study using machine learning strategies. With the consideration of providing more accurate diagnosis to patients and reducing the large amount health care cost, electronic health records (EHRs) were investigated to predict patients' health conditions and instruct the arrangements of hospital resources. A few predictive models have been developed in past decades using the public ICU data to provide retrospective prediction on patients' mortality. For example, to predict the mortality from demographical and physiological information of patients in intensive care units, the benchmark in previous literature used 17 features provided in MIMIC-III dataset. In this project, we expanded the feature set to obtain higher AUC for prediction of mortality. Also, the nature of the imbalanced dataset prevents accurate prediction of mortality. We optimized the input features in our project by various sampling methods.

Introduction

Our project is primarily based on the recently published paper by Harutyunyan et.al[3], which provides four benchmarks for clinical EHRs on MIMIC- III datasets. Two machine learning models were applied in this paper: (1) Linear regression (2) LSTM and channel-wise LSTM methods with deep supervision and multitask training. Four learning tasks were studied: (1) In-hospital mortality (2) Length of stay (LOS) (3) Decompensation rate (4) Phenotyping. This work provided a systematic study on the MIMIC–III dataset with publicly available data preprocessing methods and model baselines. Ghassemi et.al[5] studied mortality predictions using SVM model with linear kernel on MIMIC-II data. The predictions include both in-hospital and post-discharge mortality. The highlight of this study is that they included care staff notes in predictions, which were usually ignored by other studies. An LDA model was used to construct a feature matrix generated from clinical notes in temporal order. Xu et. al[6] studied the physiological signals for a patient generated at the same time from multiple channels in the intensive care unit. The signals included time-dependent ECG waveform, SPO2, pulse and other information. They used guidance from lab data to construct attention weights of multiple channels. Choi et al.[7] applied RNN models to construct Doctor AI model to study sequential clinical data which aimed to provide diagnosis at patients' visit. Zheng Dai et al. studied disease characteristics on MIMIC-III dataset and found the top ten diseases and their distributions with the largest number of patients in the dataset.[8]

In our study, we propose to extend one of the prediction benchmarks tasks, in-hospital mortality, on Medical Information Mart for Intensive Care III (MIMIC-III) dataset[1, 2, 3] provided by Harutyunyan et.al[4]. In the benchmark settings, in-hospital mortality was predicted based on the first 24 hours of an ICU stay. Accurate mortality predictions in an earlier stay of ICU admission would be extremely helpful in terms of identifying patients with high-risk of dying within hours or days in order to efficiently allocate intensive care resources. This study proposes random forest and gradient boosting approaches with additional features such as the mean, minimum, maximum values of calcium, sodium, lactate levels for in-hospital mortality prediction using the first 24-hour timeframe of patients' ICU stay and evaluates metric importance as well as model performances.

Approach

1. Database

The data source for this project is MIMIC-III database and we utilize the current version v1.4[1, 2, 3]. MIMIC-III is a freely available database including de-identified electronic health records for more than 40,000 patients who were admitted into intensive care units (ICU) of the Beth Israel Deaconess Medical Center in 2001 and 2012. This database provides us an opportunity to carry out benchmark works with reproducible and reliable data source. In this MIMIC-III v1.4 database, 26 tables were included to track patients' admission, stay, and physiological signals in ICU and list patients' demographic information.

2. Methodology

This study consists of two phases. In phase I, data ETL and feature engineering were completed using Bigquery on Google Cloud Platform. Bigquery is a serverless and multi-cloud data warehouse that has MIMIC-III data already available to use without having users set up the environment on their local machines. Output from phase I was then used as feature input in model fitting in phase II. In phase II, we built machine learning models, including random forest, gradient boosting tree and a LSTM-based model suggested by Harutyunyan et.al[4] to predict in-hospital mortality rate using Python 3 and Spark on Google Colaboratory ("Colab" for short), which provides full power of machine learning by executing code on Google's Cloud servers and allows real-time collaboration within our team.

3. Data ETL and feature engineering

By using Bigquery on Google Cloud Platform, "physionet-data" was imported into our working project. Based on the source tables, we followed the procedures as in previous publication on benchmark study[4] to filter out missing data and complicated situations in patient and clinical events to avoid errors and ambiguities in future modeling. For patients, pediatric patients and patients involved in ICU transfer or multiple stays per admission were excluded from the cohort. After this exclusion, 33046 patients were used in our next stage modeling. For ICU stays, events with missing admission ids or ICU stay ids were excluded.

Also, admission ids and ICU stay ids that cannot be found in the table for ICU stay properties were excluded too.

We have run ten SQL scripts sequentially for data ETL and each script served a specific goal in terms of feature engineering. Input, intermediate tables and a short description associated with each script are listed in Table 1.

