Canadian Hospital Re-admittance Challenge Project Report

Team : JUST BINARY SEARCH!
Adithya Sunilkumar (IMT2021068)
Kevin Adesara (IMT2021070)
Anant Ojha (IMT2021102)

December 17, 2023

Contents

1 Introduction

1.1 Background

The "Canadian Hospital Re-admittance Challenge" aims to reduce costly and disruptive hospital readmissions by applying data science and predictive modeling to identify at-risk patients. Analyzing a diverse dataset of patient information, this initiative seeks to enhance healthcare efficiency and patient outcomes through targeted interventions and improved discharge planning. It represents a crucial step in addressing the challenges of healthcare delivery and quality in Canada.

1.2 Objectives

1. Develop predictive models to identify high-risk patients for hospital readmission.
2. Improve healthcare efficiency by providing targeted interventions and personalized care plans.
3. Reduce hospital readmission rates and enhance patient outcomes in the Canadian healthcare system.

2 Data Preprocessing

2.1 Dataset Description

This dataset consists of patient information, including unique encounter and patient identifiers, demographic details (race, gender, age), medical history (diagnoses), medication prescriptions, and indicators of changes in medication dosage. It also includes information about the timing and types of healthcare encounters, such as outpatient and emergency visits, as well as the primary outcome variable "readmission_id" indicating the days to inpatient readmission.

2.2 Data Cleaning

During the data preprocessing phase, we identified several columns that were either unrelated to our analysis or had a significant number of null values. As a result, we made the decision to exclude the following columns from the dataset, because they consisted of more than 30% null values:

- weight
- medical_specialty
- payer_code
- max_glu_serum
- A1Cresult

Drug Columns: All the columns related to drugs provided little to no information about the target variable, so they were also dropped. Upon calculating the mode and its frequency for the drug columns, we noticed around 99% of entries would contain the same value. Later, we will describe how this data was modified into three features to make it more useful.

2.3 Data Encoding

As part of data preprocessing, we utilized Label Encoding to transform non-numeric categorical variables into numeric representations for improved compatibility with machine learning models. The following columns were encoded:

- **race**: Encoded the 'race' column to represent different racial categories as numeric values.
- **age**: Transformed age groups into numeric values for use in our analysis.
- **gender**: Encoded gender as numerical values, with 'male' and 'female' represented accordingly.
- **diabetesMed**: Converted 'diabetesMed' to a binary numeric representation, where 'yes' is represented as 1 and 'no' as 0.
- **change**: Transformed the 'change' column into a binary numeric variable, with 'change' as 1 and 'no change' as 0.
- **diag_1**: Encoded primary diagnosis codes ('diag_1') into numerical format.
- **diag_2**: Applied Label Encoding to secondary diagnosis codes ('diag_2') to enable numeric analysis.
- **diag_3**: Encoded additional secondary diagnosis codes ('diag_3') for numeric compatibility.

Label Encoding facilitated the integration of these categorical variables into our machine learning models, allowing us to perform quantitative analyses effectively.

2.4 Handling Missing Data

During data preprocessing, we implemented a strategy to handle missing values effectively. The following procedures were applied:

1. Rows with More Than 2 Null Values: Rows with more than two null values were identified and subsequently dropped from the dataset. This decision was made to ensure the retention of data points with a sufficient amount of information for analysis.
2. Replacement with Mode: For columns where null values were observed but did not exceed the threshold, the missing values were replaced with the mode of the respective column. Utilizing the mode helped maintain the integrity of the dataset while addressing missing data points in a meaningful way.

These steps ensured that the dataset remained suitable for subsequent analysis while minimizing the impact of missing data on our results. The initial row count of the dataset was 71,236. After the removal of rows with more than two null values, the row count was reduced to 71,225.

