Predicting 30-day Hospital Readmission of Diabetes Patient

with Machine Learning Models

Xue “Susan” Chen

Outlines:

- Problem Statement
- Data Source & Assembly
- Data Cleaning & Exploring
- Results & Discussion:
  - -Effect of imbalanced dataset
  - -Comparison of model performance
  - -Explanation of XG Boost with SHAP values
  - -Model study on Neural Network
- Conclusions

Problem Statement

According to CDC, National Diabetes Statistics Report for 2020 cases of diabetes have risen to an estimated 34.2 million. In other word 10.5% of the U.S. population, have diabetes.

According to literature, patients with diabetes have almost double the chance of being hospitalized than the general population.

My project will focus on predicting hospital readmission for patients with diabetes. I will use predictive modeling to help identify these readmissions.

Data Source & Assembly

UCI data

50 attributes of medical records
Including emergency, outpatient, inpatient
Demographics (age, sex, race)
Diagnoses (ICD-9-CM codes)
In-hospital procedures
Laboratory data, pharmacy data
In-hospital mortality

UCI Machine Learning Repository
(https://archive.ics.uci.edu/ml/datasets/Diabetes+130-US+hospitals+for+years+1999-2008)

MIMIC data

Multiple tables
Including ultrasonic images
Natural Language Processing

MIMIC CXR database
(https://physionet.org/content/mimic-cxr/2.0.0/#files)

Data Cleaning & Exploring

UCI data

All patients diagnosed of diabetes
Only the first encounter was chosen for each patient
Remove encounter related to patient death

Feature engineering include:
Categorical to numerical
Reclassifying categorical data
Transforming numerical to log
Labeling positive class as readmitted within 30 days

MIMIC data

All patients diagnosed of diabetes
Only the first encounter was chosen for each patient
Remove encounter related to patient death

Collecting only discharge summary of each patient
Cleaning discharge summary
Labeling positive class as readmitted within 30 days

Results & Discussion:

-Effect of imbalanced dataset UCI data(12 features)

Model Name	Accuracy	Sensitivity	Specificity	Precision
Random Forest	90.48%	0	1	0
Multinomial Naïve Bayes	88.32%	3.24%	96.57%	23.21%
Support Vector Machine	90.48%	0	1	0
XG Boost	90.48%	0.76%	99.92%	0.76%

Baseline of imbalanced data: 90.48%

Model Name	Accuracy	Sensitivity	Specificity	Precision
Random Forest	67.05%	77.77%	56.33%	64.04%
Multinomial Naïve Bayes	60.32%	68.84%	51.55%	58.78%
Support Vector Machine	56.18%	71.91%	45.02%	55.05%
XG Boost	86.53%	81.99%	91.08%	90.18%

Baseline of balanced data: 50.0%

SMOTE

Results & Discussion:

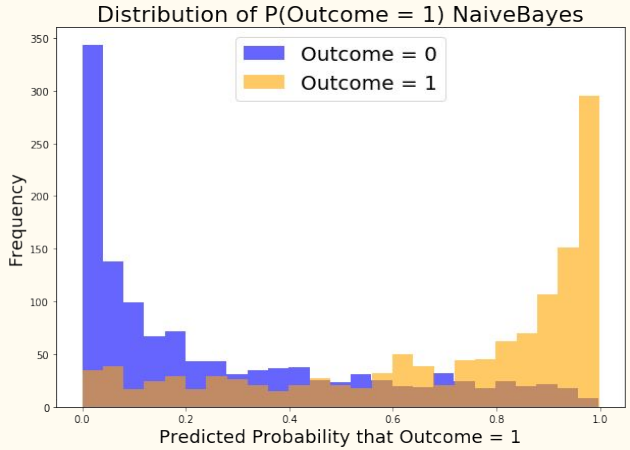
-Comparison of model performance

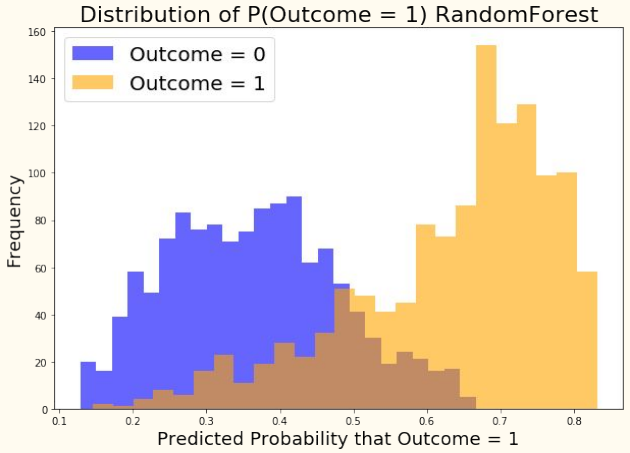
UCI data -- Metrics of models

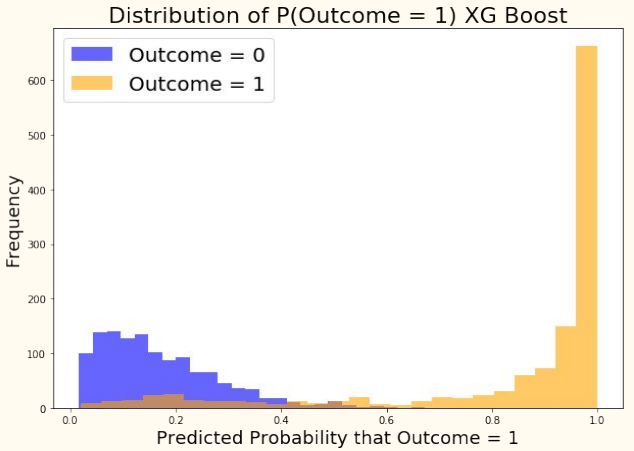
Baseline = 50%
Features = 38

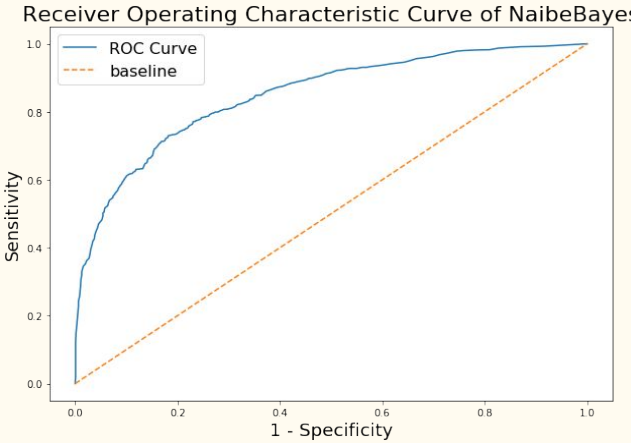
Model Name	Acc(Train)	Acc(Test)	Sensitivity	Specificity	Precision
Random Forest	85.15%	84.82%	82.71%	86.93%	86.36%
Multinomial Naïve Bayes	77.37%	76.65%	75.06%	78.25%	77.53%
Support Vector Machine	74.23%	72.00%	78.41%	65.58%	69.49%
XG Boost	93.72%	92.47%	86.69%	98.25%	98.02%

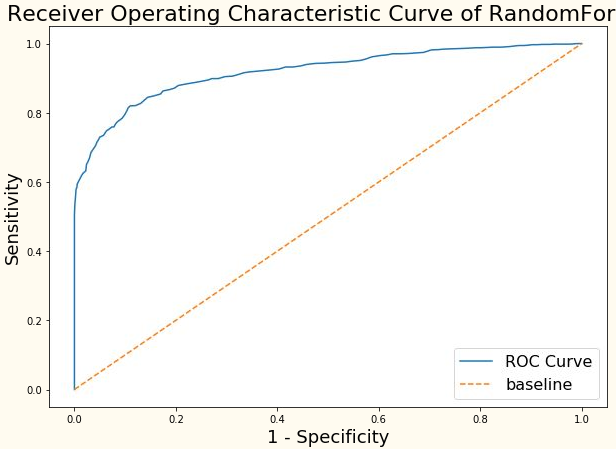
UCI data --Distribution of probability & ROC curve

