
Predicting 30-Day Readmission Outcomes for Diabetes Patients: Early, Late, or No Readmission

Loyd Flores

Department of Computer Science
CUNY Queens College
New York, NY 11355

David Mejia

Department of Computer Science
CUNY Queens College
New York, NY 11355

Abstract

Predicting 30-day readmission outcomes for diabetes patients is crucial for optimizing healthcare resource allocation and improving patient care. This study explores the classification of readmission outcomes into three categories: early readmission, late readmission, and no readmission. Leveraging a dataset from the UC Irvine Machine Learning Repository, which spans patient records from 1999 to 2008, we performed data discretization and pattern discovery to identify key features influencing readmissions. We developed predictive models using decision trees, random forests, and Bayesian networks to analyze the relationships between patient attributes and readmission likelihood.

1 Literature Review

1.1 Diabetes Readmission as a model

Machine learning has proven to be a powerful tool in healthcare, particularly for diagnosing and predicting chronic diseases such as diabetes. Dharmarathne et al. (2024) demonstrated the efficacy of machine learning models, including Decision Trees and Random Forrest, in accurately diagnosing diabetes using a publicly available dataset. This integration of machine learning with explainable artificial intelligence provides not only accurate predictions but also critical insights into the factors influencing diabetes outcomes, bridging a gap in trust and usability in medical applications

1.2 How trees are good at classification tasks

Decision trees have shown strong performance in multiclass classification tasks, particularly in medical imaging. For example, Vallée et al. (2023) applied Classification and Regression Tree (CART) models to classify brain tumors, achieving high accuracy (96%) and exceptional AUC values (0.98-1.00) for differentiating between glioblastomas, lymphomas, and metastases. These models effectively captured complex, nonlinear relationships in the data while maintaining interpretability through hierarchical decision rules. This highlights the efficacy of decision trees for structured data and multiclass classification problems

2 Introduction

Diabetes is a chronic and widespread disease that presents significant challenges for healthcare systems. One of the biggest issues is the high rate of hospital readmissions, which strain resources and highlight gaps in patient care. Predicting 30-day readmission outcomes is difficult because many factors influence a patient's condition, such as varying treatment responses and differences in health history.

Modeling these readmissions is further complicated by the complexity of diabetes as a disease and the quality of medical records, which can include missing or inconsistent information. To address these challenges, we applied information theory techniques such as data discretization and pattern discovery to uncover critical features and relationships in the dataset.

Building on these insights, we employed machine learning techniques, including decision trees, random forests, and Bayesian networks, to predict whether patients will experience an early readmission, late readmission, or no readmission. This approach aims to enhance predictive accuracy and provide actionable insights to improve diabetes care and reduce hospital readmissions.

3 Methodology

3.1 Description of our dataset

The dataset encompasses ten years (1999–2008) of clinical data from 130 U.S. hospitals and integrated delivery networks. Each record corresponds to a hospital stay for a patient diagnosed with diabetes, including details on laboratory tests, medications, and stays lasting up to 14 days. The primary objective is to predict whether a patient will experience an early readmission within 30 days of discharge.

This problem is critical because, despite strong evidence supporting preventive and therapeutic interventions for diabetic patients, many do not receive adequate care. This gap is often due to inconsistent diabetes management practices in hospital settings, which neglect proper glycemic control. Poor management not only leads to higher hospital costs due to frequent readmissions but also increases the risk of complications, morbidity, and mortality for patients.

3.2 Exploratory Data Analysis

The dataset includes 50 columns, with 47 features describing various aspects of the patient’s clinical care and outcomes with a significant amount of missingness. Missingness is present in this dataset as important features in determining diabetes readmission such as **weight** is almost completely empty.

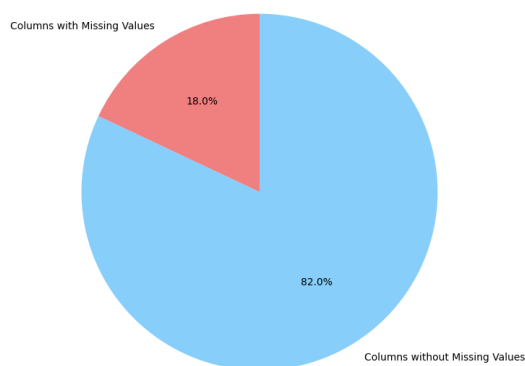


Figure 1: Proportion of Columns with Missingness

3.3 Feature Engineering

The selection of seven features from the original 50 was a deliberate balance between retaining sufficient information for accurate predictions and ensuring computational efficiency. This reduction allows the model to focus on the most relevant variables while avoiding unnecessary complexity. By narrowing the dataset, we ensured efficient use of computational resources without compromising prediction quality.

The selected features represent a comprehensive range of data, including personal attributes (e.g., patient age), medical history (e.g., duration of hospital stay, number of diagnoses recorded), and diabetes-specific details (e.g., insulin treatment type, hemoglobin A1C test result, and readmission status). Many features in the original dataset, such as the number of lab procedures, procedures, medications, outpatient visits, emergency visits, and inpatient visits, were highly similar. We carefully evaluated these overlaps and selected the seven most representative features.