SQL Scripts	Input tables	Intermediate tables	Description
mp_cohort.sql	Chartevents Icustays Admissions Patients	mp_cohort	Create patient cohort
mp_hourly_cohort.sql	mp_cohort	mp_hourly_cohort	Generate patients' sequence of ICU hours
mp_gcs.sql	Chartevents mp_cohort	mp_gcs	Extract Glasgow Coma Scale
mp_bg1.sql **mp_bg2.sql**	Labevents Chartevents mp_cohort	mp_bg mp_bg_art	Extract patients' blood gas and chemistry values
mp_lab.sql	Labevents mp_cohort	mp_lab	Extract patients' lab results
mp_uo.sql	mp_cohort outputevents	mp_uo	Extract patients' urine output
mp_vital.sql	mp_cohort Chartevents	mp_vital	Extract patients' vital signs
mp_data.sql	mp_hourly_cohort mp_vital mp_gcs mp_uo mp_bg_art mp_lab	mp_data	Merged all temp tables to get all features at every ICU hour for each patient
mp_data_24hr.sql	mp_data mp_cohort	mp_data_24hr	Aggregate data for the first 24-hour ICU stay as final data feed into RF and GBT model
mp_data_24hr_lstm.sql	mp_data mp_cohort	mp_data_24hr_lstm	first 24-hour ICU stay with 6 time series features as final data feed into LSTM model

Table 1: List of SQL scripts used for ETL and feature engineering

4. Building Machine learning models

Dataset after feature engineering indicates highly imbalanced label occurred (Table 2). In general, the imbalance has a severe effect on the predicted class probabilities. For example, the current settings imply the benchmark accuracy is high as 89.85% if no remedy applies, meaning that one model accuracy can jump beyond 90% only if that model can predict several positives correctly. In other words, any models with this imbalanced label can achieve good specificity but have poor precision. Since there is a priori knowledge of a class imbalance, one straightforward method to alleviate the problem is to create a training set to have roughly equal event rates during the initial data collection[9]. For simplicity, down-sampling is adopted here to randomly sample the majority class so that all classes have approximately the same size. That means we have 8294 samples (4147 for live and 4147 for death) in the final model input.

After down-sampling, missing data is imputed by median values for each feature. The total population then is split into 80% training set and testing set. Two machine learning models used here are random forest model (RF) and gradient boosting tree model (GBT). For each one of them, we had one model with default setting and one with tuned hyperparameters (hard-coded search range of hyperparameters with grid search and 5 cross-validation settings)\

	Deceased	Alive
Count	4147	36706
Percentage	10.15%	89.85%

Table 2: Label count and percentage

Metrics

In this study, AUC (Area Under the Curve), true positive rate, precision, and accuracy were used to evaluate model performances. Specifically, we focused on AUC and true positive rates for two reasons: 1) our model objective was to correctly predict the positive classes, meaning finding patients who deceased, and 2) accuracy and precision were no longer proper measures due to imbalanced datasets, since they did not distinguish between the numbers of correctly classified patients of different classes.

Experimental Results

Random forest (RF) and gradient boosting tree (GBT) models were built using Pyspark ML on Google Colab. Metric performances of each model are listed in Table 3 below. In addition, Table 4 shows the comparison for key hyperparameters between defaulted model and tuned model

Model	True Positive Rate	Precision	Accuracy
RF with default setting	0.77	0.84	0.81
Best RF after tuning and CV	0.80	0.83	0.81
GBT with default setting	0.82	0.85	0.83
Best GBT after tuning and CV	0.81	0.83	0.82

Table 3: Model performance comparison

Model	# of Trees	Max Depth	Max Bins	Min Instances/Node
RF with default setting	20	5	32	NA
Best RF after tuning and CV	30	10	20	NA

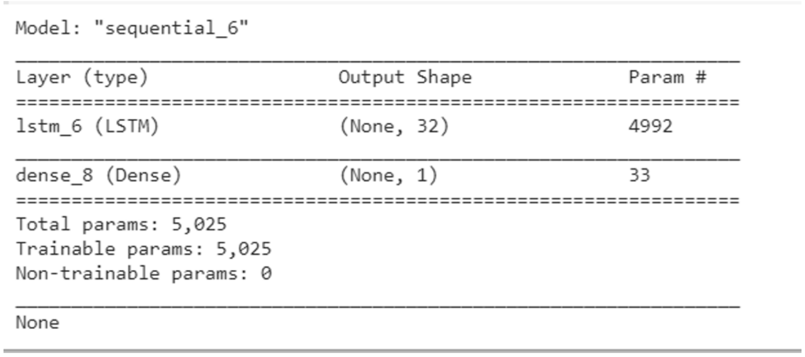
GBT with default setting	NA	5	32	1
Best GBT after tuning and CV	NA	5	30	19

Table 4: Key Hyperparameters

Overall, models with tuned parameters selected from cross validation outperformed the default model in the case of random forest. Compared to random forest, gradient boosting tree had better performance as true positive rate, precision and accuracy were all higher. ROC curves (receiver operating characteristic curve), which were used as an evaluation of classification model's performance at all thresholds, can be viewed in Figure 1.

