CS6220 Data Mining - Final Project

So Man Amanda Au Yeung
Yian Chen
Chun Wei Tseng

https://github.com/Chun-Wei-Tseng/CS6220-FinalProject

1. Introduction

The rising problem of early readmissions among diabetic patients presents formidable challenges for healthcare, resulting in complications and financial penalties. Our project seeks to forecast the combination of features that contribute to early hospital readmissions within a 30-day window, with a focus on addressing gaps in preventive care and enhancing patient outcomes. In contrast to conventional models, we delve into a range of machine learning approaches to gain deeper insights into the factors influencing early readmissions, creating a synergy between healthcare and data mining.

2. Background

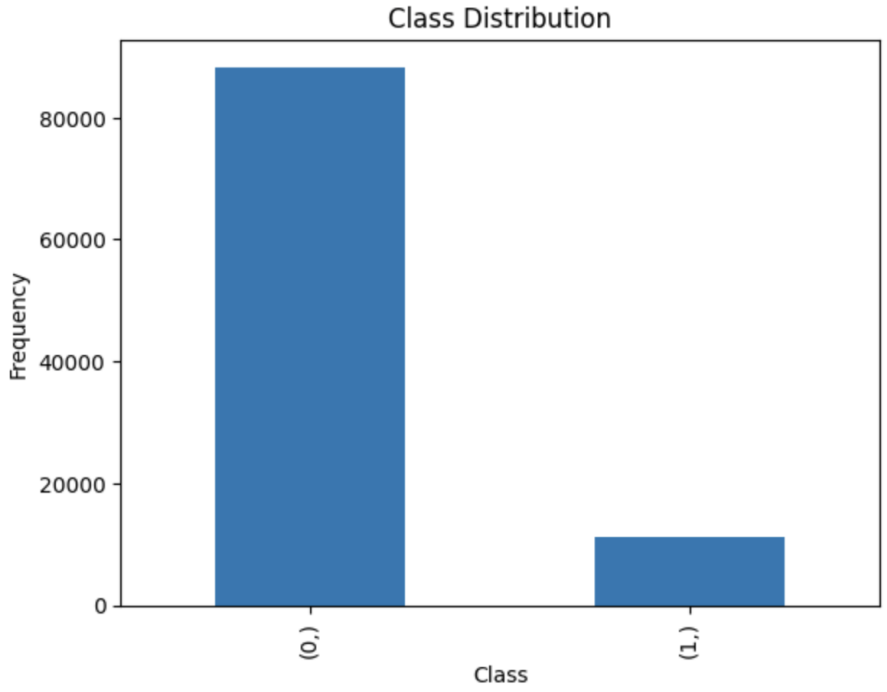
Diabetic patients frequently experience inadequate care, leading to complications and heightened hospital expenditures. The Center for Medicaid & Medicare Services (CMS) penalizes hospitals for elevated readmission rates, underscoring the financial impact and influencing quality ratings. Our project is dedicated to forecasting features that contribute early readmissions (within 30 days) for diabetic patients, addressing gaps in preventive care. In contrast to traditional regression models, we explore diverse models to gain a nuanced understanding of the factors contributing to early readmission. Positioned at the intersection of healthcare and data mining, this project aims to elevate patient outcomes, reduce costs, and optimize diabetes care practices.

3. Approach

3.1 Data Analysis

Our dataset encompasses 101,766 patient records, each described by 47 attributes. The target variable 'y' denotes whether a patient was readmitted within 30 days post-discharge. During feature selection, attributes with substantial missing data were omitted to ensure the integrity of the analysis. Medication-related features exhibiting negligible correlation using Pearson's correlation with the target variable were also excluded to sharpen the predictive power of the model. The dataset initially included primary (diag_1), secondary (diag_2), and tertiary (diag_3) diagnosis categories, which include hundreds of unique values that represent hospitalization utilization (diagnosis) codes. An initial attempt to incorporate these through feature hashing was made, but this technique expanded the feature space without