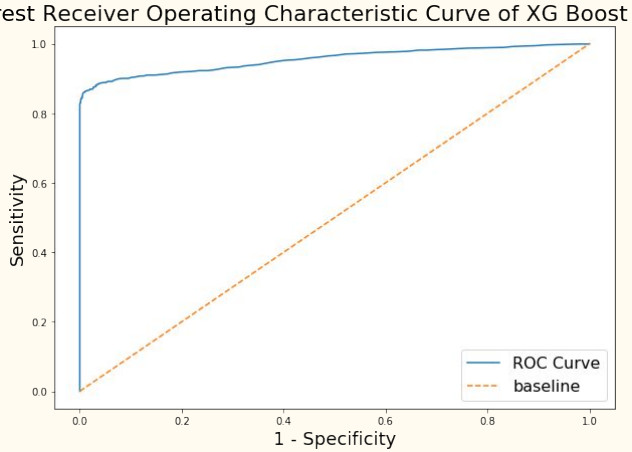












Results & Discussion: Comparison of model performance

MIMIC data -- Metrics of models

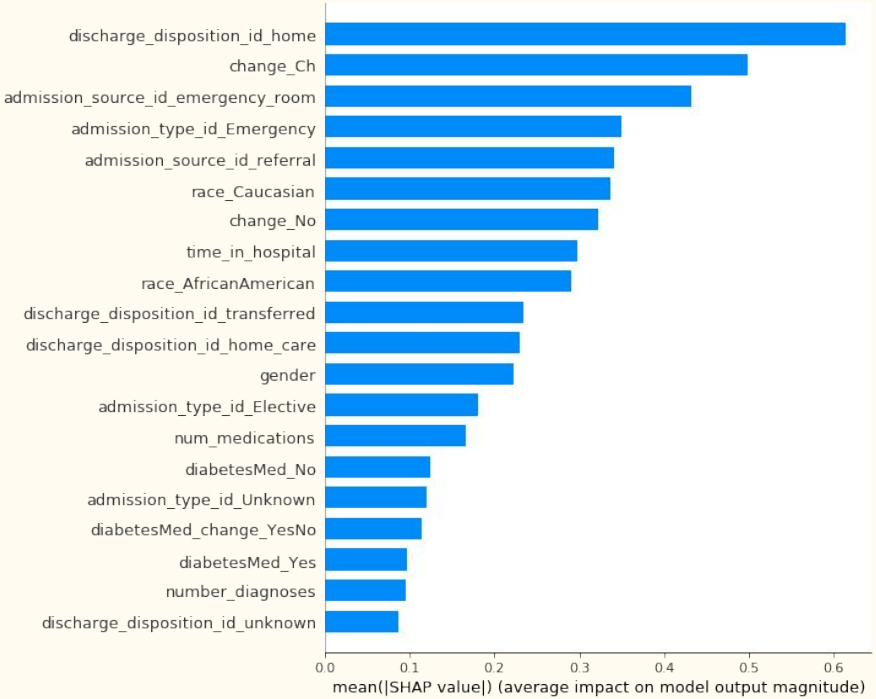
Baseline = 50%

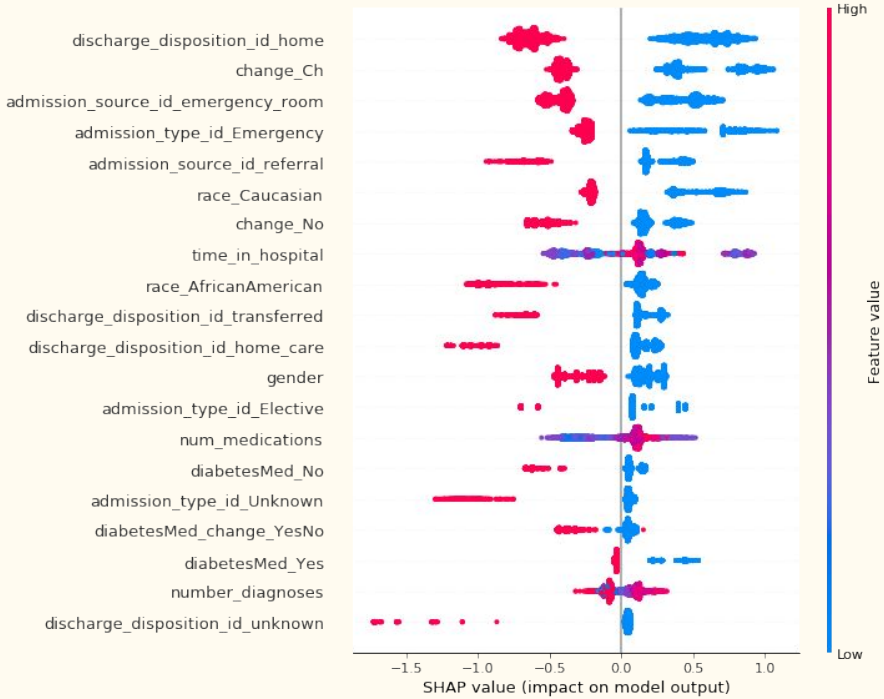
Model Name	Acc(Train)	Acc(Test)	Sensitivity	Specificity	Precision
Random Forest (cvec)	69.80%%	64.24%%	68.21%	60.26%	63.19%
Random Forest (Tfidf)	79.04%	78.58%	83.44%	73.71%	76.04%