For example, the number of medications was prioritized over similar features like the number of lab procedures because it directly captures the intensity and complexity of a patient’s treatment regimen, which is crucial for predicting diabetes outcomes. This targeted selection of the most descriptive and predictive features ensures the model retains essential information, avoids redundancy, and remains computationally efficient.

By focusing on simplicity and relevance, this streamlined approach allows the model to generalize effectively without unnecessary complexity.

3.3.1 Feature Transformations

Feature transformations are preprocessing steps applied to raw data to enhance its quality and suitability for modeling. These transformations can standardize scales, encode categorical variables, or adjust distributions to better align with the requirements of machine learning algorithms. Even though our selected seven features contain no missing values, preprocessing remains essential to ensure the data is properly normalized, scaled, and ready for effective model training. These steps aim to improve the model’s ability to learn patterns and generalize effectively from the data.

3.3.2 Feature Encoding

To prepare the dataset for modeling, certain features containing text or range-based values required encoding into scalar representations. For example, the **age** feature, originally represented as ranges (e.g., "30-40"), was binned into numerical categories to standardize its format. Similarly, the A1C result and readmission status, which contained categorical strings, were encoded into numeric values. These transformations ensure consistency across the dataset and enable the model to process the features effectively.

Age Range	Encoded Value
[0-10)	0
[10-20)	1
[20-30)	2
[30-40)	3
[40-50)	4
[50-60)	5
[60-70)	6
[70-80)	7
[80-90)	8
[90-100)	9

Table 1: Age Range Encoding Table

A1C Level	Encoded Value
NaN	0
Norm	1
>7	2
>8	3

Table 2: A1C Level Encoding Table

Insulin Level	Encoded Value
No	0
Down	1
Steady	2
Up	3

Table 3: Insulin Level Encoding Table

3.3.3 Discretization of Continuous Values

Discretization is the process of converting continuous variables into discrete categories or bins. This transformation simplifies data representation, reducing unnecessary patterns while preserving the core information. It can improve model interpretability and robustness, particularly for algorithms that benefit from categorical inputs or need a reduction in feature complexity.

Continuous features can introduce noise and complexity into the model, especially if their distributions are uneven or contain outliers. Discretization helps by grouping values into meaningful intervals, which reduces variability and focuses on the most relevant patterns. This step minimizes overfitting risks and enhances the model's ability to generalize, ensuring that critical information is retained without excessive granularity.

3.3.4 Implementation of Discretization

We used a K-Means clustering-based approach for discretizing continuous variables. This method identifies natural groupings within the data and creates bins accordingly.

Below is a breakdown of the steps:

1. **Convert Values to a NumPy Array :** Continuous data is first converted into a NumPy array and reshaped for compatibility with clustering algorithms.
2. **Apply K-Means Clustering :** The data is clustered into a specified number of bins (n_bins), with each cluster representing a discrete category.
3. **Sort Cluster Centers :** The cluster centers (centroids) are sorted to maintain a logical order of bins.
4. **Assign Values to Bins :** Each data point is assigned to the nearest cluster, with labels remapped to reflect the sorted order of bins.
5. **Determine Bin Edges :** Bin edges are calculated based on the distances between sorted cluster centers, providing a clear definition of the intervals.

3.3.5 Finding Ideal Number of Bins

To determine the optimal number of bins for discretizing continuous values, we use the Elbow Method, a common technique in clustering.

The Elbow Method evaluates the trade-off between the number of bins (clusters) and the reduction in variance within each bin, quantified by the **Within-Cluster Sum of Squares (WCSS)**. WCSS measures the total squared distance between each data point and the centroid of its assigned bin. As the number of bins increases, WCSS decreases, but the improvement diminishes at a certain point the "elbow." This point represents the optimal number of bins, balancing simplicity and accuracy.

The formula for **WCSS** is given by:

$$WCSS = \sum_{k=1}^K \sum_{i \in C_k} \|x_i - \mu_k\|^2$$

where:

- K is the number of clusters.

- C_k is the set of data points assigned to the k -th cluster.
- x_i is a data point in the k -th cluster.
- μ_k is the centroid of the k -th cluster.
- $\|x_i - \mu_k\|^2$ represents the squared Euclidean distance between the data point x_i and the cluster centroid μ_k .

3.3.6 Results after Discretization

Variable	First Iteration	Second Iteration
Age	10	10
Time in Hospital	14	9
Number of Medications	75	3
Number of Diagnoses	16	6
Insulin	4	4
A1C Result	4	4
Readmitted	3	3
Total Combinations	10,080,000	77,760

Table 4: Comparison of Possible Combinations Before and After Discretization

The total number of possible combinations is calculated as the product of the unique values for each variable. Each value represents the number of discrete categories or bins into which the variable is grouped.

The formula for calculating the total number of possible combinations is:

$$\text{Total Combinations} = \prod_{i=1}^n U_i$$

where:

- n is the number of variables.
- U_i is the number of unique values for the i -th variable.