Figure 1: ROC curve of random forest models and gradient boosting models

In addition to the traditional machine learning algorithms that use aggregated data set to make predictions, we also fit Deep learning models (LSTM), trying to capture the temporal relationship among the data. Compared with Algorithms like Random Forest (RF) and Gradient Boosting (GB), LSTM uses fewer features and still achieved good precision rate for the label of interest (Deceased patients).



```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_6 (LSTM)                (None, 32)                4992
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 33
=================================================================
Total params: 5,025
Trainable params: 5,025
Non-trainable params: 0
_________________________________________________________________
None
```

Figure 2: Model Architecture Specified for LSTM

Similar to the modeling with aggregated data, the data set extracted for LSTM modelling (time series data) is also imbalanced, posing challenge for the model to predict certain classes due to a share proportion of a particular class. In this case, most of ICU stays comes from patients who were alive at the time the data was collected. Hence, we took similar down sampling approach to sample from patients who stayed in ICU long enough to be included in the data set but were discharged from ICU as alive. After sampling, the data was processed into an array of dimension N x Timesteps x features, Where N is the number of unique ICU stays in the corresponding dataset, Timesteps is the number of inputs into LSTM model (24 hours in our case) and six time series features were used in the model.

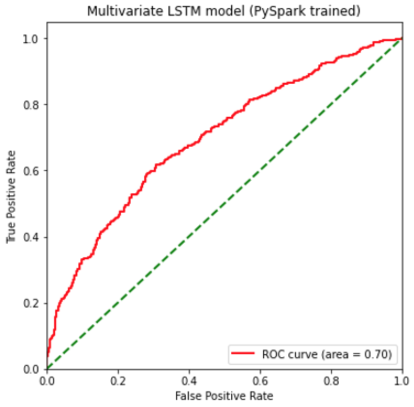


Figure 3: ROC Curve for LSTM

Table 5 shown above suggests that the LSTM has satisfactory performance in terms of precision rate for the positive label. The model is extremely useful in the situation where a classification is needed in a short time frame as it only requires six features in contrast to the aggregated features used by RF and GBM mentioned above. Also, it expected that the performance could be improved if different cohorts (e.g., focus on first 6 or 12 hours) is studied, which in turn leads to a larger sample size. In short, LSTM uses fewer number of features to generate results that could be useful in the treatment and requires less tuning.

Label	True Positive Rate	Precision	F1 Score
Alive	0.85	0.78	0.73
Deceased	0.82	0.73	0.71

Table 5: LSTM Evaluated by Different Metrics

Conclusion

The project is mainly concerned with the prediction of mortality for ICU stays on MIMIC III dataset. Data were preprocessed through google cloud computing platform and modeling part is done through Pyspark. The imbalanced nature of the data makes it difficult for the model to make accurate predictions for both classes. For example, in the data set extracted for RF and GBM, the number of observations for modeling is 40853, but only 4147 rows correspond to deceased patients. Therefore, extra steps were taken to deal with imbalanced nature of data to avoid the model paying too much attention on alive patients. Several Machine learning algorithms were fit on data and showed promising performance on key metrics defined in this context. In this study, both the algorithms that uses aggregated data and deep learning model that uses temporal data were investigated. Future research directions include the inclusion of more appropriate features, other sampling methods[9], and the use of different loss function to deal with imbalanced data.

References

1. Johnson, A., Pollard, T., & Mark, R. (2016). MIMIC-III Clinical Database (version 1.4). PhysioNet. https://doi.org/10.13026/C2XW26.
2. Johnson, A. E. W., Pollard, T. J., Shen, L., Lehman, L. H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A., & Mark, R. G. (2016). MIMIC-III, a freely accessible critical care database. Scientific Data, 3, 160035.
3. Goldberger, A., Amaral, L., Glass, L., Hausdorff, J., Ivanov, P. C., Mark, R., ... & Stanley, H. E. (2000). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation [Online]. 101 (23), pp. e215–e220.
4. H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan. Multitask learning and benchmarking with clinical time series data. Scientific data, 6(1):96, 2019.
5. Y. Xu, S. Biswal, S. R. Deshpande, K. O. Maher, and J. Sun. Raim: Recurrent attentive and intensive model of multimodal patient monitoring data. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 2565–2573. ACM, 2018.
6. M. Ghassemi, T. Naumann, F. Doshi-Velez, N. Brimmer, R. Joshi, A. Rumshisky, and P. Szolovits. Unfolding Physiological State: Mortality Modelling in Intensive Care Units. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’14, pages 75–84, New York, NY, USA, 2014. ACM.
7. E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, and J. Sun. Doctor ai: Predicting clinical events via recurrent neural networks. In Machine Learning for Healthcare Conference, pages 301–318, 2016.
8. Dai, Z., Liu, S., Wu, J., Li, M., Liu, J., & Li, K. (2020). Analysis of adult disease characteristics and mortality on MIMIC-III. PloS one, 15(4), e0232176.
9. M. Kuhn and K. Johnson, Applied Predictive Modeling, DOI 10.1007/978-1-4614-6849-3 1, © Springer Science + Business Media New York 2013