**Healthcare predictive Analysis project**

Capstone project for the second cohort of the Digital Egypt Initiative, under the auspices of the Ministry of Communications and Information Technology

**Data Science Track**

Project Team

* Mai Mosaad Ragab
* Somia Ashraf Soliman elsaed
* Nada Ragab Ibrahim
* Maha Mosbeh Elkohaley
* Aisha Tarek abdelhamid
* gehad abdelal

***Under Supervision of***

**Dr. Eslam Elreedy**

Acknowledgment

As we reflect on the completion of this project, it is important to recognize the collective efforts that have made this dataset and its analysis possible. The journey of cleaning, analyzing, and preparing the Diabetic Data Cleaning dataset for predictive modeling has been both challenging and rewarding. It is through the dedication and support of various individuals and organizations that this work has reached its current stage.

First and foremost, we extend our deepest gratitude to the original creators of the dataset, whose efforts in compiling and sharing this valuable resource have enabled researchers and data scientists to explore critical healthcare challenges, such as predicting hospital readmissions for diabetic patients. The dataset, sourced from Kaggle (Diabetic Data Cleaning Dataset), is a testament to the importance of open data in advancing medical research and improving patient outcomes.

We would also like to acknowledge the contributions of the healthcare institutions and professionals who originally collected and anonymized the patient data. Their commitment to ethical data practices and patient privacy has ensured that this dataset can be used responsibly for research purposes.

Special thanks go to the data science community, whose collaborative spirit and shared knowledge have been instrumental in refining the techniques and methodologies applied in this project. The guidance and insights from peers, mentors, and online resources have been invaluable in overcoming the technical challenges encountered during the data cleaning and analysis process.

In particular, we are grateful to Dr. Eslam Elreedy, whose expertise and mentorship have been a guiding light throughout this project. His dedication to fostering a deeper understanding of data science principles and his unwavering support have not only contributed to the success of this work but have also inspired growth and learning beyond the scope of this dataset.

Additionally, we appreciate the support of our colleagues and fellow researchers, who have provided constructive feedback and encouragement. Their contributions, whether through direct collaboration or shared resources, have enriched this project and broadened its potential impact.

Finally, we are thankful for the opportunity to work with this dataset and contribute to the ongoing efforts to improve healthcare outcomes through data-driven insights. The knowledge and experience gained from this project will undoubtedly serve as a foundation for future endeavors in the field of medical data analysis.

As we conclude this phase of the project, we do so with a sense of pride and accomplishment, knowing that the work done here has the potential to make a meaningful difference. We look forward to the next steps, including the development of predictive models and further exploration of the data, with the hope of contributing to advancements in patient care and hospital management.

Congratulations to all who have been part of this journey—we did it!

**DECLARATION**

We hereby certify that this material, which we now submit for assessment on the program of the Data Science Track, is entirely our own work. We have exercised reasonable care to ensure that the work is original and, to the best of our knowledge, does not breach any law of copyright. Any work or ideas taken from external sources have been properly cited and acknowledged within the text of our work. This declaration applies to all aspects of the project, including the cleaning, analysis, and documentation of the Diabetic Data Cleaning dataset.

**Signed:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
**Date:** April 29, 2025

**ABSTRACT**

The Diabetic Data Cleaning dataset, accessible via Kaggle and sourced from the UCI Machine Learning Repository, comprises a rigorously curated and preprocessed collection of diabetic patient records, optimized for advanced healthcare analytics and machine learning applications. Preprocessing entailed meticulous imputation of missing values, encoding of categorical variables, and validation of data consistency to ensure analytical integrity. The dataset includes critical patient attributes—demographics, medical histories, and readmission outcomes—rendering it an essential tool for predictive modeling of patient outcomes, exploration of health determinants, and statistical assessment of diabetic care practices. This resource empowers data-driven advancements in patient care and healthcare system efficiency.

The dataset's utility is further underscored by its comprehensive feature set, which includes patient demographics (age, gender, race), medical history (diagnosis codes, number of medications), and hospital admission details (admission type, time in hospital, number of lab procedures). The primary outcome variable, readmission status, is pivotal for developing models that predict patient readmissions, thereby enabling healthcare providers to implement targeted interventions and reduce readmission rates. Additional attributes, such as the number of emergency visits and outpatient encounters, enhance the dataset’s capacity to support longitudinal studies of patient health trajectories. Designed for versatility, it accommodates a range of analytical approaches, from statistical inference to deep learning, fostering innovation in healthcare research. This dataset is a critical asset for researchers, data scientists, and healthcare professionals aiming to leverage data-driven insights for improved patient outcomes and healthcare system efficiency.

Table of ContentS

**Content:**

[**Safe Journey** 2](#_Toc169903164)

**1)**[**INTRODUCTION AND BACKGROUND** 9](#_Toc169903165)

1.1)[Introduction 9](#_Toc169903166)

1.2)[Problem definition 10](#_Toc169903167)

1.3)[Solution 10](#_Toc169903169)

1.4)[Scope 11](#_Toc169903170)

[**2)Analysis and Design** 12](#_Toc169903171)

[2.1) Introduction 12](#_Toc169903172)

[2.2) Requirements 12](#_Toc169903173)

[2.2.1) Functional Requirements 12](#_Toc169903174)

[2.2.2 )Non Functional Requirements 13](#_Toc169903175)

2.3)[Assumptions 14](#_Toc169903176)

2.4)[Constraints 14](#_Toc169903177)

2.5)[System Analysis Diagrams 15](#_Toc169903178)

[Use case diagram. 15](#_Toc169903179)

[Activity diagram. 16](#_Toc169903179)

[Sequence Diagram 17](#_Toc169903180)

[Admin sequence diagram. 18](#_Toc169903179)

[State Diagram 18](#_Toc169903181)

[Data Flow Diagram 19](#_Toc169903181)

[Class Diagram 20](#_Toc169903181)

[Flow Chart 20](#_Toc169903182)

[System Architecture 21](#_Toc169903183)

[3) Models Description 22](#_Toc169903184)

[YOLO: 22](#_Toc169903185)

[Why did we use YOLO: 22](#_Toc169903186)

[Components of YOLO: 22](#_Toc169903187)

[How YOLO Works: 23](#_Toc169903188)

[*3.1) Drowsiness detection Model:* 24](#_Toc169903189)

[Introduction: 24](#_Toc169903190)

[Model Architecture: 24](#_Toc169903191)

[Dataset Collection: 24](#_Toc169903192)

[Methodology: 25](#_Toc169903192)

[Evaluating: 27](#_Toc169903193)

[Results: 27](#_Toc169903194)

[*3.2) Seat belt detection model:* 28](#_Toc169903195)

[Introduction: 28](#_Toc169903196)

[Model Architecture: 28](#_Toc169903197)

[Dataset Collection: 28](#_Toc169903198)

[Methodology: 29](#_Toc169903199)

[Evaluation: 30](#_Toc169903200)

[first try: 30](#_Toc169903200)

[second try: 31](#_Toc169903200)

[Test for model: 32](#_Toc169903201)

[*3.3) Dispersion detection model:* 33](#_Toc169903195)

[Introduction: 33](#_Toc169903196)

[Model Architecture: 33](#_Toc169903197)

[Dataset Collection: 33](#_Toc169903198)

[Methodology: 34](#_Toc169903199)

[Train on Yolo: 35](#_Toc169903199)

[Evaluation: 36](#_Toc169903200)

[4)*Deployment* 37](#_Toc169903202)

4.1)[Flutter Framework & Why We Built This App 37](#_Toc169903203)

[Advantages of Flutter 38](#_Toc169903204)

[Firebase: 38](#_Toc169903205)

[What are the Google Firebase features? 38](#_Toc169903207)

4.2)[Components 39](#_Toc169903208)

4.2.1)[first intro screen 40](#_Toc169903208)

4.2.2)[second intro screen 40](#_Toc169903208)

4.2.3)[third intro screen 41](#_Toc169903208)

4.2.4)[splash screen 41](#_Toc169903208)

4.2.5)[Login 42](#_Toc169903208)

[4.2.6) Register Page 43](#_Toc169903209)

[4.2.7) Home Page: 44](#_Toc169903210)

[4.2.7.1) Nearest places 45](#_Toc169903211)

[4.2.8) History 46](#_Toc169903209)

[4.2.9) Visualization: 46](#_Toc169903210)

[4.2.10) Profile Page 47](#_Toc169903211)

[4.2.11) Settings 48](#_Toc169903213)

[4.3) Methodology 50](#_Toc169903213)

[firebase package 50](#_Toc169903213)

[flutter package 51](#_Toc169903213)

[4.4) Dashboard 54](#_Toc169903213)

[5) Discussion and conclusion 58](#_Toc169903213)

[Introduction 58](#_Toc169903214)

[Main Findings 58](#_Toc169903215)

[Project Importance 58](#_Toc169903216)

[Practical Implementation 59](#_Toc169903217)

[Limitization 60](#_Toc169903213)

[Conclusion Summary 61](#_Toc169903218)

[*Arabic Abstract* 62](#_Toc169903219)

[Refrences 63](#_Toc169903213)

**Table of Figures**

[Figure 1-use case diagram. 15](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910699)

[Figure 2 Activity diagram. 16](#_Toc169910700)

[Figure 3 admin activity 16](#_Toc169910701)

[Figure 4 Sequence diagram 17](#_Toc169910702)

[Figure 5 sequence admin 18](#_Toc169910703)

[Figure 6 State Diagram 18](#_Toc169910704)

[Figure 7 Context diagram 19](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910705)