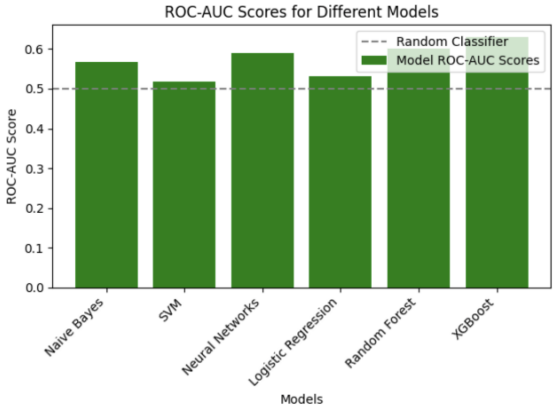
yielding predictive performance gains. Upon conducting some research, we focused exclusively on the primary diagnosis as we found it could enhance model accuracy while reducing complexity. This approach aligns with findings from an extensive clinical database analysis, which underscores the significance of the primary diagnosis in readmission rates (Strack,DeShazo,Gennings,Olmo,Ventura,Cios,Clore, 2014, Year)[1]. In our analysis, we refined the feature set by retaining diagnoses closely associated with diabetes, including specific endocrine and cardiovascular conditions known to impact diabetic patients. Our exploratory data analysis revealed a significant class imbalance: only approximately 11% of instances represented early readmissions, while the majority, over 88%, corresponded to non-early or no readmissions. To address this imbalance, we implemented two primary strategies: Synthetic Minority Oversampling Technique (SMOTE) to increase the minority class, and random undersampling to reduce the majority class. Furthermore, we utilized stratified k-fold cross-validation to ensure proportional representation of each class across all folds to help achieve a more balanced and robust model training process.



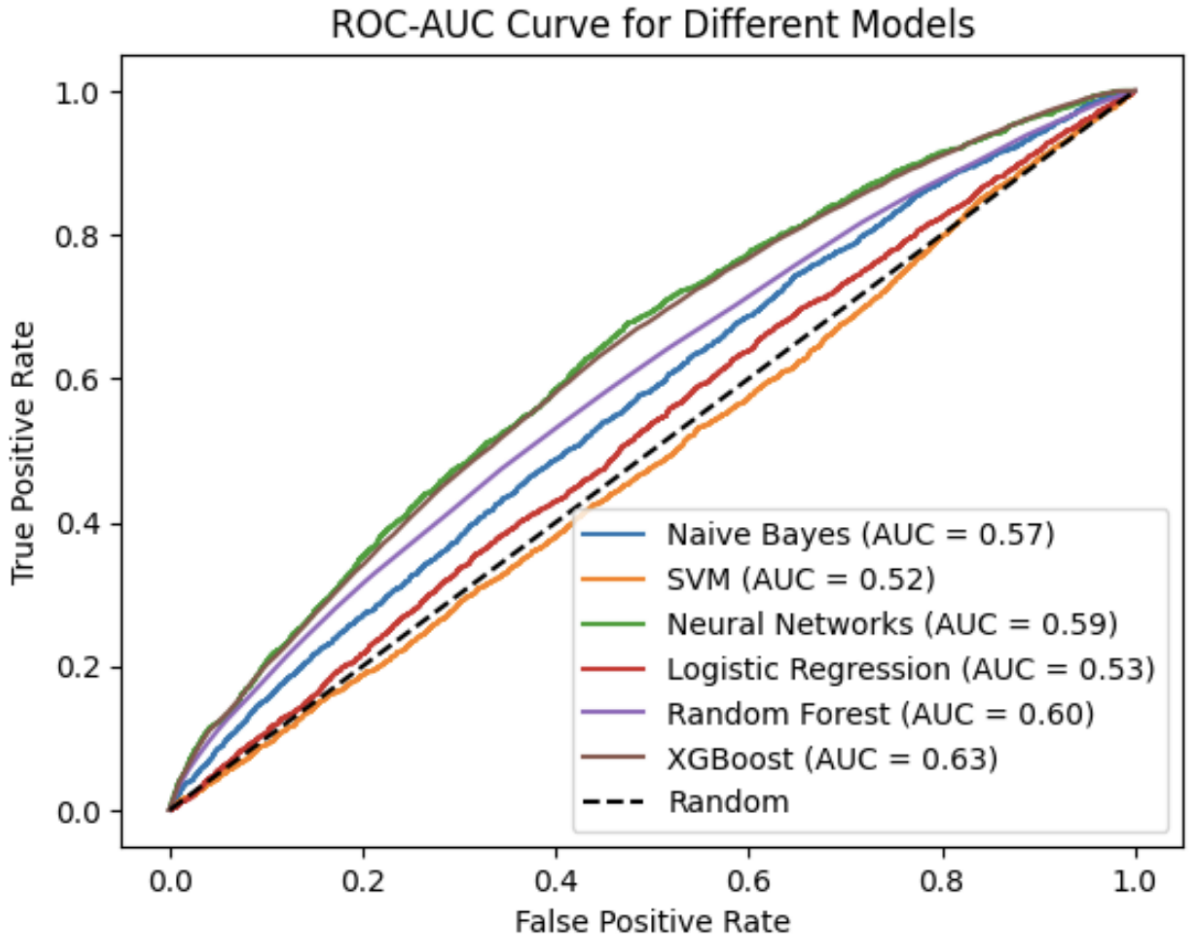
All of the six models we trained on displayed similar accuracy of around 88%. However, we do find that the models are apt to predict the majority class and may not be a good reflection on their performance on predicting the minority class. Using other evaluation metrics like F1 score, ROC-AUC score, we find that our models generally displayed moderate values of F1 score, precision, and recall, which suggest that there is room for improvement in correctly identifying true positives and reducing false positives and false negatives. The discrepancies between high accuracy and lower ROC_AUC scores for some models suggests that accuracy alone could be misleading due to class imbalance.