For example:

$$\text{First Iteration: } 10 \times 14 \times 75 \times 16 \times 4 \times 4 \times 3 = 10,080,000$$

$$\text{Second Iteration: } 10 \times 9 \times 3 \times 6 \times 4 \times 4 \times 3 = 77,760$$

The discretization process was carefully designed to balance computational efficiency, data integrity, and domain-specific insights, resulting in the final mappings. For the age variable, detailed intervals were retained because age plays a critical role in diabetes-related outcomes. Since diabetes risks and complications are highly age-sensitive, further generalization would risk losing important trends, and the added computational complexity was deemed acceptable. Variables such as time in hospital, number of medications, and number of diagnoses were discretized into broader intervals based on meaningful thresholds. These groupings, such as low, moderate, and high ranges, reduce noise while maintaining sufficient resolution for meaningful analysis. On the other hand, variables like insulin, A1C, and readmitted were kept in their original forms to preserve their diagnostic and predictive significance. Specifically, missing values in A1C were not imputed, as they carry their own clinical implication, patients without an A1C test may not require intervention, potentially indicating a lower risk of readmission. This approach upholds data integrity and avoids introducing biases. The final mappings provide an optimized representation for predictive modeling, ensuring both performance and relevance to the context of diabetes readmission prediction.

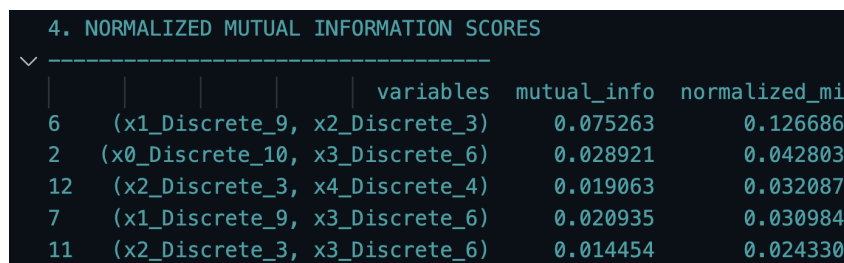
3.4 Pattern Discovery

We implemented 3 different techniques to try and uncover associations in our dataset:

1. **Normalized Mutual Information:** In simple terms, this value measures how much knowing one variable tells you about another variable. So, the higher the MI value between 2 variables (x & y), the higher the mutual dependence for those variables. We find this value and normalize it so that each value is between 0 and 1.
2. **Apriori Algorithm:** We look for frequent itemsets in our dataset using a minimum support threshold of 0.1 (10%). The algorithm iteratively identifies combinations of variables that frequently occur together, starting with individual items and progressively building larger sets. From these frequent itemsets, we generate association rules with a minimum confidence threshold of 0.5 (which we get by computing the frequency of that itemset over the count of all itemsets' frequencies), revealing strong relationships between different medical conditions and patient characteristics. The discovered rules are then ranked by lift to identify the most significant associations.
3. **Chi-squared test:** We employ the chi-squared test of independence to identify statistically significant associations between variable pairs, using an alpha level of 0.05. This test helps determine whether the observed relationships between variables are likely to have occurred by chance. For each significant pair, we calculate a strength measure by normalizing the chi-squared statistic by the sample size and minimum dimension of the contingency table, allowing us to compare the relative strength of different associations. The "alpha" value used in this technique is the significance level, which is basically a threshold to filter out p-values of variables. So, the smaller your alpha is, the more strict you are being to select significant values of specific relationships between variables. Smaller alpha's reduce false positives, but increase false negatives, while larger alpha's do the opposite. Generally, 0.05 tends to be a good threshold for this test.

Pattern Discovery Results:

1. **Normalized Mutual Information:** The two variables that had the highest MI score are *time_in_hospital* and *number_medications*, with a score of 0.127. Below you can see the top five MI scores we achieved, and all were below 0.3, so this metric did not tell us about any strong associations between two variables. This is why we used more than just one technique to assess our data.



```
4. NORMALIZED MUTUAL INFORMATION SCORES
├── variables      mutual_info    normalized_mi
├── 6  (x1_Discrete_9, x2_Discrete_3)  0.075263      0.126686
├── 2  (x0_Discrete_10, x3_Discrete_6)  0.028921      0.042803
├── 12 (x2_Discrete_3, x4_Discrete_4)    0.019063      0.032087
├── 7  (x1_Discrete_9, x3_Discrete_6)    0.020935      0.030984
├── 11 (x2_Discrete_3, x3_Discrete_6)    0.014454      0.024330
```

Figure 2: Normalized Mutual Information Scores

2. **Apriori Algorithm:** Below you will see the first 5 frequent itemsets we found in our dataset. These first five values in figure 3 represent the frequency of a specific value in a column (*xN_Discrete_<value>*).

For example, the frequency of the value 5 in the age variable is 0.169%. We then calculated the frequencies for increasing combinations of different variables.