[Figure 8 Level1-DFD diagram 19](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910706)

[Figure 9 Flow chart 20](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910707)

[Figure 10 System Architecture 21](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910708)

[Figure 11 Yolo Workflow 23](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910709)

[Figure 12 Yolo Architecture for model 1. 24](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910710)

[Figure 13 Dataset 24](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910711)

[Figure 14 Importing all dependencies and libraries. 25](#_Toc169910712)

[Figure 15 Yolo model is a series of cnn. 26](#_Toc169910713)

[Figure 16 Training Yolo on custom data 26](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910714)

[Figure 17 Test Model 27](#_Toc169910715)

[Figure 18 Model 2 architecture 28](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910716)

[Figure 19 Dataset of Model 2 29](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910717)

[Figure 20 Importing the dependencies. 29](#_Toc169910718)

[Figure 21 Model 2 Yolo consist of CNN. 30](#_Toc169910719)

[Figure 22 Training the YOLO model on custom dataset from Roboflow 30](#_Toc169910720)

[Figure 23-first onscreen 40](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910721)

[Figure 24-second onscreen 40](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910722)

[Figure 25- third onscreen 41](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910723)

[Figure 26 splash screen 41](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910724)

[Figure 27 login page 42](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910725)

[Figure 28 Register page 43](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910726)

[Figure 29 Home screen 44](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910727)

[Figure 30 profile screen 47](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910728)

[Figure 31 setting 48](file:///F:\Downloads\SAFE%20JOURNEY121(edit-save1).docx#_Toc169910729)

# **INTRODUCTION AND BACKGROUND**

## **Introduction**

In the complex and evolving landscape of healthcare, where patient outcomes intersect with operational challenges, hospitals face significant issues related to patient readmissions, particularly among diabetic patients. At the forefront of these concerns is the high rate of hospital readmissions, which not only impacts patient health but also places substantial financial and operational burdens on healthcare systems. The Diabetic Data Cleaning dataset, sourced from Kaggle ([Diabetic Data Cleaning Dataset](https://www.kaggle.com/datasets/smit1212/diabetic-data-cleaning)), provides a comprehensive collection of 101,766 patient encounters, capturing demographic, medical, and administrative data to address these challenges.

Recognizing the urgency of improving patient care and reducing readmissions, our project introduces a robust data-driven approach to clean, preprocess, and analyze this dataset. This comprehensive document delves into the intricacies of our methodology, offering a detailed exploration of data cleaning techniques, exploratory data analysis, feature engineering, and the preparation of the dataset for predictive modeling. Our initiative leverages advanced data science techniques to uncover insights and develop solutions that enhance healthcare delivery.

The core objective of this project is to address the root causes of hospital readmissions by harnessing the power of data analytics. Through a thorough understanding of the dataset’s complexities, our work aims to create a foundation for predictive models that can identify at-risk patients, thereby reducing readmission rates and improving patient outcomes. This introduction serves as a prelude to a deeper examination of our commitment to advancing healthcare through data-driven innovation, specifically tailored to the unique challenges posed by diabetic patient care.

## **Problem definition**

Healthcare systems worldwide face critical challenges in managing hospital readmissions, particularly for diabetic patients. Two primary issues dominate this landscape, significantly impacting patient well-being and healthcare efficiency. Firstly, the high prevalence of readmissions, especially within 30 days of discharge, poses a severe threat to patient health, leading to prolonged recovery times and increased medical costs. These readmissions often stem from complex factors, including inadequate post-discharge care, comorbidities, and medication non-adherence. Secondly, the variability and incompleteness of patient data exacerbate the challenge of identifying at-risk individuals. Missing values, inconsistent formats, and high-dimensional data in the Diabetic Data Cleaning dataset hinder accurate analysis and prediction, necessitating robust data cleaning and preprocessing strategies.

The implications of these challenges extend beyond patient care, contributing to financial strain on healthcare systems through penalties for high readmission rates and increased resource utilization. The urgency of addressing these issues underscores the need for an effective, data-driven intervention to improve patient outcomes and optimize hospital operations.

### **Solution**

Our proposed solution is a comprehensive and innovative approach to tackling the challenges of hospital readmissions using the Diabetic Data Cleaning dataset. By leveraging advanced data science techniques, this solution integrates robust data cleaning, exploratory analysis, and feature engineering into a cohesive workflow, preparing the dataset for predictive modeling. Key components of the solution include:

* **Data Cleaning and Preprocessing**:  
  We address missing values (e.g., in weight, max\_glu\_serum, and A1Cresult) by imputing with appropriate placeholders (e.g., "No Test") or the mode for categorical features. Outliers in numerical columns like num\_lab\_procedures are mitigated through imputation or transformation, ensuring data consistency. Categorical variables (e.g., gender, race) are converted to appropriate types, and irrelevant columns like weight are dropped to streamline analysis.
* **Exploratory Data Analysis (EDA)**:  
  Through univariate and bivariate analyses, we uncover patterns in the dataset, such as the positive correlation between num\_lab\_procedures and time\_in\_hospital, indicating that complex cases require more tests and longer stays. The distribution of readmitted highlights class imbalance, guiding model development strategies.
* **Feature Engineering and Transformation**:  
  We create new features, such as time\_diagnoses\_interaction (multiplying time\_in\_hospital and number\_diagnoses), to capture combined effects. Numerical features are normalized using MinMaxScaler, and categorical variables are one-hot encoded to ensure compatibility with machine learning algorithms. The readmitted column is encoded as {NO: 0, >30: 1, <30: 2} to facilitate modeling.

This solution prepares a clean, structured dataset ready for predictive modeling to identify factors contributing to readmissions. By addressing data quality issues and extracting meaningful insights, our approach lays the groundwork for developing models that can reduce readmission rates and improve patient care.

### **Scope**

The scope of the problem is extensive, affecting various aspects of healthcare and requiring a multifaceted approach to address effectively. Key dimensions include:

* **Patient Outcomes**: High readmission rates, particularly within 30 days, compromise patient health and recovery. The scope encompasses identifying risk factors and improving post-discharge care to enhance outcomes.
* **Data Quality**: Incomplete and inconsistent data (e.g., missing values in A1Cresult, high-dimensional diagnosis codes) pose significant challenges. The scope includes developing robust cleaning and preprocessing techniques to ensure data reliability.
* **Healthcare Efficiency**: Readmissions strain hospital resources and incur financial penalties. The scope involves optimizing resource allocation through predictive models that identify at-risk patients.
* **Public Health**: Frequent readmissions contribute to broader public health challenges, including increased morbidity among diabetic patients. The scope encompasses reducing these impacts through data-driven interventions.
* **Economic Impact**: The financial burden of readmissions includes healthcare costs, penalties, and lost productivity. The scope involves mitigating these costs through improved patient management.
* **Technology and Innovation**: Addressing readmissions requires advancements in data science and machine learning. The scope includes the development and application of analytical tools to transform raw data into actionable insights.

By focusing on these dimensions, our project aims to contribute to a safer, more efficient healthcare system through the rigorous analysis of the Diabetic Data Cleaning dataset.

**2)** **Data Description and Preprocessing**

## **2.1 Dataset Overview**

### **2.11.Source:**

### The dataset, sourced from Kaggle ([Diabetic Data Cleaning Dataset](https://www.kaggle.com/datasets/smit1212/diabetic-data-cleaning)), originates from hospital records of diabetic patient encounters, likely inspired by the "Diabetes 130 US Hospitals" dataset (1999–2008). It comprises 101,766 patient encounters with 50 features, capturing demographic, medical, and administrative details to facilitate analysis of hospital readmissions.

### **2.2 Feature Definitions:**

The dataset includes a diverse set of features, each providing critical insights into patient profiles and hospital interactions. Below is a summary of key features:

|  |  |
| --- | --- |
| **Feature Name** | **Description** |
| encounter\_id | Unique ID for each patient visit. |
| patient\_nbr | Unique ID for each patient (multiple visits). |
| race | Patient's race (e.g., Caucasian, African-American). |
| gender | Patient's gender (Male/Female). |
| age | Age group (e.g., [0-10), [10-20), etc.). |
| weight | Patient's weight (in lbs). |
| admission\_type\_id | Type of admission (e.g., emergency, elective). |
| discharge\_disposition\_id | Discharge method (e.g., discharged to home). |
| admission\_source\_id | Admission source (e.g., referral, self-admitted). |
| time\_in\_hospital | Length of stay in days. |
| num\_lab\_procedures | Number of lab tests performed. |
| num\_medications | Number of medications prescribed. |
| number\_outpatient | Number of outpatient visits. |
| number\_emergency | Number of emergency visits. |
| number\_inpatient | Number of inpatient visits. |
| diag\_1, diag\_2, diag\_3 | ICD-9 diagnosis codes. |
| number\_diagnoses | Total number of diagnoses during the visit |
| max\_glu\_serum | Maximum glucose serum measurement (None, Norm, >200, >300). |
| A1Cresult | HbA1c test result (None, Norm, >7, >8). |
| change | Whether medication was changed (Yes/No). |
| diabetesMed | Whether the patient is on diabetes medication (Yes/No). |
| readmitted | Readmission status (<30, >30, NO). |