[1] (Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records, Strack,DeShazo,Gennings,Olmo,Ventura,Cios,Clore, 2014).

- Accuracy:
  - Naive Bayes: 88.76%
  - SVM: 88.76%
  - Neural Networks: 88.76%
  - Logistic Regression: 88.54%
  - Random Forest: 89%
  - XGBoost: 88.78%
- AUC for ROC (Receiver Operating Characteristic):
  - Naive Bayes: 56.75%
  - SVM: 51.87%
  - Neural Networks: 59%
  - Logistic Regression: 51%
  - Random Forest: 60%
  - XGBoost: 62.93%



3.2 Implementation



For our implementation, we trained the data with six models, namely Naive Bayes, SVM, Neural Networks, Logistic Regression, Random Forest, and XGBoost and compared the results of these models. We found that SVM and Random Forest models were sensitive to the data imbalance. The Neural Networks model had a tendency to overfit, so we had to fine-tune the parameters and regularize it.

Model Optimization Approaches

Optimizing the model with grid search method might not work for all the models:

- When we were trying to optimize the SVM using grid search, the runtime turned out to be too long for computing. When the C parameter in the SVM is set as 10, it took extremely long to run.

SVM Objective Function:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\max(0, 1 - y_i(\mathbf{w}\cdot\mathbf{x}_i + b))$$

Where:

- $\mathbf{w}$ is the weight vector.
- b is the bias term.
- C is the regularization parameter.
- $\mathbf{x}_i$ is the feature vector of the i-th training example.
- y_i is the class label of the i-th training example.

Within the neural network layer, the results have shown overfitting. Therefore, the dropout layer is incorporated within the keras layer to randomly set the input unit to 0 with a frequency of rate at each step during training time, which prevents overfitting. The incorporated dropout layer with a rate of 0.2 has shown significant improvement, compared to the initial model without. It was also tested with different learning rate, number nodes, batch size, drop out rate to search for the least loss model. However, the run time took too long to finish the different combinations of different parameters (learning rate, number nodes, batch size, drop out rate). Thus, a compromised approach was used with a random choice of each value every time with iterations. After using the compromised approach, the parameters were reduced to make sure all combinations were tested for accuracy.