1. TOP ASSOCIATION RULES		

Total number of frequent itemsets found: 142		
Support threshold: 0.1		
Confidence threshold: 0.5		
Frequent Itemsets Summary:		
support	itemsets	
0 0.169565	(x0_Discrete_10_5)	
1 0.220928	(x0_Discrete_10_6)	
2 0.256156	(x0_Discrete_10_7)	
3 0.168986	(x0_Discrete_10_8)	
4 0.139614	(x1_Discrete_9_1)	

Figure 3: Apriori Frequent Itemsets

Top 10 Association Rules by Lift:									
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift		
86	(x1_Discrete_9_1)	(x2_Discrete_3_0, x5_Discrete_4_0)	0.139614	0.610410	0.112366	0.804828	1.318504		
116	(x3_Discrete_6_1)	(x2_Discrete_3_0, x6_Discrete_3_0)	0.294067	0.400890	0.150748	0.512631	1.278732		
14	(x1_Discrete_9_1)	(x2_Discrete_3_0)	0.139614	0.729075	0.129149	0.925042	1.268790		
85	(x1_Discrete_9_1, x5_Discrete_4_0)	(x2_Discrete_3_0)	0.122202	0.729075	0.112366	0.919508	1.261199		
217	(x5_Discrete_4_0, x3_Discrete_6_1)	(x2_Discrete_3_0, x6_Discrete_3_0)	0.244679	0.400890	0.122683	0.501406	1.250730		

Figure 4: Apriori association rules ranked by Lift score

In figure 4 we see the top five values of conditional variables based on lift. Lift scores are calculated based on two other metrics, Support and Confidence.

Support is calculated by computing the frequency of A and C: $P(AC)$, where A is the Antecedent, and C is the Consequent.

Confidence of A-> C is computed by calculating $P(C|A)/(P(AC)/P(A))$

Lift score is found by: $Confidence(A \rightarrow C)/Support((C))$

So in figure 4, we see for Antecedent = *x1_Discrete_9_1* (time_in_hospital when value = 1) and Consequents: *x2_Discrete_3_0* and *x5_Discrete_4_0* (num_medications when value = 0 and A1Cresult when value = 4). Given this formula which essentially calculates the frequency of A and C occurring, and dividing that by the probability of event C happening given A has occurred, we achieve the highest association score between time_in_hospital, num_medications, and A1Cresult.

Our highest lift score returned is 1.318504. Any values greater than 1 mean that A and C are positively correlated. That means that time_in_hospital is positively correlated to num_medications and A1Cresult, for the specific values stated above. Furthermore, from our mutual information results, we saw that time_in_hospital and num_medications scored the highest MI score. That suggests that they are mutually dependent regardless of specific values. Also, time_in_hospital did not get a high MI score with A1Cresult. These two pieces of information together suggest that time_in_hospital has a stronger correlation with num_medications than it does with A1Cresult.

3. **Chi-squared test:** In the following results, you will see the top 5 most significant associations from our chi-squared test.

3. STATISTICALLY SIGNIFICANT ASSOCIATIONS				
	variables	chi2	p_value	strength
6	(x1_Discrete_9, x2_Discrete_3)	15403.143358	0.000000e+00	0.050453
2	(x0_Discrete_10, x3_Discrete_6)	14270.745793	0.000000e+00	0.023372
12	(x2_Discrete_3, x4_Discrete_4)	4038.583266	0.000000e+00	0.013228
11	(x2_Discrete_3, x3_Discrete_6)	2735.940065	0.000000e+00	0.008962
4	(x0_Discrete_10, x5_Discrete_4)	3468.451904	0.000000e+00	0.008521

Figure 5: Significant Associations via chi-squared test

In Figure 5, we see that all top five scores have a p_value of 0. This score tells us that the associations of these variable pairs are highly unlikely to have occurred by chance. So, all of our top five scores are candidates for statistically significant relationships. This p_value is computed with formula: $P(x^2 \geq \chi^2)$, which represents the area under the chi-square distribution curve. Now, the chi-square distribution curve is computed by the degrees of freedom in our contingency table: $(rows - 1) * (columns - 1)$. We created this table by taking all combinations of 2 variables and using Python `pandas.crosstab()` to count the frequency of each combination of values. The results of that function gives us our contingency table.

Next, we calculated the strength score by computing: $\chi^2/N * \min(r,c)$ where N is the total number of observations in our dataset, and the $\min(r,c)$ represents the minimum dimension of our contingency table (so it simply returns either `num_rows` or `num_cols`).

Lastly, we calculated the chi-square score using `scipy`'s built-in function: `chi2_contingency()`. Although it's abstracted away, this functions uses the contingency table we created using `pd.crosstab()` to calculate the expected frequencies of each piece of observed data: $(row_total * col_total) / grand_total$. Next, using all of these expected frequencies, it calculates $(O - E)^2 / E$ for each cell in O and E . For this to work, O and E must be the same size, and the values in each cell must be respective to each other and related to the same piece of info in our dataset.

Using these three metrics, we see that the highest chi-square score is for variables (`x1_Discrete_9` and `x2_Discrete_3`) at 15403.143358. This suggests that `time_in_hospital` is most closely related to `num_medications`.

Pattern Discovery Conclusion:

1. After running these three tests, in each test the variable combination that achieves the highest score is `time_in_hospital` and `num_medications`. This suggests that those two variables have the strongest correlation in our dataset.