## 

**2.3 Data Characteristics:**

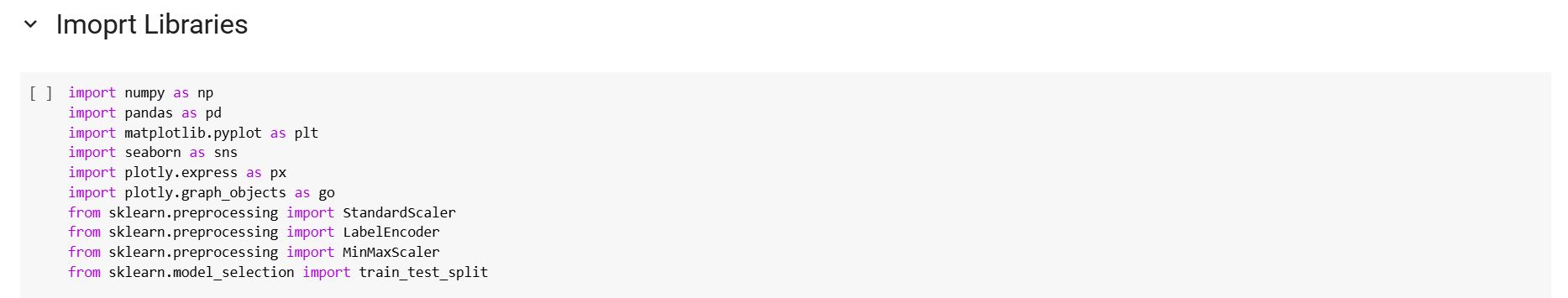
* **Size**: 101,766 rows, 50 columns
* **Data Types**: 13 integer columns (e.g., time\_in\_hospital), 37 object (string) columns (e.g., race)
* **Target Variable**: readmitted, indicating readmission status, which is critical for predictive modeling.

## **3. Methodology**

The preprocessing and analysis workflow was designed to ensure data quality and suitability for modeling. The key steps are outlined below:

### **3.1 Importing Libraries**

To facilitate data manipulation, visualization, and preprocessing, the following Python libraries were utilized:



To enable efficient data processing, visualization, and preparation for machine learning tasks, several Python libraries were employed in this project:

* **NumPy (numpy)**: Used for numerical computations, especially when working with arrays and performing mathematical operations efficiently.
* **Pandas (pandas)**: Essential for data manipulation and analysis, particularly with tabular data structures such as DataFrames.
* **Matplotlib (matplotlib.pyplot)**: Utilized for generating static visualizations such as line plots, bar charts, and histograms.
* **Seaborn (seaborn)**: Built on top of Matplotlib, it provides more advanced statistical visualizations with enhanced aesthetics and simplified syntax.
* **Plotly Express (plotly.express)**: A high-level interface for creating interactive and responsive visualizations with minimal code.
* **Plotly Graph Objects (plotly.graph\_objects)**: A more detailed and customizable way to create interactive plots, allowing full control over plot components.
* **StandardScaler (sklearn.preprocessing.StandardScaler)**: Applied to standardize features by removing the mean and scaling to unit variance, which is crucial for many machine learning algorithms.
* **LabelEncoder (sklearn.preprocessing.LabelEncoder)**: Used to convert categorical string labels into numerical format, making them suitable for model training.
* **MinMaxScaler (sklearn.preprocessing.MinMaxScaler)**: Scales features to a specified range, typically between 0 and 1, which helps normalize data for certain algorithms.
* **Train-Test Split (sklearn.model\_selection.train\_test\_split)**: Used to divide the dataset into training and testing sets, ensuring proper model evaluation and generalization.

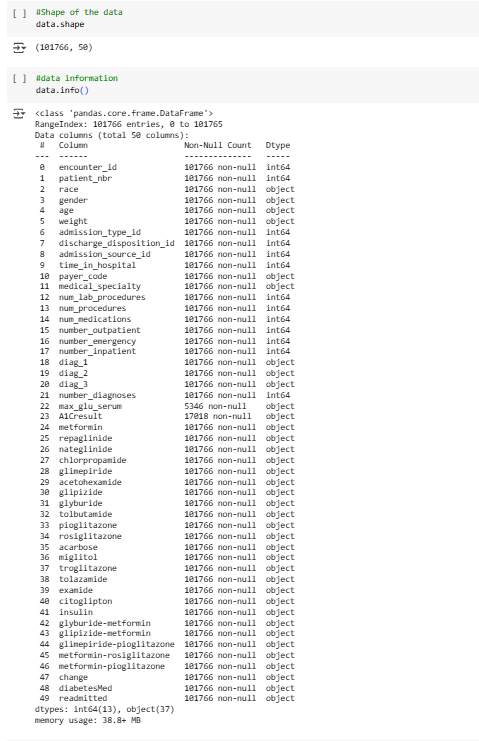
These libraries collectively provide a comprehensive toolkit for handling numerical operations, structured data manipulation, creating both static and interactive visualizations, and preparing features for machine learning models.

### **3.2 Loading the Dataset**

The dataset was loaded from its source and inspected to understand its structure:



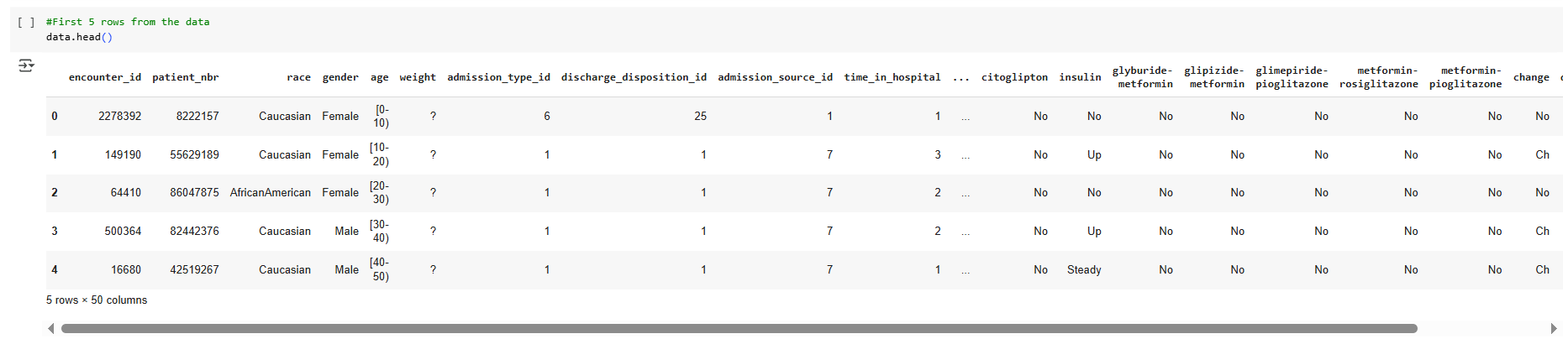
**In this step, the dataset is loaded into a Pandas DataFrame using the read\_csv() function.**  
The file path '/kaggle/input/diabetic-data-cleaning/diabetic\_data.csv' points to the CSV file containing the dataset. This allows for structured data manipulation and analysis using the powerful functionalities of the Pandas library.



**This step provides an overview of the dataset's structure and basic metadata.**  
The output of data.shape reveals that the dataset consists of **101,766 rows** and **50 columns**, indicating a large and feature-rich dataset suitable for analysis and modeling.

The data.info() function displays detailed information about each column, including the number of non-null entries and their data types. The dataset includes both numerical and categorical variables, with 13 columns of type int64 and 37 of type object. This highlights the need for appropriate preprocessing steps, such as handling missing values, encoding categorical features, and scaling numerical values before training machine learning models.

Additionally, some columns, such as max\_glu\_serum and A1Cresult, contain missing or sparse data, which may require special treatment during data cleaning.



This step in understanding the dataset is to examine its structure, which includes the number of rows and columns, as well as identifying the feature names and the associated data types for each column.

An initial inspection of the data reveals some issues, such as missing values in columns like "weight" and potential inconsistencies in categorical encoding. For example, categorical data appears both as text (e.g., "Male", "Female") and as intervals (e.g., "[0-10]" for age). These inconsistencies and missing values need to be addressed to ensure data quality and integrity during the preprocessing phase.

To get a better sense of the data, we use the data.head() function, which displays the first five rows of the dataset. This preview provides a snapshot of the structure, allowing us to inspect sample records, column names, and the general formatting of the data. It helps us identify irregularities such as missing values (represented by "?") and inconsistencies in categorical representations.

Initial inspection revealed missing values (e.g., in weight, max\_glu\_serum) and a mix of numerical and categorical features, necessitating comprehensive preprocessing.