Later on, regularization l1 was added instead of the dropout layer towards the neural network. The data was far less overfitting and the slope of the curve diminishes,which has faster convergence reaching maximum learning ability.

SMOTE and stratifiedKFold was also used in Neural Network, but it has shown little to no improvement.

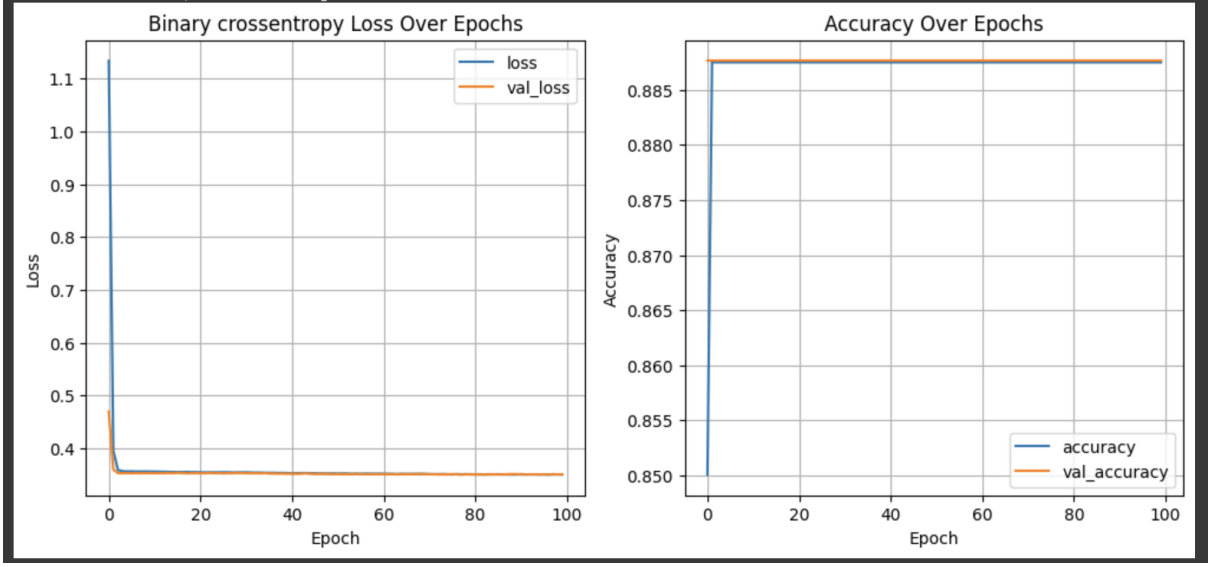
Neural Network objective function:

Binary Cross-Entropy (Log Loss): $-\frac{1}{n}\sum_{i=1}^{n}[y_i\log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$

$$\text{Binary Cross-Entropy Loss with L1 Regularization} = -\frac{1}{n}\sum_{i=1}^{n}[y_i\log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)] + \lambda\sum_j |w_j|$$

- w_j represents the weights in the network,
- λ is the regularization strength.

```
Best Hyperparameters:
{'num_nodes': 16, 'dropout_prob': 0, 'lr': 0.001, 'batch_size': 128, 'epochs': 100}
Test Loss: 34.86%, Test Accuracy: 88.76%
```



In logistic regression, gradient descent was used to optimize the model. However, the improvement in ROC was only 1%.

Logistic regression objective function:

$$\hat{\mathbf{y}}=\sigma(W\mathbf{x}+\mathbf{b})$$

The parameters that you need to optimize for are $W\in\mathbb{R}^{mxd}$, a matrix of size $m\times d$ and $b\in\mathbb{R}^{m}$, a vector of size m.