3.5 Dataset conclusion and comparison

To validate the discretization steps and to ensure integrity we compared the predictive power of our dataset to a college's dataset (Dai, W, Khandaker, R, Zhong, B) as they employed similar discretization steps but with a different intuition in feature selection. They had one extra column compared to our results as they used `gender` and `diag1 / initial diagnosis`. This is a fair comparison as the two groups received the same education in the Fall Semester of 2024 under the Dr. Bon Sy

To quickly build a model we used AWS SageMaker Canvas. SageMaker Canvas offers a no-code ML interface for business analysts can create highly accurate machine learning models without any ML experience. This allows us to generate a baseline model to predict on **Readmitted** and **A1Cresult**

The results of both models show very similar results across the board only showing slight differences. When predicting `Readmitted` as the target the similarity was almost identical only differentiating in differences in the tenth decimal which is quite insignificant. These slight difference may imply that our approach is better at avoiding false positives but sacrifices Recall. While their implementation has a better trade-off between Precision and Recall but at the cost of reduced Precision. Since false positives are costly when predicting the readmission of patients, the drop in Precision for Dai, Khandaker, & Zhong could be problematic despite the improved balance.

The larger differences show when predicting **A1Cresult**. Our results show better accuracy and lower validation loss, but struggles on balanced metrics, suggesting it favors majority classes. While our colleagues have balanced accuracy, `f1Macro`, and Recall, which means it performs better on imbalanced datasets, despite higher validation loss. If the focus is on balanced performance across all classes or fairness, Dai, Khandaker, & Zhong performs better. If pure accuracy and generalization are priorities, Flores & Mejia is superior.

Table 5: Validation Metrics Comparison for Readmitted Models

Metric Name	Flores & Mejia	Dai, Khandaker, & Zhong
Accuracy	0.538	0.543
Balanced Accuracy	0.334	0.342
f1Macro	0.238	0.270
Precision Macro	0.476	0.334
Recall Macro	0.334	0.342
Validation Loss	1.008	0.938

Table 6: Validation Metrics Comparison for A1Cresult Models

Metric Name	Flores & Mejia	Dai, Khandaker, & Zhong
Accuracy	0.832	0.611
Balanced Accuracy	0.257	0.367
f1Macro	0.241	0.329
Precision Macro	0.326	0.381
Recall Macro	0.257	0.367
Validation Loss	0.632	0.891

3.6 Modeling readmission with machine learning

The task on hand is to predict how long will a patient be readmitted after diagnoses. The researchers at UC Irvine defined 3 possible outcomes. A patient who was readmitted under 30 days, after 30 days, and was not readmitted at all. This is the perfect use case for machine learning as this is a multi class classification problem. Machine learning is a subset of Statistics that tries to replicate real world phenomena by learning from data and performing various statistical techniques to understand the complex relationships between the features of a phenomena. Multi class because there are 3 possible outcomes or ways we can classify a patient.

3.6.1 Decision Tree

A decision tree is a machine learning model used for classification and regression tasks. It works by recursively splitting the data into subsets based on feature values to maximize the homogeneity of target variables within each subset. At each node in the tree, the model selects the feature and threshold that best separates the data according to a specific criterion, such as Gini impurity, entropy, or mean squared error.

Decision trees are a good choice for modeling readmission because they can handle complex, non-linear relationships between features and outcomes. For example, the likelihood of readmission may depend on a combination of factors like age, number of medications, and prior hospital visits in ways that are not easily captured by linear models. Additionally, decision trees naturally accommodate both numerical and categorical variables, making them well-suited for healthcare datasets with diverse feature types. Their interpretability also makes them valuable in understanding the key drivers of readmission, which is critical in clinical decision-making.

Decision Tree Example: Readmission Prediction

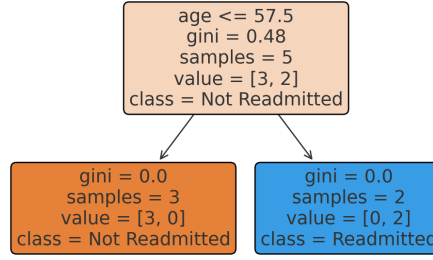


Figure 6: Decision Tree Visualized

3.6.2 Random Forrest

A random forest builds on the concept of decision trees by constructing multiple trees and combining their predictions for improved accuracy and robustness. Each tree in the forest is trained on a random subset of the data (using bootstrap sampling) and a random subset of features, ensuring diversity among the trees. This ensemble approach reduces overfitting by averaging predictions across trees, while random feature selection prevents any single feature from dominating the model. Unlike a single decision tree, which can overfit to noise in the data, random forests are more generalized and robust. They are particularly effective for predicting readmission, as they can model complex, non-linear interactions between features like age, number of medications, and comorbidities. Their ability to handle outliers and missing values, combined with feature importance scores that provide insights into the most influential factors, makes random forests an excellent choice for healthcare applications where accurate and interpretable predictions are crucial.

3.6.3 Bayesian Network

A Bayesian Network (BN) is a probabilistic graphical model that represents a set of variables and their conditional dependencies using a directed acyclic graph (DAG). The nodes in the graph represent random variables, while the edges indicate direct dependencies between variables, with the direction of the edge showing parent-child relationships. The joint probability distribution (JPD) of all variables is expressed as the product of local conditional probabilities:

$$P(X_1, X_2, \dots, X_n) = \prod_{i=1}^n P(X_i \mid \text{Parents}(X_i)),$$

where $\text{Parents}(X_i)$ are the variables that directly influence X_i . This structure encodes conditional independence, allowing efficient computation of marginal and conditional probabilities.