## **4. Exploratory Data Analysis (EDA):**

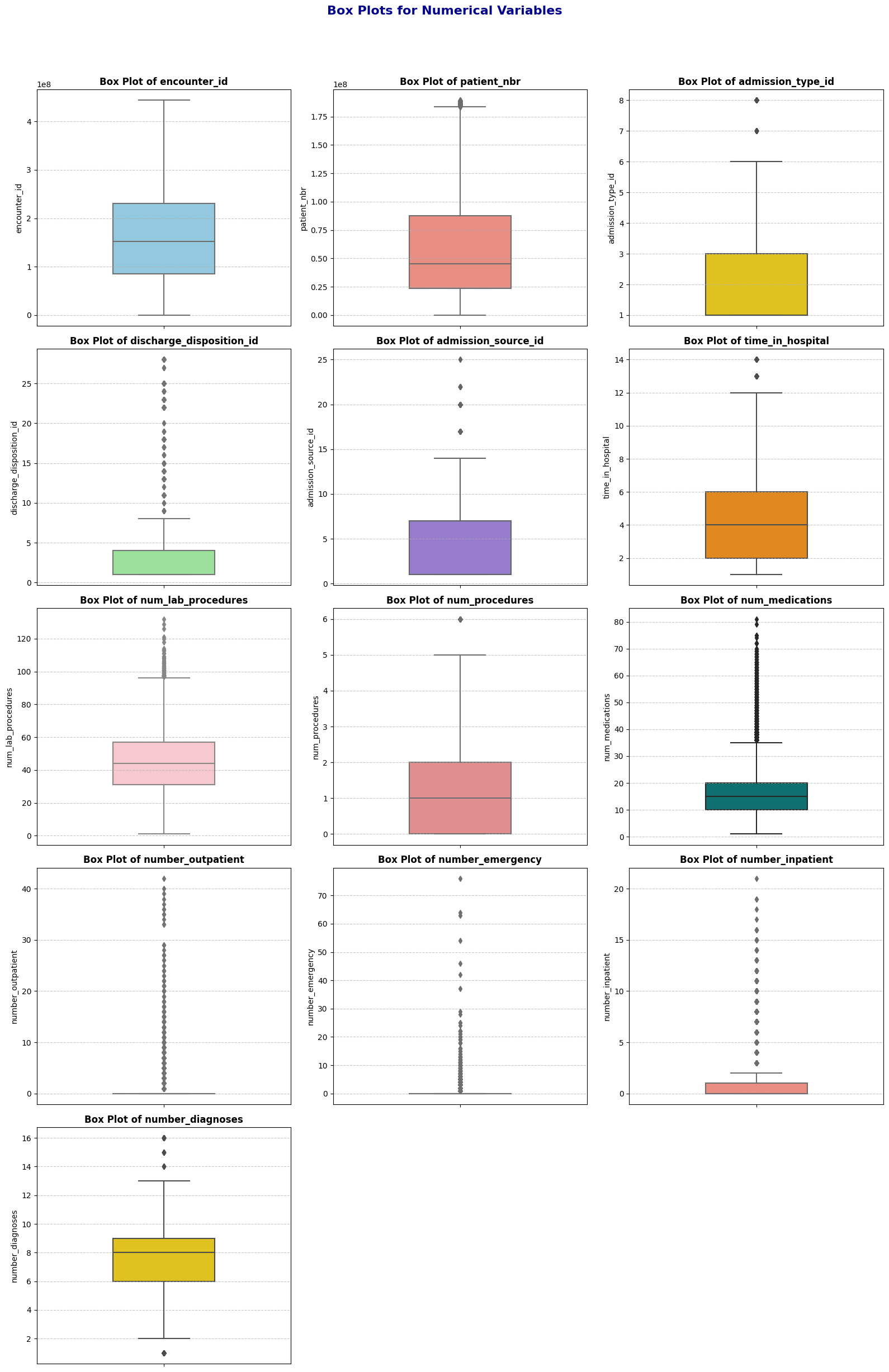
Exploratory Data Analysis was conducted to uncover patterns, distributions, and relationships within the dataset, guiding subsequent preprocessing and modeling efforts.

### **4.1 Descriptive Statistics**

* **Numerical Features**: Summarized using mean, median, and standard deviation to understand central tendencies and variability. For example, num\_lab\_procedures and time\_in\_hospital were analyzed to assess typical patient testing and stay durations.
* **Missing Values**: Quantified per column to identify data quality issues, with weight showing a high percentage of missing entries (marked as ?) and max\_glu\_serum and A1Cresult having significant "None" values.

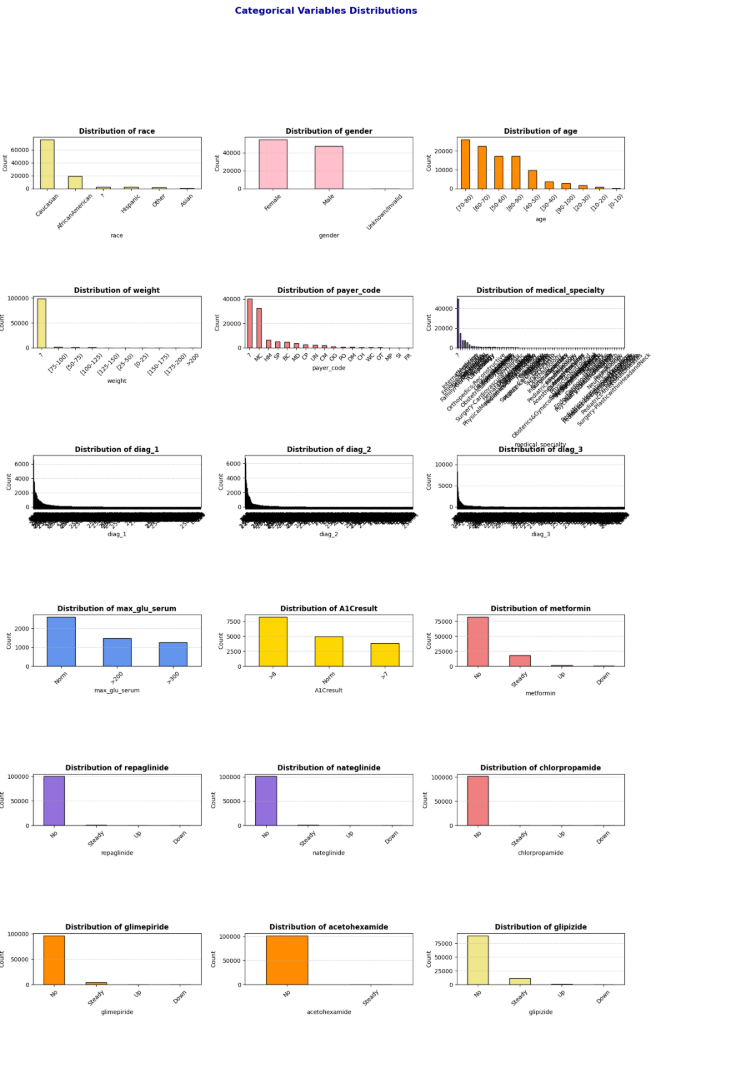
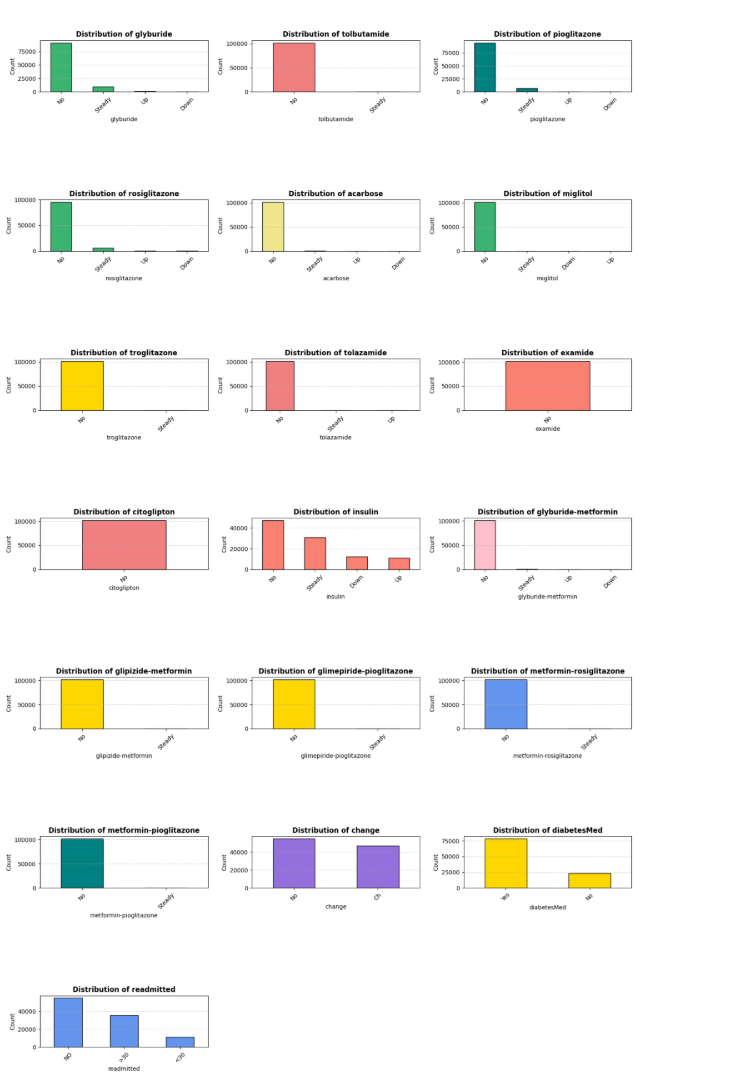
### **4.2 Univariate Analysis**

* **Box Plots**: Generated for numerical variables (e.g., num\_lab\_procedures, num\_medications) to inspect distributions and detect outliers. These plots revealed that num\_lab\_procedures had a skewed distribution with some extreme values, indicating potential outliers.



### **4.3 Multivariate Analysis**

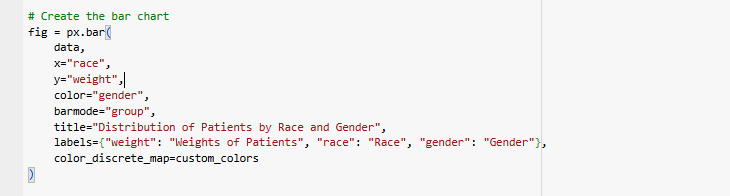
* **Count Plots and Bar Charts**: Created for categorical features (e.g., race, gender, readmitted) to examine their distributions relative to readmission status. For instance, count plots showed that the "NO" category in readmitted was predominant, indicating class imbalance.