Cross-Entropy Cost Function is defined as:

$$L(\{(\mathbf{x},\mathbf{y})\},W,\mathbf{b})=-\sum_i \mathbf{y}^{(i)T}\log\hat{\mathbf{y}}^{(i)}+(1-\mathbf{y})^{(i)T}\log(1-\hat{\mathbf{y}}^{(i)})$$

Gradient of Cross-Entropy with respect to W

$$\nabla_W L=\sum_i\left(\sigma(W\mathbf{x}^{(i)}+\mathbf{b})-\mathbf{y}^{(i)}\right)\mathbf{x}^{(i)T}$$

This should yield a matrix $\mathbb{R}^{mxd}$, where m is the length of a label and d is the length of a feature vector. If $\hat{\mathbf{y}}$ is the prediction, then

$$\nabla_W L=\sum_i\left(\hat{\mathbf{y}}^{(i)}-\mathbf{y}^{(i)}\right)\mathbf{x}^{(i)T}$$

Gradient of Cross-Entropy with respect to b

$$\nabla_b L=\sum_i\left(\sigma(W\mathbf{x}^{(i)}+\mathbf{b})-\mathbf{y}^{(i)}\right)$$

This should yield a matrix $\mathbb{R}^m$, where m is the length of a label. If $\hat{\mathbf{y}}$ is the prediction, then

$$\nabla_b L = \sum_i \left(\hat{\mathbf{y}}^{(i)} - \mathbf{y}^{(i)}\right)$$

Naming convention of dimensions
N is the number of data samples (or batch size)
m is the number of labels
d is the number of features
W is the weight matrix of size $m \times d$
b is the offset vector of size m

Random Forest Optimization

Random forest generally uses Gini impurity index or entropy to measure the impurity of a node. Scikit Learn's library Random Forest Classifier uses Gini index as a default, which appears in the following form:

$$\text{Gini} = 1 - \sum_{j=1}^{J} p_j^2$$

For our random forest optimization, we trained the model using k-fold cross validation at first, then tried other optimization methods such as SMOTE oversampling with stratified k-fold cross validation, and also rebalancing the weights of the predictors to help improve the model. Random Forest models are generally more resistant to model overfitting due to its ensemble learning nature. However, the ROC-AUC performance is not particularly good, it is still displaying sensitivity to the imbalance of our data.

XGBoost Optimization and Objective function

$L(\theta) = \sum_i l(y_i, \hat{y}_i) + \sum_k \Omega(f_k)$ where $l(y_i, \hat{y}_i)$ is the loss function that measures the difference between the predicted value and the actual label and $\sum_k \Omega(f_k)$ is the regularization term. In XGBoost, this term penalizes the complexity of the model, which includes the number of leaves and the magnitude of the leaf scores in each tree in the ensemble.

In order to optimize the XGBoost model, we employed SMOTE oversampling and stratified K-fold cross validation to balance our dataset, as well as using gradient search cross validation to find the optimal parameters to improve model performance. The regularization in XGBoost helps prevent overfitting. The XGBoost model can also handle imbalance data better than the other models, and due to its ensemble learning nature it can also perform better at classification problems than other single models, which is shown in our model training results.

4. Results & Evaluation

Given high accuracy across all models, it suggests the dataset may be imbalanced and the models might be predicting the majority class well only. Since we are working on a binary classification problem, especially with imbalanced classes. AUC is useful as it is independent of the threshold set for classification and provides an aggregate measure of performance across all classification thresholds. We employed the use of ROC-AUC as our main evaluation metric which better measures a model's ability to distinguish between the classes. The discrepancy between high accuracy and lower ROC-AUC scores for some models suggests that accuracy alone could be misleading due to class imbalance. The models excel in predicting the majority class but struggle with the minority class.
To evaluate the confidence in our models, we need to look at the aggregate of other metrics, including F1 score, precision, and recall. The moderate scores of these metrics for our models suggest that there is room for improvement, particularly in correctly identifying true positives and reducing false positives and false negatives, which again shows the imbalance of the dataset affects the overall predictions. In addition, our employment of cross-fold validation also shows consistent performance across the models, which is also an indicator of higher confidence.