Marginal Probability Distribution The marginal probability of a variable is computed by summing over all possible states of other variables in the network. For example, if we want the probability of X_1 , we marginalize over all other variables:

$$P(X_1) = \sum_{X_2, \dots, X_n} P(X_1, X_2, \dots, X_n).$$

This allows us to focus on the variable of interest while considering the uncertainty in others.

Bayesian Networks utilize the chain rule of probability, breaking down the joint probability distributions into smaller conditional probabilities. They explicitly encode conditional independence relationships, reducing the complexity of inference. For example, if $A \rightarrow B \rightarrow C$, then A and C are conditionally independent given B . This property simplifies computations significantly.

The intuition behind Bayesian Networks is that they represent how the state of one variable influences others, capturing the probabilistic dependencies in a structured way. Instead of computing a potentially intractable joint probability distribution, the DAG decomposes it into manageable pieces using local

conditional probabilities. This is particularly useful in domains like healthcare, where complex, non-linear interactions exist between factors such as age, comorbidities, and readmission likelihood.

Bayesian Networks rely on Bayes' rule to update beliefs when new evidence is introduced:

$$P(A | B) = \frac{P(B | A) \cdot P(A)}{P(B)}.$$

To compute probabilities given evidence, marginalization is applied. For example, to compute $P(A | C)$, we sum over all possible states of intermediate variables:

$$P(A | C) = \frac{\sum_B P(A, B, C)}{P(C)}.$$

Each node is associated with a Conditional Probability Table (CPT), specifying probabilities for all possible states of the node given its parents.

Example Consider a simple network with three variables: A (Smoking), B (Lung Disease), and C (X-ray Result). The DAG structure is:

$$A \rightarrow B \rightarrow C.$$

The Joint probability distribution is expressed as:

$$P(A, B, C) = P(A) \cdot P(B | A) \cdot P(C | B).$$

If we observe that $C = \text{Abnormal}$, we can compute the probability of $B = \text{Lung Disease}$ and update our belief about $A = \text{Smoking}$ using Bayes' rule and marginalization.

Bayesian Networks are particularly effective in healthcare applications, such as predicting hospital readmission. They handle uncertainty, represent causal relationships, and allow inference even with missing data. Their interpretability and scalability make them invaluable tools for clinical decision-making and understanding the interactions between risk factors.

4 Experimentation

4.1 Data Dictionary of Final Data

Variable	Mapping
Age Mapping	
	[0-10): 0 [10-20): 1 [20-30): 2 [30-40): 3 [40-50): 4 [50-60): 5 [60-70): 6 [70-80): 7 [80-90): 8 [90-100): 9
Time in Hospital (Days) Mapping	
	below: 0 (1.0, 2.0): 1 (2.0, 3.0): 2 (3.0, 5.0): 3 (5.0, 7.0): 4 (7.0, 8.0): 5 (8.0, 9.0): 6 (9.0, 10.5): 7 (10.5, 13.0): 8 above: 9
Number of Medications Mapping	
	below: 0 (19.5, 57.4): 1 above: 2
Number of Diagnoses Mapping	
	below: 0 (2.5, 7.0): 1 (7.0, 11.0): 2 (11.0, 13.5): 3 (13.5, 15.5): 4 above: 5
Insulin Mapping	
	No: 0 Down: 1 Steady: 2 Up: 3
A1C Mapping	
	NaN: 0 Norm: 1 >7: 2 >8: 3
Readmitted Mapping	
	NO: 0 >30: 1 <30: 2

4.2 Performance Measurement

4.2.1 Accuracy

Accuracy is a performance metric that measures the proportion of correct predictions made by a model out of the total predictions. It is a straightforward and commonly used metric for evaluating classification models. Accuracy calculates how often the model's predictions match the actual labels. It works by comparing the predicted labels to the true labels and counting the number of correct predictions.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

4.2.2 Precision

Precision measures the proportion of correctly predicted positive instances out of all instances predicted as positive. It indicates the accuracy of positive predictions. Precision evaluates how many of the predicted positives are actually correct. It is useful in scenarios where minimizing false positives is critical.

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

4.2.3 Recall

Recall measures the proportion of correctly predicted positive instances out of all actual positive instances. It is also called sensitivity or true positive rate. Recall evaluates how well the model captures all the actual positives. It is crucial in scenarios where minimizing false negatives is important.