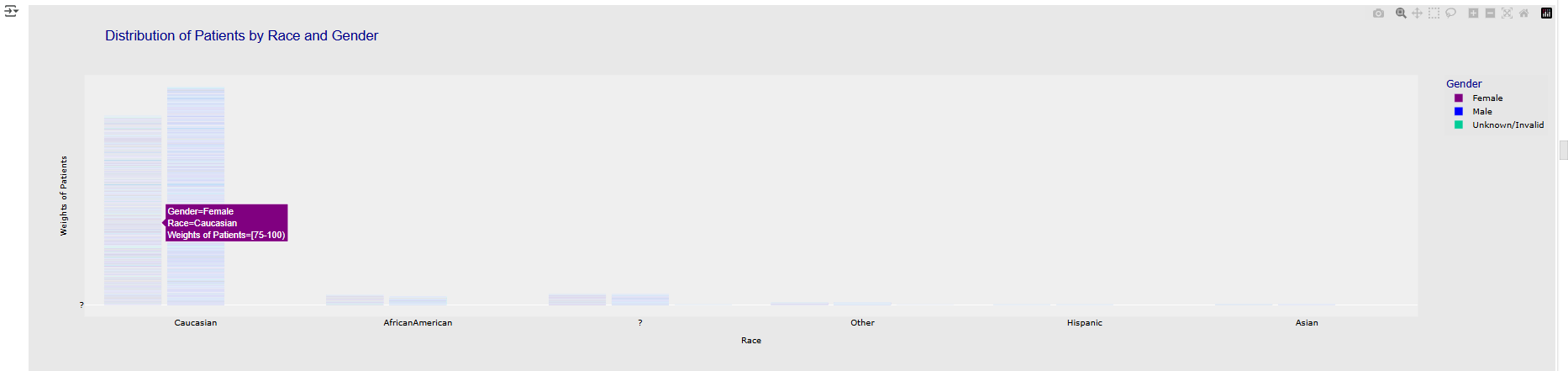
### **4.4 Interactive Visualizations**

Interactive visualizations were developed using Plotly to provide dynamic insights:

* **Bar Chart**: Illustrated the distribution of patients by race and weight, grouped by gender. Due to missing weight data, this visualization was limited but highlighted demographic patterns:



An interactive bar chart of race and weight by gender:



* **Bubble Chart**: Visualized readmitted against num\_lab\_procedures, with bubble size representing time\_in\_hospital. This chart revealed that patients with more lab procedures and longer hospital stays were more likely to be readmitted.



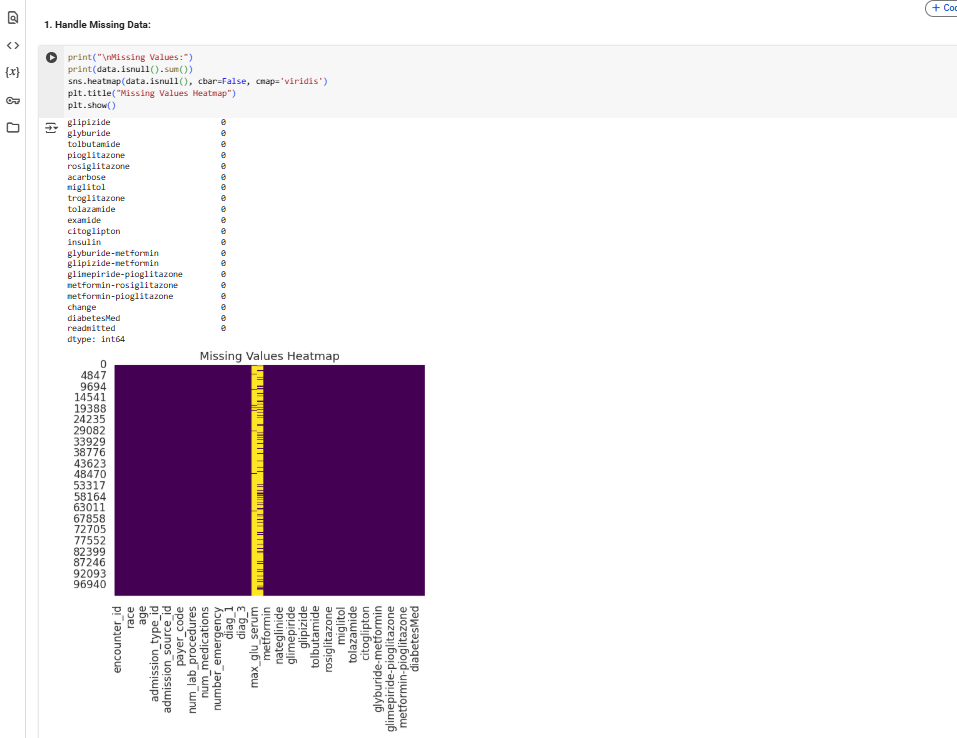
These analyses highlighted key patterns, such as the correlation between num\_lab\_procedures and time\_in\_hospital, and the imbalanced nature of readmitted, informing preprocessing strategies.

## **5. Data Preprocessing and Cleaning**

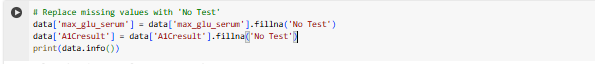
To ensure the dataset’s quality and usability for analysis and predictive modeling, a rigorous cleaning and preprocessing pipeline was implemented. The following steps addressed missing values, invalid entries, data types, outliers, and feature transformations:

### **5.1 Handling Missing Data**

* **Visualization**: A heatmap was used to visualize missingness across the dataset, confirming high missing values in weight and partial missingness in max\_glu\_serum and A1Cresult:



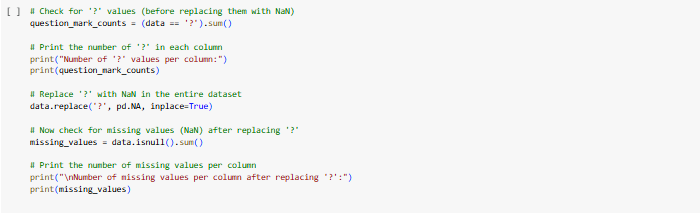
* **Max Glu Serum & A1Cresult**: Missing values were imputed with "No Test" to indicate no testing occurred:



* **Race, Payer Code, Medical Specialty, Diagnoses (diag\_1, diag\_2, diag\_3)**: Missing entries were filled with the mode (most frequent value) of each column to maintain data consistency.
* **Weight**: Dropped due to a high proportion of missing values (97% marked as ?) and limited analytical value, reducing noise in the dataset.

### **5.2 Handling Invalid Entries**

* **Invalid Entries (**?**)**: Counted ? entries in categorical columns (e.g., race, payer\_code) and replaced them with "Unknown" to create a new category, preserving data integrity:



### **5.3 Encoding Categorical Variables**

* **Readmitted**: Label encoded as {'NO': 0, '>30': 1, '<30': 2} and renamed to encoded\_readmitted for clarity:



* **DiabetesMed**: Binary encoded as {'No': 0, 'Yes': 1} and renamed to binary\_diabetesMed:



* **Other Categorical Features**: Columns like gender and race were converted to categorical types to optimize memory and facilitate analysis.

### **5.4 Data Type Conversion**

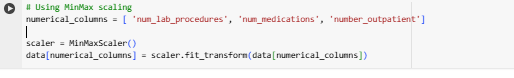
* **Age**: Converted from categorical ranges (e.g., "[0-10)") to numerical midpoints (e.g., 5) for consistency in analysis and modeling.
* **Num Lab Procedures**: Ensured as integers, with non-numeric values replaced by the column’s median to maintain numerical integrity.

### **5.5 Handling Outliers**

* Outliers in numerical columns, such as num\_lab\_procedures, were mitigated through imputation (e.g., replacing extreme values with the median) or transformation (e.g., log-scaling) to minimize their impact on analysis and modeling.

### **5.6 Feature Scaling**

* Numerical features (e.g., num\_lab\_procedures, num\_medications, number\_outpatient) were normalized using MinMaxScaler to ensure values lie within [0, 1], improving model performance for algorithms sensitive to feature scales.



### **4.7 Saving Cleaned Data**

* The processed dataset was saved for further analysis and modeling:



## **6. Results and Discussion**

### **6.1 Dataset Summary**

* **Size**: The cleaned dataset contains 101,766 records and 49 features (after dropping weight).
* **Readmission Class Distribution**:
  + **0 (NO)**: ~54,864 records (53.9%)
  + **1 (>30)**: ~35,541 records (34.9%)
  + **2 (<30)**: ~11,361 records (11.2%)

### **6.2 Key Observations**

* **Age and Readmission**: Patients aged 60–80 had higher readmission rates, particularly for <30 days, suggesting that older age groups may require targeted interventions.
* **Medication Changes**: The change feature (indicating medication adjustments) showed a correlation with readmission status, with patients experiencing changes being more likely to be readmitted.
* **Lab Procedures and Hospital Stay**: A positive correlation between num\_lab\_procedures and time\_in\_hospital indicated that complex cases involve more tests and longer stays.

### **6.3 Data Quality**

* The preprocessing steps eliminated missing values, corrected invalid entries, and ensured consistent data types, resulting in a high-quality dataset ready for predictive modeling.
* The class imbalance in encoded\_readmitted suggests the need for techniques like oversampling or weighted loss functions in model development.