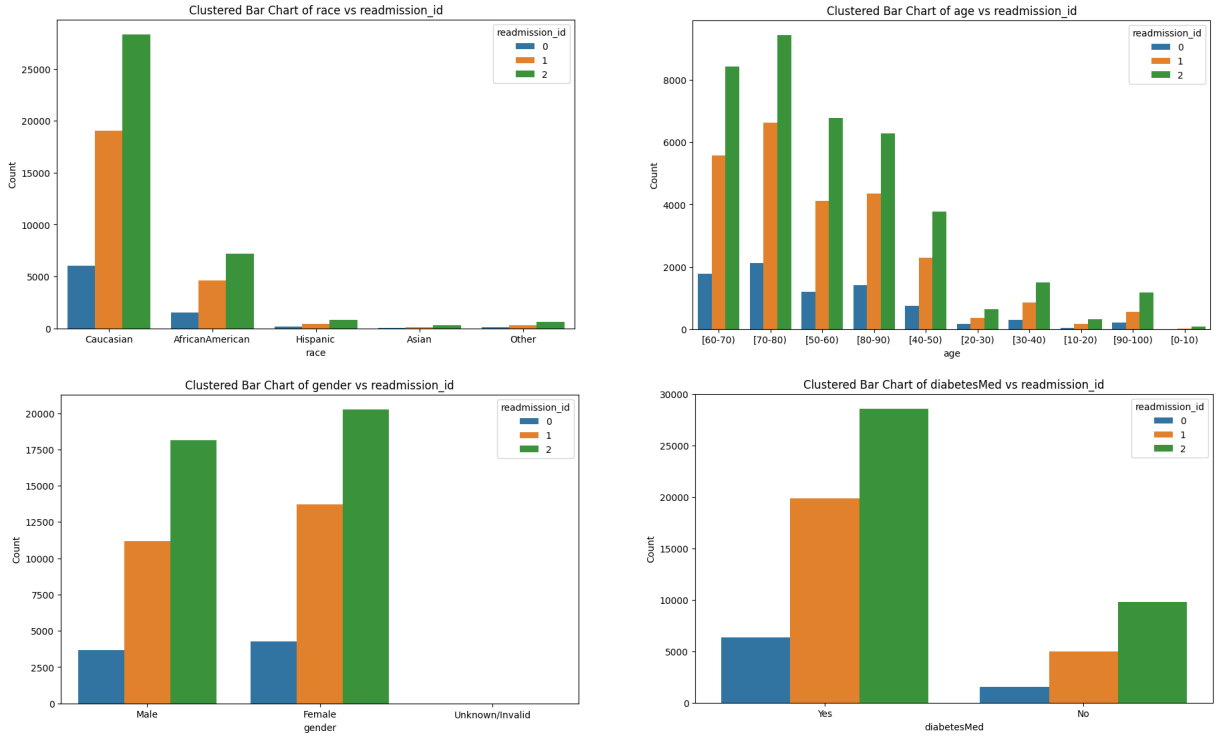
3 Exploratory Data Analysis (EDA)

3.1 Data Summary

The dataset contains a total of 71,236 records, each representing a patient encounter. It encompasses various features, including demographic details, medical history, medication information, and healthcare encounter attributes. The primary outcome variable, "readmission_id," categorizes patient encounters by days to inpatient readmission, with values 0, 1, and 2 indicating readmission within 30 days, beyond 30 days, and no recorded readmission, respectively.