Multinomial Naïve Bayes (cvec)	64.98%	61.92%%	63.44%	60.40%	61.57%
Multinomial Naïve Bayes (Tfidf)	61.49%	54.97%	53.77%	56.16%	55.09%
Support Vector Machine (cvec)	65.75%	63.54%	66.03%	61.06%	62.90%
Support Vector Machine (Tfidf)	66.25%	64.37%	61.72%	67.02%	65.17%
XG Boost (cvec)	95.81%	96.09%	92.19%	100%	100%
XG Boost (Tfidf)	94.54%	90.93%	85.50%	96.36%	95.91%

Results & Discussion:

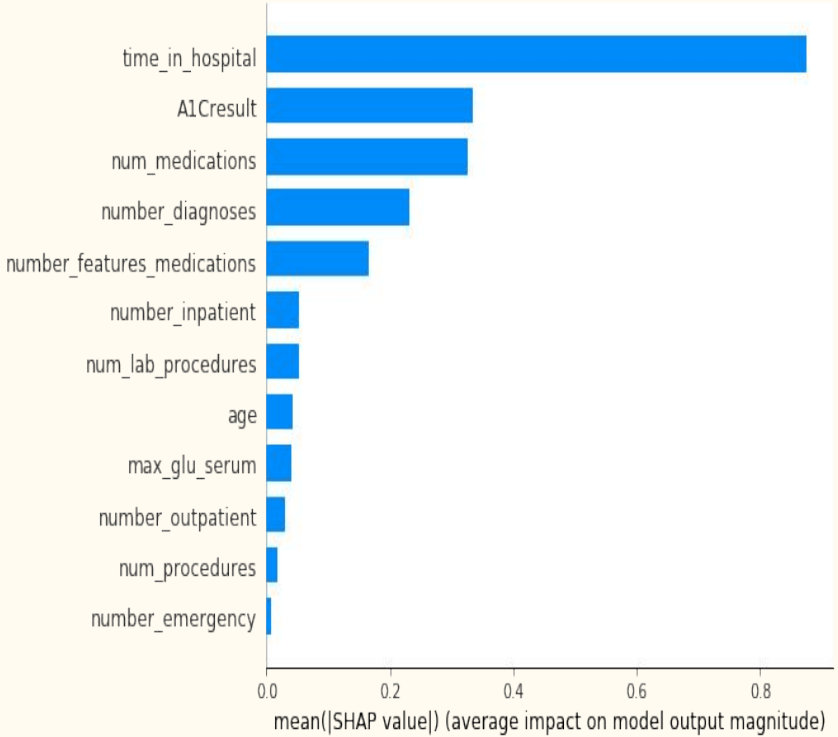
-Explanation of XG Boost with SHAP values