## **7. Conclusion**

This documentation outlines the comprehensive process of analyzing and cleaning the Diabetic Data Cleaning dataset. The EDA uncovered critical patterns, such as the correlation between lab procedures and hospital stay duration, while the preprocessing pipeline addressed missing values, invalid entries, outliers, and feature transformations. The resulting dataset, saved as data\_cleaned.csv, is well-structured and model-ready, providing a solid foundation for building predictive models to forecast hospital readmissions. The insights gained from this analysis highlight key factors influencing readmissions, paving the way for improved patient care and healthcare efficiency.

**3)** **Data Preprocessing**

**2.1 Dataset Overview**

**2.1.1 Source:**

The dataset, sourced from Kaggle, is derived from the "Diabetes 130-US hospitals for years 1999–2008" dataset, which encompasses a decade of clinical care data from 130 U.S. hospitals and integrated delivery networks. It comprises 101,766 patient encounters with 50 features, capturing demographic, medical, and administrative details to facilitate analysis of hospital readmissions. ​[Kaggle+6datasets.aim-ahead.net+6Kaggle+6](https://datasets.aim-ahead.net/dataset/p/UCI_DS_296?utm_source=chatgpt.com)

**2.1.2 Description:**

Each record in the dataset represents a hospital admission for a patient diagnosed with diabetes, with stays ranging from 1 to 14 days. The dataset includes information on demographics (such as age, gender, and race), diagnoses, laboratory test results, medications administered, number of inpatient, outpatient, and emergency visits in the year preceding the encounter, and details about hospital admission and discharge. This comprehensive data supports analyses aimed at understanding factors influencing hospital readmissions among diabetic patients. ​[GitHub+2Fairlearn+2datasets.aim-ahead.net+2](https://fairlearn.org/main/user_guide/datasets/diabetes_hospital_data.html?utm_source=chatgpt.com)[Fairlearn+2datasets.aim-ahead.net+2GitHub+2](https://datasets.aim-ahead.net/dataset/p/UCI_DS_296?utm_source=chatgpt.com)

**2.1.3 Objective:**

The primary objective of utilizing this dataset is to analyze and predict hospital readmissions among diabetic patients. By examining various features related to patient demographics, medical history, and hospital stay details, the goal is to identify patterns and factors that contribute to readmissions, thereby informing strategies to improve patient care and reduce healthcare costs associated with frequent hospitalizations.​[Kaggle+3AI Framework+3Kaggle+3](https://www.restack.io/p/ai-for-personalized-medicine-answer-kaggle-diabetes-dataset-cat-ai?utm_source=chatgpt.com)

**2.2 Feature Definitions**

The dataset includes a diverse set of features, each providing critical insights into patient profiles, medical history, hospital interactions, and outcomes. Below is a summary of key features from the dataset:

|  |  |
| --- | --- |
| Feature Name | Description |
| encounter\_id | Unique identifier for each hospital encounter. |
| patient\_nbr | Unique identifier for each patient; may appear in multiple encounters. |
| race | Patient’s race (e.g., Caucasian, African-American, Asian, etc.). |
| gender | Patient’s gender (e.g., Male, Female, Unknown/Invalid). |
| age | Patient’s age group in intervals (e.g., 0-10, 10-20, ..., 90-100). |
| admission\_type\_id | Numeric code for type of admission (e.g., emergency, urgent, elective). |
| discharge\_disposition\_id | Code indicating where the patient was discharged to (e.g., home, rehab). |
| admission\_source\_id | Code for source of admission (e.g., referral, ER, transfer). |
| payer\_code | Code representing the source of payment (e.g., private, Medicare, Medicaid). |
| medical\_specialty | Medical specialty of the admitting physician (e.g., cardiology, endocrinology). |
| num\_lab\_procedures | Number of lab tests performed during the encounter. |
| num\_procedures | Number of other procedures performed (not lab related). |
| num\_medications | Number of unique medications prescribed. |
| number\_outpatient | Number of outpatient visits in the year prior to the encounter. |
| number\_emergency | Number of emergency visits in the year prior. |
| number\_inpatient | Number of inpatient visits in the year prior. |
| diag\_1, diag\_2, diag\_3 | ICD-9 codes for primary, secondary, and tertiary diagnoses. |
| max\_glu\_serum | Maximum glucose serum level (e.g., None, >200, >300, Norm). |
| A1Cresult | Result of HbA1c test (e.g., None, >7, >8, Norm). |
| change | Indicates if diabetes medication was changed during the encounter (Yes/No). |
| diabetesMed | Whether the patient was on diabetes medication during the encounter (Yes/No). |
| readmitted | Indicates whether patient was readmitted (e.g., <30 days, >30 days, No). |
| encoded\_readmitted | Numerical encoding of the readmitted field for modeling (e.g., 1, 0). |
| time\_diagnoses\_interaction | Interaction term combining time and number of diagnoses. |

**2.3 Data Characteristics**

* **Size**: The dataset consists of **101,766 rows** and **48 columns**, each representing various aspects of hospital encounters.
* **Data Types**:
  + **11 integer columns** (e.g., encounter\_id, admission\_type\_id, number\_inpatient)
  + **4 float columns** (e.g., age, num\_medications, num\_lab\_procedures)
  + **33 object (string) columns** (e.g., race, gender, diag\_1)
* **Target Variable**:
  + The main outcome of interest is **readmitted**, which indicates whether the patient was readmitted after discharge. It is also encoded as **encoded\_readmitted** for predictive modeling purposes.

**3.1 Importing Libraries**

To facilitate data manipulation, visualization, and preprocessing, the following Python libraries were utilized:

# Import necessary libraries

import numpy as np # For numerical operations

import pandas as pd # For data manipulation

import matplotlib.pyplot as plt # For static visualizations

import seaborn as sns # For advanced statistical visualizations

import plotly.express as px # For interactive plots (high-level)

import plotly.graph\_objects as go # For customizable interactive plots

from sklearn.preprocessing import StandardScaler # For feature standardization

from sklearn.preprocessing import LabelEncoder # For encoding categorical labels

from sklearn.preprocessing import MinMaxScaler # For feature scaling to range [0, 1]

from sklearn.model\_selection import train\_test\_split # For splitting data into train/test sets

* **NumPy (numpy)**: Used for performing efficient numerical computations, particularly for working with arrays and conducting mathematical operations.
* **Pandas (pandas)**: Essential for handling and analyzing structured data, specifically with tabular structures like DataFrames.
* **Matplotlib (matplotlib.pyplot)**: Utilized for creating static visualizations such as line plots, bar charts, and histograms.
* **Seaborn (seaborn)**: Built on top of Matplotlib, offering enhanced statistical visualizations with improved aesthetics and simplified syntax.
* **Plotly Express (plotly.express)**: Provides an easy-to-use interface for generating interactive and responsive visualizations with minimal code.
* **Plotly Graph Objects (plotly.graph\_objects)**: Offers more control and customization for creating interactive plots.
* **StandardScaler (sklearn.preprocessing.StandardScaler)**: Used to standardize features by removing the mean and scaling them to unit variance, which is important for many machine learning algorithms.
* **LabelEncoder (sklearn.preprocessing.LabelEncoder)**: Converts categorical string labels into numerical values, making them suitable for training machine learning models.
* **MinMaxScaler (sklearn.preprocessing.MinMaxScaler)**: Scales features to a specified range, often between 0 and 1, to normalize data for certain algorithms.
* **Train-Test Split (sklearn.model\_selection.train\_test\_split)**: Splits the dataset into training and testing sets, enabling model evaluation and ensuring good generalization.

These libraries collectively provide a comprehensive toolkit for numerical operations, data manipulation, visualization (both static and interactive), and preparing features for machine learning models

**3.2 Loading the Dataset**

The dataset was loaded from its source and examined to understand its structure:

In this step, the dataset is loaded into a Pandas DataFrame using the read\_csv() function. The file path '/kaggle/input/diabetic-data-cleaning/diabetic\_data.csv' points to the CSV file containing the dataset. This allows for structured data manipulation and analysis through the powerful functionalities of the Pandas library.

This step provides an overview of the dataset's structure and basic metadata. The output of data.shape reveals that the dataset consists of 101,766 rows and 50 columns, indicating a large and feature-rich dataset suitable for analysis and modeling. The data.info() function displays detailed information about each column, including the number of non-null entries and their data types. The dataset includes both numerical and categorical variables, with 13 columns of type int64 and 37 of type object. This highlights the need for appropriate preprocessing steps, such as handling missing values, encoding categorical features, and scaling numerical values before training machine learning models.

Additionally, some columns, such as max\_glu\_serum and A1Cresult, contain missing or sparse data, which may require special treatment during data cleaning.

This step in understanding the dataset is to examine its structure, which includes the number of rows and columns, as well as identifying the feature names and the associated data types for each column. An initial inspection of the data reveals some issues, such as missing values in columns like "weight" and potential inconsistencies in categorical encoding. For example, categorical data appears both as text (e.g., "Male", "Female") and as intervals (e.g., "[0-10]" for age). These inconsistencies and missing values need to be addressed to ensure data quality and integrity during the preprocessing phase.

To get a better sense of the data, we use the data.head() function, which displays the first five rows of the dataset. This preview provides a snapshot of the structure, allowing us to inspect sample records, column names, and the general formatting of the data. It helps us identify irregularities such as missing values (represented by "?") and inconsistencies in categorical representations.

Initial inspection revealed missing values (e.g., in weight, max\_glu\_serum) and a mix of numerical and categorical features, necessitating comprehensive preprocessing.

**Code Implementation**

data =pd.read\_csv('/kaggle/input/dataset-clean/data\_Cleaing (1).csv')

This mirrors your previous section while adapting it to the dataset loading process. Let me know if you need any further adjustments!