3.2 Data Visualization

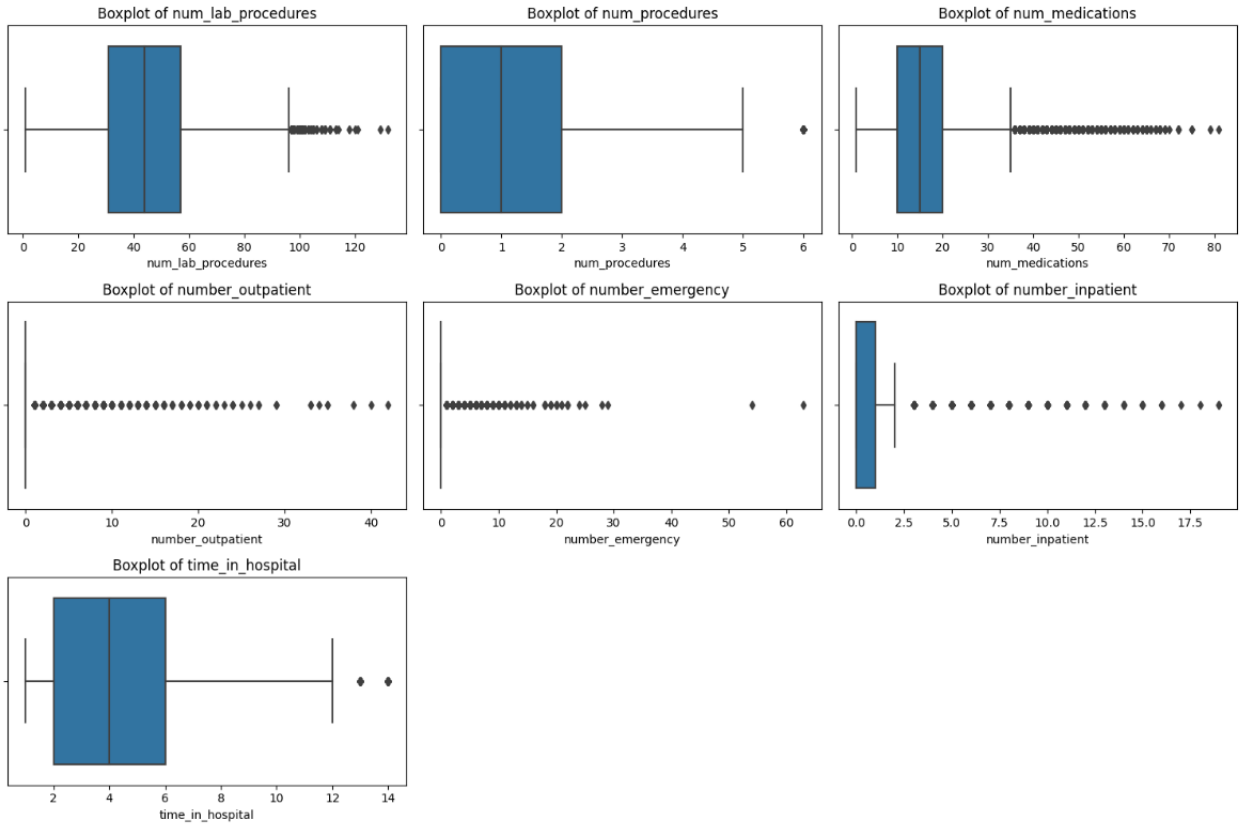
3.2.1 Clustered Bar Graph



We decided to treat the categorical and numeric features separately. For the former, we wanted to see the distribution of readmission_id by clustering the data belonging to a common category. Therefore, a bar graph was drawn for some of the categorical features as shown above. Note that age is also considered as a categorical feature here because it is given in ranges.

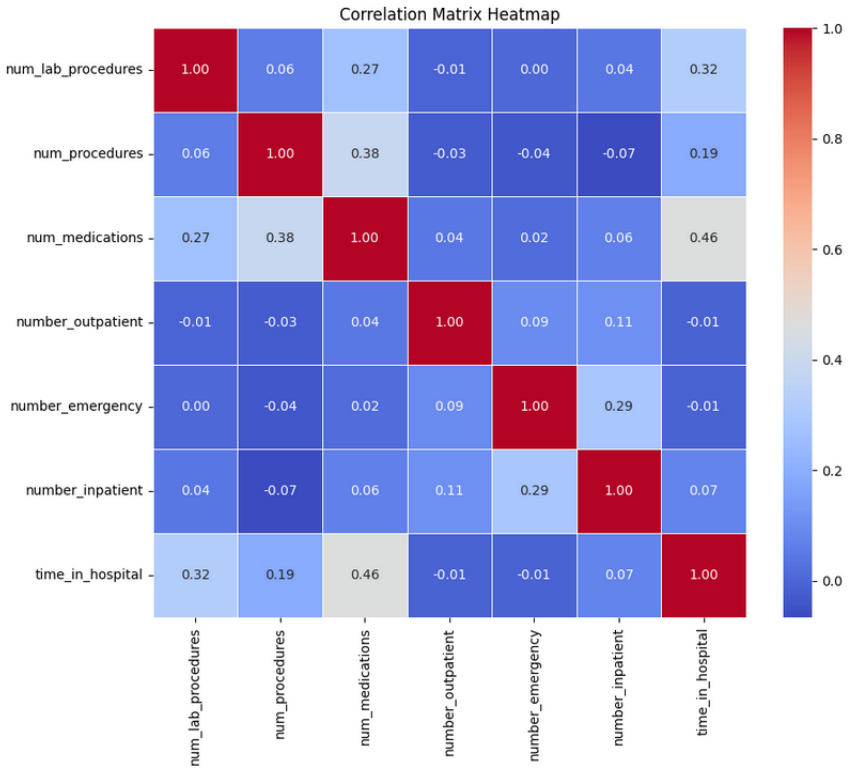
The colour of the bar represents the value of readmission_id and from the above graphs it was evident that irrespective of the feature or category, the readmission_id value 2 was most common followed by 1, and then 0. Unfortunately, there was no single category or feature which stood out, so it was logical to assume that one categorical feature alone did not play a large role in determining the readmission_id value.