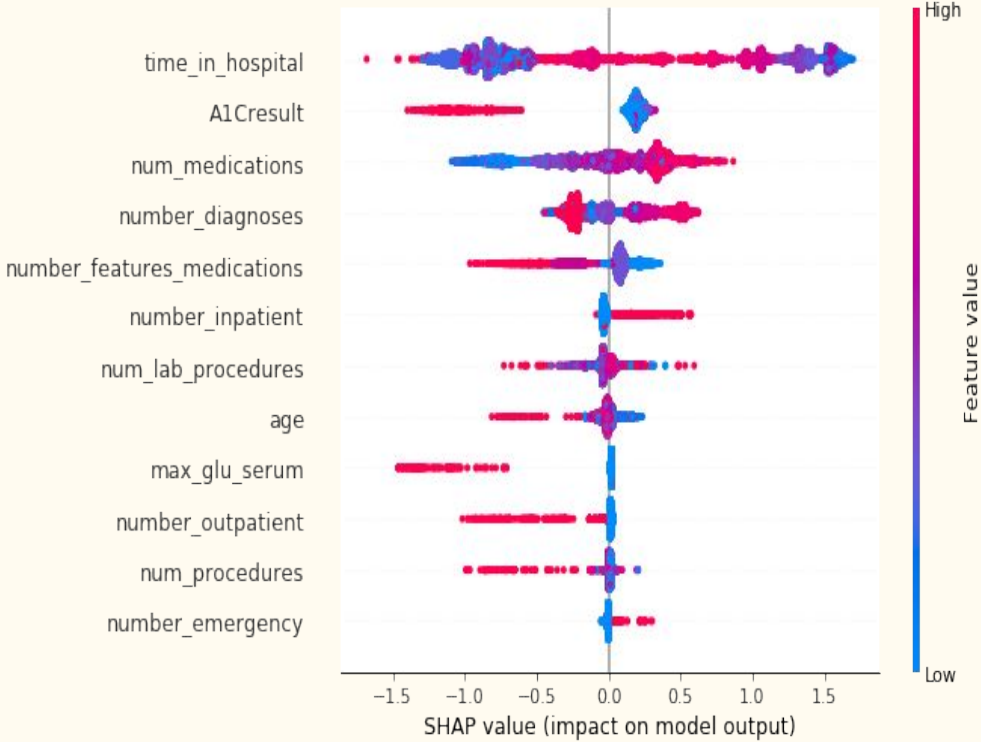
UCI data -- 38 features





UCI data -- 12 numerical features

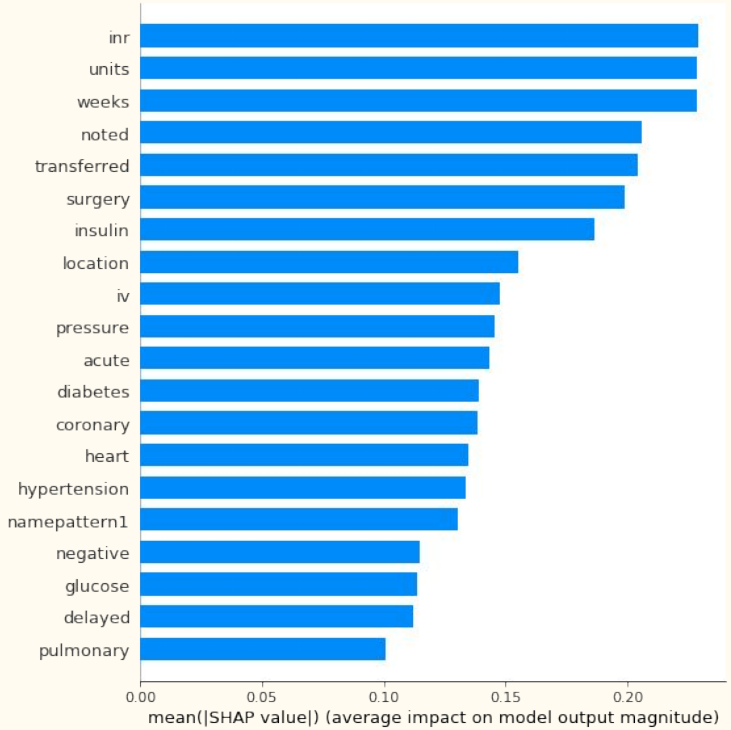


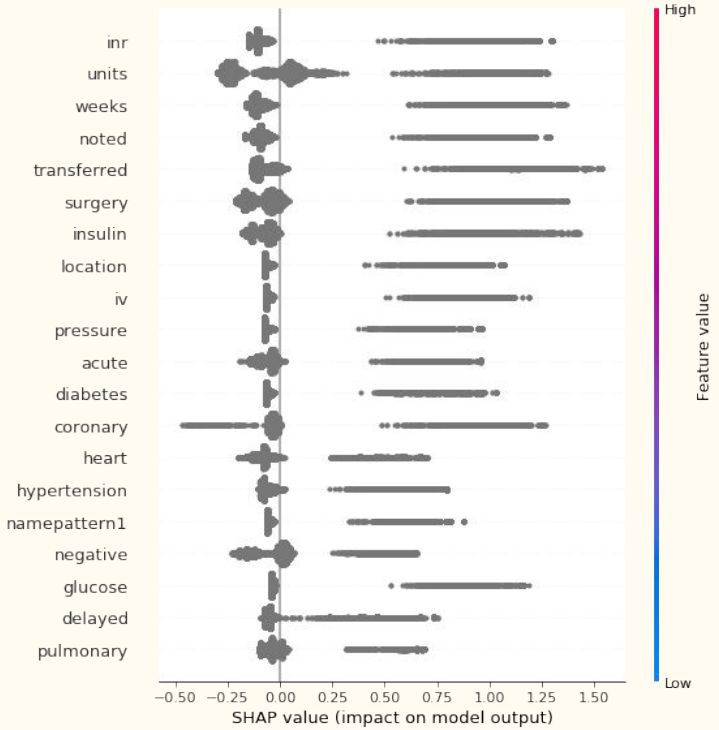


Results & Discussion:

-Explanation of XG Boost with SHAP values

MIMIC data





Results & Discussion:

-Model study on Neural Network

UCI data

Baseline = 50%

Metrics	Random Forest	Multinomial Naïve Bayes	Support Vector Machine	XG Boost	Neural Network
Accuracy (Train)	85.15%	77.37%	74.23%	93.72%	78.01%
Accuracy (Test)	84.82%	76.65%	92.47%	92.47%	79.40%

Results & Discussion:

-Model study on Neural Network

MIMIC data -- NLP

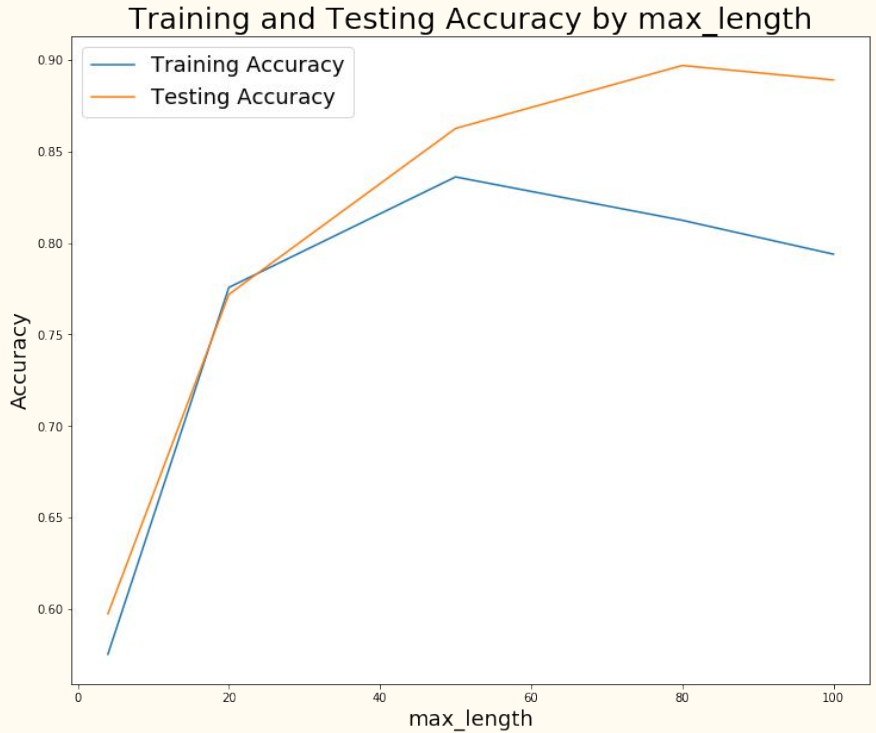
Baseline = 50%

Metrics	Random Forest	Multinomial Naïve Bayes	Support Vector Machine	XG Boost	Neural Network
CountVectorizer Accuracy (Train)	69.80%	64.98%	65.75%	95.81%%	73.27%
CountVectorizer Accuracy (Test)	64.24%	61.92%	63.54%	96.09%	68.54%
TfidfVectorizer Accuracy (Train)	79.04%	61.49%	66.25%	94.54%	58.27%
TfidfVectorizer Accuracy (Test)	78.58%	54.97%	64.37%	90.93%	57.28%

Results & Discussion:

-Model study on Neural Network

MIMIC data -- NLP with embedding layer and hidden layer



- Tokenizer -count by frequency
- Max_length -number of words selected in each row
- Trained embedding matrix from GloVe
- Trained embedding matrix with 50 dimensions
- Hidden layer with 64 neurons

MIMIC data -- NLP with embedding layer and hidden layer

Metrics	Neural Network
Without Embedding Accuracy (Train)	73.27%
Without Embedding Accuracy (Test)	68.54%
With Embedding matrix 50 dim Accuracy (Train)	81.24%
With Embedding matrix 50 dim Accuracy (Test)	89.70%
With Embedding matrix 100 dim Accuracy (Train)	75.31%
With Embedding matrix 100 dim Accuracy (Test)	88.97%

Baseline = 50%

Conclusions:

1. Due to the less information from the minority class, machine learning models couldn't predict the minority class efficiently. By using SMOTE to balance the dataset can solve this problem in my study.
2. XG Boost showed much better results than other models.
3. The features in UCI data and MIMIC data were studied with SHAP values. Many useful informations were obtained from analyzing the value of each features.
4. Neural Network didn't show success in UCI dataset. But neural network did show some significant improvement in NLP data(MIMIC dataset) when I induced the embedding matrix.