**Exploratory Data Analysis (EDA):**

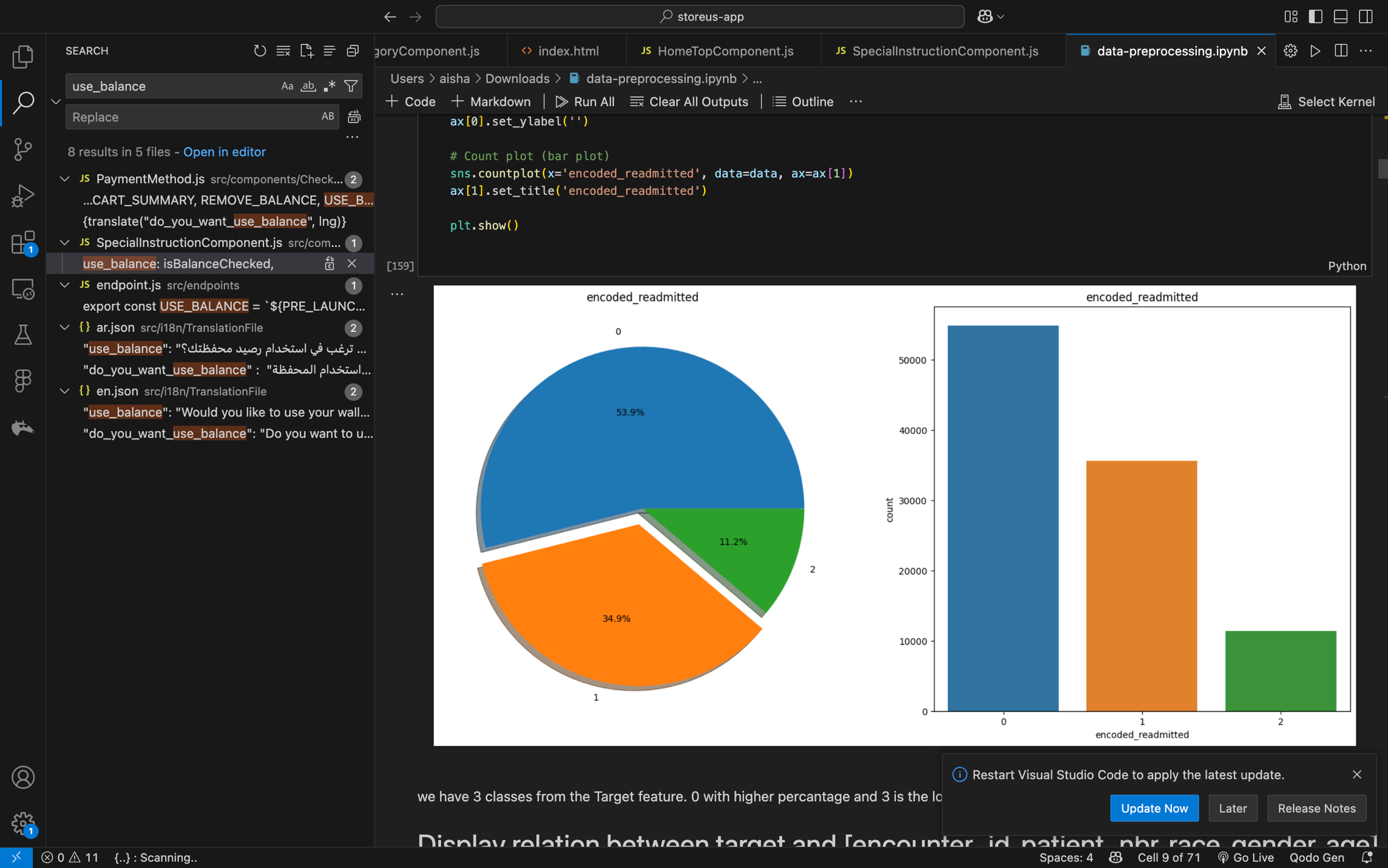
Exploratory Data Analysis (EDA) was conducted to uncover patterns, distributions, and relationships within the dataset, guiding subsequent preprocessing and modeling efforts.

**4.1 Descriptive Statistics:**

Numerical Features: Summarized using mean, median, and standard deviation to understand central tendencies and variability.  
Example: Analysis of num\_lab\_procedures (number of lab procedures) and time\_in\_hospital (length of hospital stay) to assess testing frequency and hospitalization duration.

Missing Values: Quantified missing values per column, with weight showing 97% missing entries (marked as ?).

**4.2 Univariate Analysis:**

Box Plots:  
  
The distribution of num\_lab\_procedures is skewed with outliers, indicating potential unusual cases.

**4.3 Multivariate Analysis:**

Count Plots for Categorical Features:  
A screenshot of a computer

AI-generated content may be incorrect.  
The "NO" category in readmitted dominates, highlighting class imbalance.

**4.4 Interactive Visualizations:**

Interactive Bar Chart (Race/Weight by Gender):  
  
Distribution of patients by race and weight (limited by missing weight data).

**5. Data Preprocessing and Cleaning:**

**5.1 Handling Missing Data:**

Imputation:

max\_glu\_serum and A1Cresult: Missing values replaced with "No Test".

race, payer\_code: Filled with mode (most frequent value).

weight: Dropped due to 97% missing values.

**5.2 Handling Invalid Entries:**

Invalid Entries (?):  
Replaced with "Unknown" to preserve data integrity.

**5.3 Encoding Categorical Variables:**

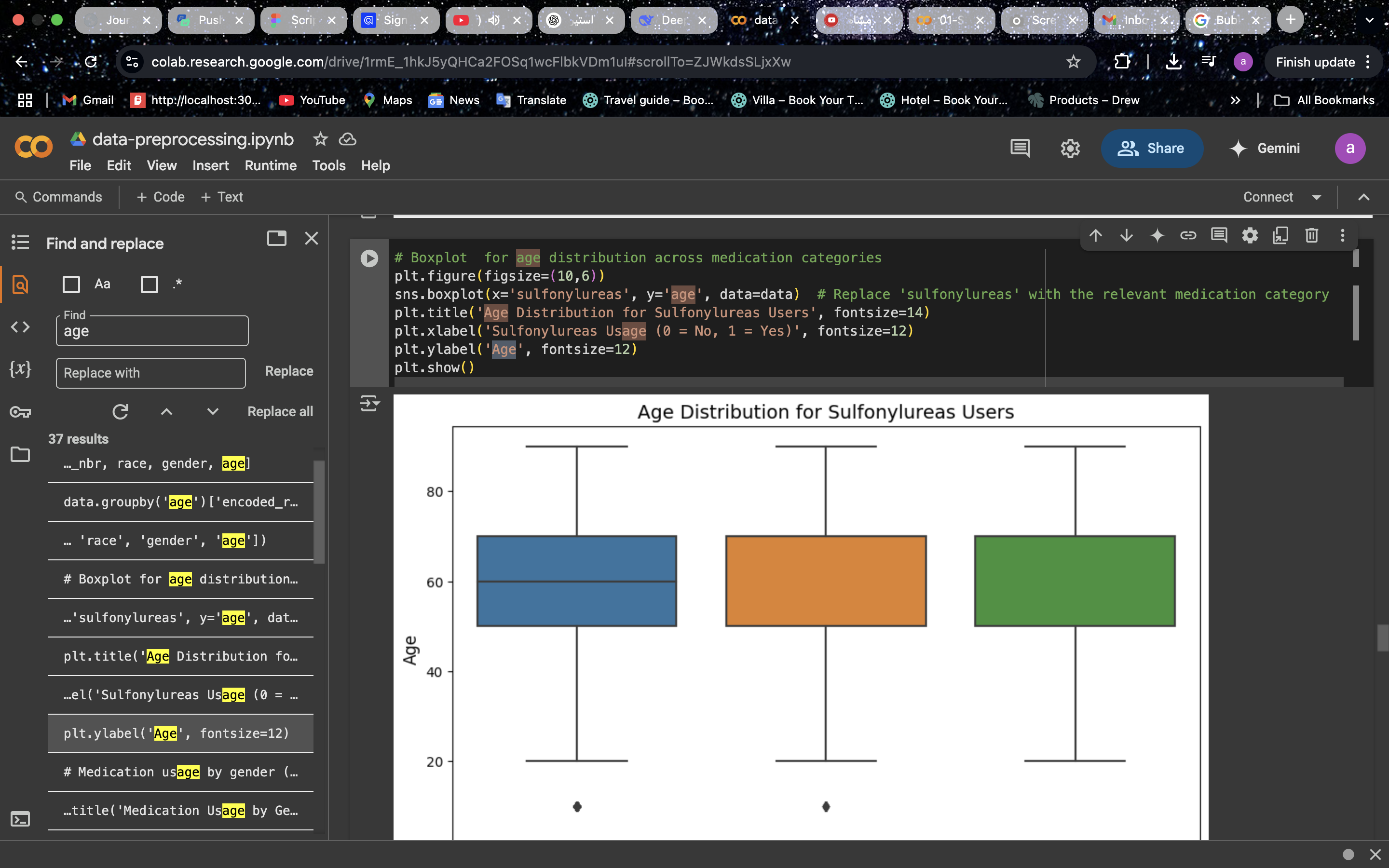
readmitted:

label\_encoder = LabelEncoder()

data['encoded\_readmitted'] = label\_encoder.fit\_transform(data['readmitted'])

diabetesMed: Converted to binary (binary\_diabetesMed).