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

4.2.4 F1-Score

The F1-score is the harmonic mean of precision and recall, providing a balanced metric that considers both false positives and false negatives. F1-score is particularly useful when there is an uneven class distribution or when a balance between precision and recall is needed.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

4.3 Experiment 1: Decision Trees

4.3.1 80 - 20 Train Test Split Baseline

This first run of experiment 1 utilizes a default 80 - 20 train test split. This serves as a baseline model to use as a comparator. Running the model just once on a default split is not the most reliable which is why it is only a baseline

Table 8: Classification Metrics and Accuracy for Datasets; Label: Readmitted

Dataset	Precision (Macro Avg)	Recall (Macro Avg)	F1-Score (Macro Avg)	Accuracy
Flores & Mejia	0.38	0.35	0.30	0.5314
Dai, Khandaker, & Zhong	0.36	0.34	0.29	0.5354

Table 9: Classification Metrics and Accuracy for Datasets; Label: A1C

Dataset	Precision (Macro Avg)	Recall (Macro Avg)	F1-Score (Macro Avg)	Accuracy
Flores & Mejia	0.33	0.26	0.25	0.8316
Dai, Khandaker, & Zhong	0.45	0.36	0.32	0.6026

4.3.2 5-Fold Cross-Validation 90-20 Split

In our experimentation, we utilized 5-fold cross-validation to evaluate model performance, as it ensures a comprehensive and robust evaluation by splitting the dataset into 5 equal parts, using 4 folds for training and 1 fold for testing in each iteration, and averaging the results to reduce variance and improve reliability. This method allows the model to train and be tested on all data points, mitigating bias from a single train-test split and enhancing generalization. We opted for a 90-10 train-test split within each fold because our large dataset provided ample rows, enabling the validation set to be smaller while maximizing the training data. This allowed the model to learn better representations from more examples, leveraging the dataset’s richness to improve performance without compromising evaluation reliability. Choosing 5 folds, instead of 10, struck a balance between computational efficiency and robust evaluation, as 10 folds introduce more dependency on smaller validation sets, increasing noise and variability, whereas 5 folds maintain stability with reasonably sized test sets, ensuring a reliable assessment of the model’s generalization capability.

Table 10: Classification Metrics and Accuracy for Datasets; Label: Readmitted

Dataset	Precision (Macro Avg)	Recall (Macro Avg)	F1-Score (Macro Avg)	Accuracy
Flores & Mejia	0.3716	0.3428	0.2943	0.5275
Dai, Khandaker, & Zhong	0.3722	0.3432	0.2888	0.5347

Table 11: Classification Metrics and Accuracy for Datasets; Label: A1C

Dataset	Precision (Macro Avg)	Recall (Macro Avg)	F1-Score (Macro Avg)	Accuracy
Flores & Mejia	0.3301	0.2577	0.2443	0.8292
Dai, Khandaker, & Zhong	0.4055	0.3624	0.3256	0.6039

4.4 Experiment 2: Random Forrest

4.4.1 80 - 20 Train Test Split Baseline

Table 12: Classification Metrics and Accuracy for Datasets; Random Forest (Label: Readmitted)

Dataset	Precision (Macro Avg)	Recall (Macro Avg)	F1-Score (Macro Avg)	Accuracy
Flores & Mejia	0.37	0.35	0.30	0.5295
Dai, Khandaker, & Zhong	0.36	0.34	0.29	0.5360

Table 13: Classification Metrics and Accuracy for Datasets; Random Forest (Label: A1C)

Dataset	Precision (Macro Avg)	Recall (Macro Avg)	F1-Score (Macro Avg)	Accuracy
Flores & Mejia	0.32	0.26	0.25	0.8315
Dai, Khandaker, & Zhong	0.45	0.36	0.32	0.6023

4.4.2 5-Fold Cross-Validation 90-20 Split

Table 14: Classification Metrics and Accuracy for Datasets; Random Forest (Label: Readmitted)

Dataset	Precision (Macro Avg)	Recall (Macro Avg)	F1-Score (Macro Avg)	Accuracy
Flores & Mejia	0.3662	0.3431	0.2987	0.5248
Dai, Khandaker, & Zhong	0.3627	0.3432	0.2924	0.5324

Table 15: Classification Metrics and Accuracy for Datasets; Random Forest (Label: A1C)

Dataset	Precision (Macro Avg)	Recall (Macro Avg)	F1-Score (Macro Avg)	Accuracy
Flores & Mejia	0.3662	0.3431	0.2987	0.5248
Dai, Khandaker, & Zhong	0.3627	0.3432	0.2924	0.5324