3.2.2 Box Plots



For the numeric features, we have generated box plots, with whiskers extending 1.5 times the inter-quartile range. The above graphs give us some information about the data and the outliers. For example, number_outpatient and number_emergency columns have all positive values marked as outliers. Similarly, the number_inpatient column also has a very low upper-quartile and a median of 0.

3.2.3 Heatmap and Correlation Matrix



Again, for the numeric features, we calculated the correlation matrix and plotted the corresponding heatmap as shown above. In order for two variables to be moderately correlated, it should have a correlation coeffecient of above 0.5, but as shown in the figure, the highest correlation value is 0.46. This is between time in hospital and number of medications.

3.3 Post Analysis Actions

We introduced new columns to enrich the dataset and capture additional information:

- **Frequency of Patients**: A new column was created to record the frequency of patient encounters. This column provides insights into how often individual patients were encountered in the dataset, which can be valuable for certain analyses.
- **Frequency of Drug Changes (Up, Down, Steady)**: To better understand the usage patterns of different drugs, we introduced three new columns, one for each type of drug change (up, down, and steady). These columns represent the frequency of changes in drug dosage during patient encounters, allowing us to assess trends in medication adjustments over time.

These additional columns enhance the dataset's richness and provide valuable information for our analysis.

4 Model Building

4.1 Model Training

Models were trained using the following methods:

1. Random Forest with Randomized Grid Search.
2. XGBoost.
3. AdaBoost.
4. K-Nearest Neighbors (KNN).
5. LightGBM Boost.
6. Cat Boost.
7. Support Vector Machine (SVM).
8. Neural Network (NN).

5 SVM Model Performance

5.1 Model Accuracy

- Polynomial Kernel Accuracy: 0.7142 (approximately 71.42%)
- RBF Kernel Accuracy: 0.7127 (approximately 71.27%)

5.2 Time Taken for Model Training

- Polynomial Kernel SVM: Training took approximately 26 minutes.
- RBF Kernel SVM: Training took approximately 28 minutes.
- Linear Kernel SVM: Although omitted from accuracy comparison, the linear kernel took significantly longer to run than both the polynomial and RBF kernels. (More than 4 hours)

5.3 Observations and Insights

The Support Vector Machines (SVMs) were implemented with different kernel functions—polynomial, radial basis function (RBF), and a linear kernel. Preprocessing steps involved imputation and transformation of numerical and categorical features.

- **Model Performance:** Both kernels achieved similar accuracy scores on the validation set, with the polynomial kernel slightly outperforming the RBF kernel by a negligible margin.
- **Kernel Comparison:** The slightly superior accuracy of the polynomial kernel suggests potential feature representation advantages or better separability by a polynomial decision boundary in the dataset.

- **Linear Kernel Performance:** Although accuracy results are not provided, it's notable that the linear kernel took a significantly longer time to run compared to both polynomial and RBF kernels, indicating potential computational challenges with the chosen dataset.
- **Training Time:** Notably, the polynomial kernel took approximately 26 minutes, while the RBF kernel took about 28 minutes for training. The considerable training duration for both models suggests potential computational complexity.

In conclusion, while the polynomial kernel exhibited slightly better accuracy and the RBF kernel was also competitive, the linear kernel's extended runtime suggests possible computational challenges. Further analysis and optimizations might be necessary to improve performance or reduce training times, especially with larger datasets.

6 Non-Neural Ensembles

6.1 Random Forest

- **Description:** Random Forest model trained using Randomized Grid Search for hyperparameter tuning.
- **Accuracy:** 0.7233 (approximately 72.33%)

6.2 XGBoost

- **Description:** XGBoost classifier trained with hyperparameters tuned via RandomizedSearchCV.
- **Accuracy:** 0.73 (approximately 73%)

6.3 AdaBoost

- **Description:** AdaBoost classifier trained with hyperparameters tuned via RandomizedSearchCV.
- **Accuracy:** 0.70 (approximately 70%)

6.4 KNN

- **Description:** K-Nearest Neighbors (KNN) classifier trained with hyperparameters tuned via RandomizedSearchCV.
- **Accuracy:** 0.51 (approximately 51%)

6.5 CatBoost

- **Description:** CatBoost classifier trained with specified hyperparameters.
- **Accuracy:** 0.726 (approximately 72.6%)

6.6 LightGBM

- **Description:** LightGBM classifier trained with specified hyperparameters.
- **Accuracy:** 0.717 (approximately 71.7%)

7 Neural Network Architecture and Components

7.1 Activation Functions: ReLU and Softmax

The neural network uses Rectified Linear Unit (ReLU) activation in its hidden layers, facilitating non-linearity and computational efficiency. ReLU's simplicity and effectiveness in capturing complex patterns make it a popular choice.