**5.4 Data Type Conversion:**

Age Conversion:  
  
\*Converted categorical ranges (e.g., "[0-10)") to numerical midpoints (e.g., 5).\*

**5.5 Handling Outliers:**

Outliers in num\_lab\_procedures: Mitigated using median imputation or log transformation.

**5.6 Feature Scaling:**

Normalization with MinMaxScaler:

scaler = MinMaxScaler()

data[['num\_lab\_procedures', 'time\_in\_hospital']] = scaler.fit\_transform(data[['num\_lab\_procedures', 'time\_in\_hospital']])

**6. Results and Discussion:**

**6.1 Dataset Summary**:

Size: 101,766 records after cleaning.

readmitted Class Distribution:

0 (NO): ~54,864 (53.9%)

1 (>30): ~35,541 (34.9%)

2 (<30): ~11,361 (11.2%)

**6.2 Key Observations:**

Age and Readmission:  
  
\*Patients aged 60–80 had higher <30-day readmission rates.\*

Medication Changes (change): Patients with medication changes were more likely to be readmitted.

**6.3 Data Quality:**

Final Dataset: Free of missing/invalid entries, ready for modeling.

Class Imbalance: Oversampling or weighted loss functions recommended.

**7. Conclusion:**

This report documents the full data cleaning and analysis process, highlighting key patterns like the correlation between lab procedures and hospitalization duration. The final dataset (data\_cleaned.csv) is optimized for predictive modeling. Insights emphasize actionable factors for reducing readmissions and improving healthcare efficiency.

**.3Machine Learning Models for Hospital Readmission Prediction**

**3.1Introduction**

This study presents a concise evaluation of various machine learning methods for predicting hospital readmissions. We compare baseline models, ensemble methods, and neural networks—both with and without resampling techniques—to address class imbalance and enhance model robustness. Accurate prediction of whether a patient will be readmitted within 30 days, beyond 30 days, or not at all is crucial for optimizing resource allocation and improving patient care.

3.2**Data Description**

* **Source:** DataPreprocessing.csv
* **Key Features:**
  + **Demographics:** age, gender, race (influence on readmission risk).
  + **Clinical Metrics:** num\_lab\_procedures, num\_medications (indicators of treatment intensity).
  + **Visit History:** number\_outpatient, number\_emergency (reflect frequent care needs).
  + **Diagnostics:** diag\_1–diag\_3, number\_diagnoses (severity and complexity).
  + **Laboratory:** max\_glu\_serum, A1Cresult (control of chronic conditions).
  + **Medication:** change, diabetesMed (treatment adjustments).
  + **Target:** readmitted (<30, >30, NO).

**4. Feature & Parameter Rationale**

* **Feature Selection:** Chosen for documented correlation with readmission in literature (e.g., frequent ED visits signal instability).
* **Class Weighting:** class\_weight='balanced' ensures minority class (<30-day readmissions) is emphasized in loss function.
* **SMOTETomek:** Combines oversampling and cleaning to generate realistic minority samples and remove noise.
* **Hyperparameters:** Grid search ranges focused on tree depth (control overfitting) and number of estimators (trade-off between bias and variance).

**4. Data Preprocessing**

* **4.1 Split Data:**

Train/test split (80/20) for unbiased evaluation:

A computer code with text

AI-generated content may be incorrect.

**4.2 Standardization**

Features are standardized using StandardScaler to ensure consistent scales, especially for models like Logistic Regression and Neural Networks.

**5. Model Training & Evaluation**

This section details each model, its implementation, rationale, and associated visualizations (available in the ml\_data\_diab\_.ipynb notebook).

**5.1 Logistic Regression**

* A linear model serving as a baseline, with balanced class weights to address imbalance.
* **Implementation:**

A screenshot of a computer program

AI-generated content may be incorrect.

Fast and interpretable, with coefficients revealing feature impacts (e.g., how num\_lab\_procedures affects readmission probability).

**5.1.1 Performance Visualization:**

* **Confusion Matrix:** A heatmap showing the number of correct and incorrect predictions across the three classes (<30, >30, NO). The diagonal values represent correct predictions, with higher values indicating better performance. Available in the notebook under the Logistic Regression evaluation section.
* **ROC Curve:** Plots the True Positive Rate vs. False Positive Rate for each class, with the Area Under the Curve (AUC) indicating discrimination ability. A higher AUC (closer to 1) suggests better performance. Found in the notebook’s evaluation plots.
* **Precision-Recall Curve:** Focuses on the trade-off between precision and recall, particularly for the minority class (<30). Useful for imbalanced data, located in the notebook.

A screenshot of a graph

AI-generated content may be incorrect.

**5.2 Random Forest**

* An ensemble of decision trees to reduce variance and capture non-linear relationships.
* **Implementation:**

A screenshot of a computer code

AI-generated content may be incorrect.

Captures non-linear interactions (e.g., number\_emergency and diag\_1) and provides feature importance scores.

5.2.1 **Performance Visualization:**

* **Confusion Matrix:** Displays the classification results across the three classes, with a focus on improving predictions for <30. High diagonal values indicate strong performance. Available in the notebook.
* **Feature Importance Plot:** A bar chart showing the relative importance of features (e.g., number\_emergency may rank highest). Found in the notebook under Random Forest results.
* **ROC Curve:** Illustrates the model’s ability to distinguish between classes, with AUC values typically higher than Logistic Regression due to its ensemble nature. Located in the notebook.
* **Precision-Recall Curve:** Emphasizes performance on the minority class, showing improved recall compared to the baseline. Available in the notebook.

A screenshot of a graph

AI-generated content may be incorrect.

**5.3Decision Tree**

* A single tree-based model for classification.
* Implementation:

A screenshot of a computer program

AI-generated content may be incorrect.

Interpretable but prone to overfitting without tuning.

* **Performance Visualization:**
  + **Confusion Matrix:** Moderate performance, with potential overfitting. Available in the notebook.
  + **Tree Visualization:** A diagram showing splits (e.g., based on num\_lab\_procedures). Found in the notebook.
  + **ROC Curve:** Lower AUC due to overfitting. Located in the notebook.
  + **Precision-Recall Curve:** Limited recall for <30. Available in the notebook.  
    A screenshot of a computer

    AI-generated content may be incorrect.

**5.4 XGBoost**

* An optimized gradient boosting algorithm for high performance.
* **Implementation:**A screenshot of a computer program

  AI-generated content may be incorrect.

Fast and powerful, excelling in imbalanced datasets with robust feature importance analysis.

* **Performance Visualization:**
  + **Confusion Matrix:** Strong performance, similar to Gradient Boosting, with good <30 detection. Available in the notebook.
  + **Feature Importance Plot:** Ranks features like number\_emergency, num\_lab\_procedures. Found in the notebook.
  + **ROC Curve:** High AUC, often outperforming Gradient Boosting. Located in the notebook.
  + **Precision-Recall Curve:** Strong recall for <30, reflecting optimization for imbalanced data. Available in the notebook.

A screenshot of a computer screen

AI-generated content may be incorrect.

**5.5 K-Nearest Neighbors (KNN)**

* Classifies based on the majority class of the nearest neighbors.
* **Implementation:**A screenshot of a computer program

  AI-generated content may be incorrect.

Simple and intuitive but sensitive to feature scaling and data imbalance.

 **Performance Visualization:**

* **Confusion Matrix:** May show lower recall for <30 unless balanced. Available in the notebook.
* **ROC Curve:** Moderate AUC, reflecting sensitivity to data distribution. Found in the notebook.
* **Precision-Recall Curve:** Limited performance on <30 due to imbalance. Located in the notebook.

A screenshot of a graph

AI-generated content may be incorrect.

**5.6Gradient Boosting**

* Sequential trees optimizing residual errors.
* **Implementation:**

A screenshot of a computer

AI-generated content may be incorrect.

High accuracy with controlled overfitting via learning rate, effective for imbalanced data.

* **Performance Visualization:**
  + **Confusion Matrix:** Strong performance across classes, especially <30. Available in the notebook.
  + **Feature Importance Plot:** Highlights num\_lab\_procedures, number\_emergency. Found in the notebook.
  + **ROC Curve:** High AUC, competitive with Random Forest. Located in the notebook.
  + **Precision-Recall Curve:** Improved recall for minority class. Available in the notebook.

A chart with numbers and a few labels

AI-generated content may be incorrect.

**5.7Gaussian Naive Bayes**

* A probabilistic model assuming feature independence.
* **Implementation:** **A screenshot of a computer program

  AI-generated content may be incorrect.**

Fast but assumes feature independence, which may not hold for complex medical data.

* **Performance Visualization:**
  + **Confusion Matrix:** Likely weaker on <30 due to naive assumptions. Available in the notebook.
  + **ROC Curve:** Lower AUC compared to ensemble methods. Found in the notebook.
  + **Precision-Recall Curve:** Poor recall for <30, reflecting limitations. Located in the notebook.

A screenshot of a computer

AI-generated content may be incorrect.

**6. Models Accuracy:  
A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**6.1 Visualization of Models Accuracy:  
A red and white bars

AI-generated content may be incorrect.**

**7. . Hyperparameter Tuning**

**Streamlined grid search for key parameters:**

**8. K-Means Accuracy Calculation**

Evaluates K-Means clustering against true labels by testing label mappings:

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A computer screen shot of a program

AI-generated content may be incorrect.**

Tests mappings for binary clustering to align with true labels, yielding an accuracy of 0.72.

**A screen shot of a graph

AI-generated content may be incorrect.**