4.5 Experiment 3: Bayesian Network

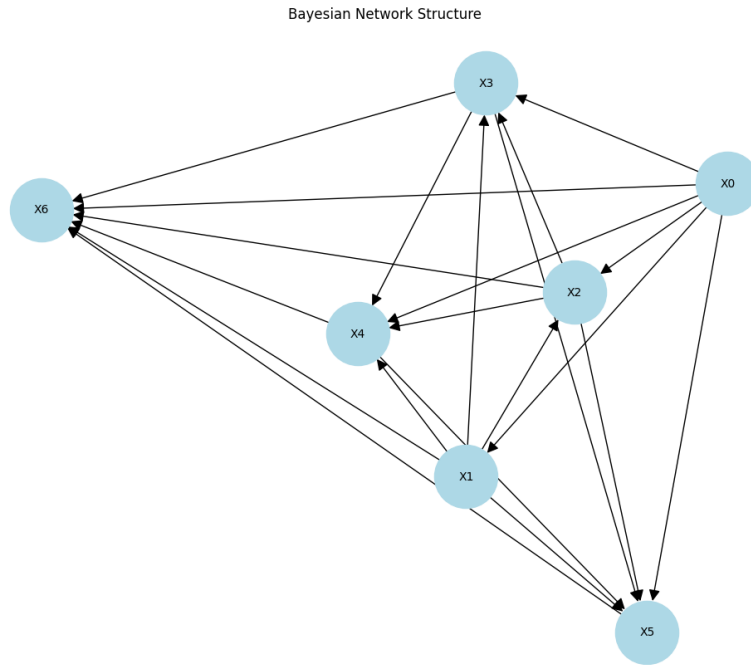


Figure 7: Representation Of Bayesian Network

Table 16: Average Metrics Across Folds for Bayesian Network

Metric	Accuracy	Precision	Recall	F1 Score
Average	0.67	0.76	0.84	0.80

5 Conclusion

In comparing the results from Flores and Mejia versus Dai, Khandaker, and Zhong for decision trees and random forest models, notable differences arise in their predictive performance for A1C results and readmission. While the Dai, Khandaker, and Zhong dataset demonstrated marginally higher accuracy and F1-scores across both predictors, the Flores and Mejia dataset exhibited a stronger performance on A1C prediction using random forest, with an accuracy of 0.8315. This discrepancy highlights the influence of data characteristics on model outcomes. Dai, Khandaker, and Zhong's dataset's moderate improvement in recall and precision for A1C indicates a broader representation of features relevant to glycemic control, making it slightly better suited for this classification task. However, both datasets struggled with readmission prediction, reflecting the inherent complexity of predicting this outcome.

When comparing our results across different model types the decision tree, random forest, and Bayesian networks, we observed that random forests consistently yielded better overall metrics, particularly for A1C prediction, likely due to their ensemble nature, which mitigates overfitting seen in decision trees. Bayesian networks, while theoretically advantageous in capturing probabilistic dependencies, underperformed, likely due to the relatively simplistic feature representation. The adoption of 5-fold cross-validation with a 90-10 split also enhanced the stability and reliability of our models by reducing variance compared to a single train-test split, although the incremental improvement in metrics was modest, suggesting the model's limited sensitivity to these adjustments.

The inherently difficult nature of predicting diabetes readmission must be emphasized. This is a complex and multifaceted issue that subject matter experts devote their careers to understanding, often requiring deep domain knowledge and sophisticated methodologies. Attempting to implement a predictive model in just two weeks as undergraduate students inevitably led to untrustworthy results. Beyond the lack of domain knowledge, our limited experience in data science posed additional challenges. Each stage of the process, from feature selection to data transformation and discretization, was difficult, as we relied heavily on intuition and empirical results without established expertise. Implementing these transformations manually underscored the need for more experience and highlights the challenges of designing robust models with limited resources and time.

Despite experimenting with two datasets and three models, the reliability of the models remains questionable, particularly for readmission prediction. With probabilities hovering around 50%, the predictions are close to random, underscoring challenges in feature selection and model generalization for this outcome. Conversely, the A1C result predictions were markedly more robust, benefiting from well-defined input features directly related to glycemic levels, thus explaining the higher accuracy and F1-scores.

In conclusion, while our models demonstrated moderate success in predicting A1C results, their performance in predicting readmission remains unreliable. This disparity highlights the need for further feature engineering, deeper domain expertise, and more advanced models tailored to the nuances of readmission prediction. The results of this project serve as a testament to the complexity of this problem and the value of subject matter expertise in tackling it effectively.

6 References

[1] Gangani Dharmarathne & Thilini N. Jayasinghe & Madhusa Bogahawaththa & D.P.P. Meddage & Upaka Rathnayake (2024) A novel machine learning approach for diagnosing diabetes with a self-explainable interface, Healthcare Analytics 5 (2024) 100301, ScienceDirect.

[2] Rodolphe Vallée, Jean-Noël Vallée, Carole Guillevin, Athéna Lallouette, Clément Thomas, Guillaume Rittano, Michel Wager, Rémy Guillevin & Alexandre Vallée (2023) Machine learning decision tree models for multiclass classification of common malignant brain tumors using perfusion and spectroscopy MRI data, Frontiers in Oncology, 13:1089998, doi: 10.3389/fonc.2023.1089998.

A Appendix

Loyd Flores: Research/ Paper

David Mejia: Research/ Paper writing

Contribution split:

Flores, Loyd: 50%

1. **Discretization:** Minimized dataset size by manually implementing data cleaning in python by implementing scalers scalers and discretization algorithm using clustering.
2. **Modeling:** Developed DT, RF, Bayesian Network, performed multiple experiments on our own dataset and performed the same experiments on another groups dataset
3. **Research Writing :** spearheaded writing the research paper

Mejia, David 50%:

1. **Sagemaker:** Setup AWS canvas and domain to run 4 separate models. The first predicts the readmitted variable using our discretized dataset. The second also predicts the readmitted variable using Rahib's, Ben's and Weite's discretized dataset. The other 2 models use the same datasets but to predict the A1Cresult column instead.
2. **Pattern Discovery:** Implemented 3 tests to measure relations/associations between different variables of our dataset. That included mutual information, Apriori algorithm, and chi-squared test. This required understanding what these tests performed and how to interpret their results which we have never done before.
3. **Research Writing:** Contributed to the writing of the results and the organizing of the paper.