Softmax activation is employed in the output layer for multi-class classification. It transforms the network's outputs into a probability distribution across classes, aligning with the multi-class nature of the classification task. This allows for effective calculation of categorical cross-entropy loss during training.

7.2 Dropout Layers for Overfitting Prevention

The inclusion of a Dropout layer in between hidden layers serves to mitigate overfitting by randomly deactivating a fraction of neurons during training. This technique promotes the learning of more robust and generalized representations, enhancing the model's performance on unseen data.

7.3 Neural Network Structure and Design Considerations

The neural network architecture is a fundamental component defining its ability to learn intricate patterns and make accurate predictions. It comprises four key layers, each tailored with specific configurations to optimize learning and generalization:

1. **Input Layer:** This layer implicitly defines the network's input shape, aligning with the number of preprocessed features. In this context, it acts as a conduit for passing the processed data into subsequent layers for feature extraction and learning.

2. **First Hidden Layer:** Positioned after the input layer, this layer is pivotal in the extraction of diverse and intricate feature representations from the input data. By employing Rectified Linear Unit (ReLU) activation, it fosters non-linearity, enabling the network to capture complex relationships within the data. The presence of a substantial number of neurons empowers the network's capacity to discern and learn intricate patterns.

3. **Dropout Layer:** A dropout layer is strategically inserted after the first hidden layer to combat overfitting. This layer plays a pivotal role in regularizing the network by randomly deactivating 20% of its neurons during each training iteration. By doing so, it prevents excessive reliance on specific neurons and encourages the network to develop more robust and generalized representations.

4. **Second Hidden Layer:** Subsequent to the dropout layer, the network features a second hidden layer with a reduced number of neurons compared to the initial hidden layer. This reduction in

neuron count facilitates the abstraction of learned features and hierarchical learning. The hierarchical learning process enables the network to progressively extract more abstract and generalized features as the data traverses through successive layers.

5. **Output Layer:** Positioned as the final layer, the output layer leverages Softmax activation. This activation function is ideal for multi-class classification tasks as it computes the probabilities associated with each class. By producing class probability distributions, it enables the network to assign the most probable class for a given input instance.

7.4 Optimization with Adam

The Adam optimizer is chosen to enhance gradient descent efficiency during training. It dynamically adapts learning rates for each parameter, promoting faster convergence and improved performance, particularly on complex datasets commonly encountered in healthcare domains.

7.5 Model Evaluation and Tuning

The model configuration, including the choice of activation functions, dropout layers, and optimizer, is a balanced approach aiming to prevent overfitting, capture intricate patterns in the data, and generalize well to unseen instances. The specific architecture and hyperparameters are influenced by the dataset's complexity, size, and characteristics.

In summary, the neural network architecture is crafted to strike a balance between complexity and generalization, catering to the nuances of hospital data, which often includes a mix of numerical and categorical features.

8 Neural Network Performance

8.1 Model 1

- **Description:** Sequential neural network architecture with a dropout rate of 0.2, ReLU activation function, and a learning rate of 0.001.
- **Accuracy:** 0.722 (approximately 72.2%)

8.2 Model 2

- **Description:** Sequential neural network architecture with a dropout rate of 0.5, ReLU activation function, and a learning rate of 0.001.
- **Accuracy:** 0.719 (approximately 71.9%)

8.3 Model 3

- **Description:** Sequential neural network architecture with a dropout rate of 0.2, Sigmoid activation function, and a learning rate of 0.001.
- **Accuracy:** 0.720 (approximately 72.0%)

8.4 Model 4

- **Description:** Sequential neural network architecture with a dropout rate of 0.2, ReLU activation function, and a learning rate of 0.01.
- **Accuracy:** 0.712 (approximately 71.2%)

8.5 Model 5

- **Description:** Sequential neural network architecture with 6 layers, a batch size of 64, trained for 20 epochs.
- **Accuracy:** 0.723 (approximately 72.3%)

8.6 Model Training

The neural network models were trained using the provided data with varying configurations. The training was carried out for 10 epochs.

The accuracies reported above represent the performance of each neural network model on the validation data after training.

8.7 Model Evaluation

The performance of each model was assessed, and the accuracy on the validation set is as follows:

- Random Forest: 0.7233
- XGBoost: 0.73
- AdaBoost: 0.70
- LightGBM Boost: 0.717
- Cat Boost: 0.726
- KNN: 0.51
- SVM: 0.71
- NN: 0.72