We have found that XGBoost outperformed other models in ROC-AUC, indicating its robustness in handling imbalanced datasets. Its built-in regularization methods and ensemble learning approach contribute to superior generalization capabilities.

In addition, SVM faced computational challenges during grid search optimization, highlighting potential limitations in scalability. Neural Networks, while achieving accuracy, showed overfitting issues initially, mitigated by dropout layers and a compromised hyperparameter tuning approach.

Overall, the dataset's characteristics significantly influenced model performance. XGBoost's superiority diminished when tested on a small batch of a balanced dataset, emphasizing the importance of model selection based on dataset characteristics. Our confidence is bolstered by transparent documentation, a metric-aware approach, and a readiness to iteratively enhance our models based on observed limitations.

- Naive Bayes:
    - Accuracy: 88.76%
    - Area under curve for Receiver Operating Characteristic: 56.75%

- Support Vector Machine (SVM):
    - Accuracy: 88.76%
    - Area under curve for Receiver Operating Characteristic: 51.87%

- Neural Networks:
    - Accuracy with least loss model: 88.75%
    - Area under curve for Receiver Operating Characteristic: 60%

- Logistic Regression
    - Accuracy: 88.74%

    - Area under curve for Receiver Operating Characteristic: 51%

- Random Forest:
    - Accuracy: 89%
    - Area under curve for Receiver Operating Characteristic: 60%
- XGBoost:
    - Accuracy: 88.78%
    - Area under curve for Receiver Operating Characteristic: 62.93%

Comparison between methods:
- Naive Bayes and SVM exhibit lower ROC-AUC scores.
- Neural Networks and Logistic Regression show moderate ROC-AUC scores.
- Random Forest demonstrates a balanced performance.
- XGBoost outperforms other models with the highest ROC-AUC.

5. Conclusions

All of the models we used have accuracy of predictions on the testing dataset around 88% or above, which seems to be really good. However, when we take the ROC-AUC (Area under curve for the Receiver Operating Characteristic), most of the models didn't receive a high score. Having ROC-AUC below 70, or even close to 50 or 60, for binary classification models means that they have poor performance. This can be caused by the unbalanced amount of targeting values ("Readmitted to hospital, 0 or 1"). XGBoost has benefits in this case. Compared to other models, it handles dataset with missing or unbalanced values a lot better. During the training process, it combines multiple weak learners to create a stronger learner, which typically has better generalization capability and is less likely to be overfitted. XGBoost has built in regularization methods to handle missing values and imbalance data. When taking a small badge of a balanced dataset for testing, the result changed. Other models that have low ROC-AUC because of unbalanced dataset performed better now. On the other hand, XGBoost decreases in performance.
In practical applications, leveraging XGBoost's resilience in healthcare predictions could enhance decision-making. Reflecting on these findings, future directions could involve refining model parameters, exploring ensemble methods, and integrating domain-specific features to further enhance accuracy. Additional time investment in these areas, coupled with staying abreast of evolving literature on healthcare data mining, especially in explainability and real-time data integration, holds promise for advancing predictive modeling in healthcare.

Citation:

- Centers for Disease Control and Prevention, “By the Numbers: Diabetes in America,” Centers for Disease Control and Prevention, 2022. https://www.cdc.gov/diabetes/health-equity/diabetes-by-the-numbers.html
- Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records, Strack, DeShazo, Gennings, Olmo, Ventura, Cios,Clore, 2014.
- Clore,John, Cios,Krzysztof, DeShazo,Jon, and Strack,Beata. (2014). Diabetes 130-US hospitals for years 1999-2008. UCI Machine Learning Repository. https://doi.org/10.24432/C5230J.
- “Regularization in Machine Learning || Simplilearn,” Simplilearn.com. https://www.simplilearn.com/tutorials/machine-learning-tutorial/regularization-in-machine-learning#:~:text=ProgramExplore%20Program-
- T. Kanstrén, “A Look at Precision, Recall, and F1-Score,” Medium, Oct. 31, 2020. https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec