**Final Project Report Draft**


**Machine Learning Algorithms for Predicting Hospital Readmissions**

**Group Members**

Daniel Dankwa

Soumyabrata Ghosh

YNV Tanmayee

Sahana Ranganathan

**Machine Learning – STAT 4650**

**December 3rd, 2024**

# Project Report: Hospital Readmission Analysis

## 1. Introduction

## 2. Data Preparation

- **Overview of the Dataset**
- **Data Preprocessing Steps**
    - **2.1 Loading and Understanding the Dataset**
    - **2.2 Cleaning Data**
    - **2.3 Converting Categorical Variables**
    - **2.4 Creating New Features**
    - **2.5 Scaling and Normalizing Data**
    - **2.6 Target Variable Preparation**
    - **2.7 Data Splitting**

---

## 3. Exploratory Data Analysis (EDA)

- **Key Findings**:
    - **3.1 Readmission Distribution**
    - **3.2 Significant Predictors**
- **Visualization Insights**
    - **3.3 Readmission and Medication Change**
    - **3.4 Demographic Analysis**
    - **3.5 Correlation and Feature Interactions**
- **3.6 Summary of Insights**

---

## 4. Correlation Analysis

- **4.1 Strong Correlations**:
- **4.2 Feature Interaction**:
- **4.3 Heatmap Visualization**:
- **4.4 Summary of Correlation Analysis**

---

## 5. Risk Stratification

- **5.1 High-Risk Patients**
- **5.2 Risk Score Development**
- **5.3 Risk Categories**
- **5.4 Benefits of Risk Stratification**
- **5.5 Validation of the Risk Score**
- **5.6 Future Enhancements**
- **5.7 Case Study**

---

## 6. Machine Learning Analysis

- **6.1 Models Used**
- **6.2 Individual Model Analysis**
- **6.3 Combined Ensemble Model**

- **6.4 ROC-AUC Scores Comparison**
- **6.5 Feature Importance**
- **6.6 Insights and Recommendations**

---

## 7. Recommendations

- **7.1 Targeted Interventions**
- **7.2 Policy Adjustments**
- **7.3 Model Enhancements**
- **7.4 Feature Engineering**

---

## 8. Conclusion

- **8.1 Key Predictors of Readmission**
- **8.2 Proposed Interventions**
- **8.3 Modeling Improvements**
- **8.4 Resource Optimization**
- **8.5 Policy Implications**
- **8.6 Future Directions**
- **8.7 Final Thoughts**

---

### Introduction

Hospital readmissions, particularly those occurring within 30 days of discharge, are a critical metric in healthcare quality and cost management. Reducing readmissions has the potential to improve patient outcomes, optimize resource utilization, and lower healthcare costs. This project undertakes an in-depth analysis of hospital readmission patterns using a dataset comprising 25,000 patient records. These records include a rich variety of features such as demographic information, clinical details, and hospital admission histories.

The overarching aim of this analysis is threefold:

1. **Identify Key Factors Influencing Readmission:**
    - By examining the interplay between various patient and treatment attributes, the analysis seeks to uncover critical predictors of hospital readmissions. These predictors may include inpatient visit frequency, medication regimens, emergency visits, and other demographic or clinical factors.

2. **Explore Feature Relationships:**
    - The relationships between variables such as hospital stay duration, number of diagnoses, and treatment changes are examined to understand their roles in predicting readmissions. Statistical and visual methods are used to uncover trends, correlations, and potential interaction effects.
3. **Provide Actionable Insights:**
    - Insights derived from the analysis are geared toward practical applications, including designing targeted interventions for high-risk patients, optimizing discharge planning, and improving healthcare delivery strategies.

By leveraging statistical analyses and machine learning techniques, the project not only aims to understand the underlying patterns of readmissions but also to provide data-driven recommendations for reducing their frequency. These insights are essential for healthcare providers and policymakers seeking to enhance care quality and efficiency while minimizing avoidable hospital readmissions.

## 1. Data Preparation

### Overview of the Dataset

The dataset used for this project contains information on 25,000 hospital admissions. It encompasses a diverse set of features that provide valuable insights into the demographic, clinical, and administrative aspects of patient care. This includes information such as patient age, gender, race, number of inpatient visits, outpatient visits, emergency department visits, hospital stay duration, number of lab procedures, and medication details. Additionally, the dataset includes information about diagnoses, diabetes management, and whether the patient was readmitted within a specified timeframe.

Understanding the dataset is critical for building a robust analytical framework. It provides the foundation for identifying patterns and predictors of readmissions. Before any meaningful analysis can be performed, the data must undergo several preprocessing steps to ensure its quality and suitability for the intended analytical techniques.

### Data Dictionary

| Column Name | Description | Data Type |
|---|---|---|
| **time_in_hospital** | Number of days the patient stayed in the hospital. | int64 |
| **num_lab_procedures** | Number of lab procedures performed during the hospital stay. | int64 |
| **num_procedures** | Number of procedures (excluding lab tests) performed during the hospital stay. | int64 |
| **num_medications** | Number of distinct medications prescribed during the hospital stay. | int64 |
| **number_outpatient** | Number of outpatient visits before the hospitalization. | int64 |
| **number_emergency** | Number of emergency department visits before the hospitalization. | int64 |
| **number_inpatient** | Number of inpatient visits before the current hospitalization. | int64 |
| **number_diagnoses** | Number of diagnoses recorded during the hospital stay. | int64 |
| **race_Caucasian** | Indicator if the patient is Caucasian (1 = Yes, 0 = No). | bool |
| **race_AfricanAmerican** | Indicator if the patient is African American (1 = Yes, 0 = No). | bool |

| | | |
|---|---|---|
| **gender_Female** | Indicator if the patient is female (1 = Yes, 0 = No). | bool |
| **age_[70-80)** | Indicator if the patient is between 70 and 80 years old. | bool |
| **age_[60-70)** | Indicator if the patient is between 60 and 70 years old. | bool |
| **age_[50-60)** | Indicator if the patient is between 50 and 60 years old. | bool |
| **age_[80-90)** | Indicator if the patient is between 80 and 90 years old. | bool |
| **age_[40-50)** | Indicator if the patient is between 40 and 50 years old. | bool |
| **payer_code_?** | Missing/unknown payer code indicator. | bool |
| **payer_code_MC** | Indicator if the payer code is MC (Medicare). | bool |
| **payer_code_HM** | Indicator if the payer code is HM (HMO). | bool |
| **payer_code_SP** | Indicator if the payer code is SP (Self-Pay). | bool |
| **payer_code_BC** | Indicator if the payer code is BC (Blue Cross). | bool |
| **medical_specialty_?** | Missing/unknown medical specialty indicator. | bool |
| **medical_specialty_InternalMedicine** | Indicator if the specialty is Internal Medicine. | bool |
| **medical_specialty_Emergency/Trauma** | Indicator if the specialty is Emergency/Trauma. | bool |
| **medical_specialty_Family/GeneralPractice** | Indicator if the specialty is Family/General Practice. | bool |
| **medical_specialty_Cardiology** | Indicator if the specialty is Cardiology. | bool |
| **diag_1_428** | Indicator if primary diagnosis code is 428 (Heart Failure). | bool |
| **diag_1_414** | Indicator if primary diagnosis code is 414 (Ischemic Heart Disease). | bool |
| **diag_1_786** | Indicator if primary diagnosis code is 786 (Respiratory Symptoms). | bool |
| **diag_2_276** | Indicator if secondary diagnosis code is 276 (Disorders of Fluid Balance). | bool |
| **diag_2_428** | Indicator if secondary diagnosis code is 428 (Heart Failure). | bool |
| **diag_2_250** | Indicator if secondary diagnosis code is 250 (Diabetes Mellitus). | bool |
| **diag_2_427** | Indicator if secondary diagnosis code is 427 (Cardiac Dysrhythmias). | bool |
| **diag_3_250** | Indicator if tertiary diagnosis code is 250 (Diabetes Mellitus). | bool |
| **diag_3_401** | Indicator if tertiary diagnosis code is 401 (Hypertension). | bool |
| **diag_3_276** | Indicator if tertiary diagnosis code is 276 (Disorders of Fluid Balance). | bool |
| **diag_3_428** | Indicator if tertiary diagnosis code is 428 (Heart Failure). | bool |
| **max_glu_serum_None** | Indicator if no glucose serum test was performed. | bool |
| **A1Cresult_None** | Indicator if no A1C test result is recorded. | bool |
| **metformin_No** | Indicator if Metformin was not prescribed. | bool |
| **repaglinide_No** | Indicator if Repaglinide was not prescribed. | bool |
| **nateglinide_No** | Indicator if Nateglinide was not prescribed. | bool |
| **chlorpropamide_No** | Indicator if Chlorpropamide was not prescribed. | bool |
| **glimepiride_No** | Indicator if Glimepiride was not prescribed. | bool |
| **acetohexamide_No** | Indicator if Acetohexamide was not prescribed. | bool |
| **glipizide_No** | Indicator if Glipizide was not prescribed. | bool |
| **glyburide_No** | Indicator if Glyburide was not prescribed. | bool |
| **tolbutamide_No** | Indicator if Tolbutamide was not prescribed. | bool |
| **pioglitazone_No** | Indicator if Pioglitazone was not prescribed. | bool |

| | | |
|---|---|---|
| **rosiglitazone_No** | Indicator if Rosiglitazone was not prescribed. | bool |
| **acarbose_No** | Indicator if Acarbose was not prescribed. | bool |
| **miglitol_No** | Indicator if Miglitol was not prescribed. | bool |
| **troglitazone_No** | Indicator if Troglitazone was not prescribed. | bool |
| **tolazamide_No** | Indicator if Tolazamide was not prescribed. | bool |
| **examide_No** | Indicator if Examide was not prescribed. | bool |
| **citoglipton_No** | Indicator if Citoglipton was not prescribed. | bool |
| **insulin_No** | Indicator if Insulin was not prescribed. | bool |
| **glyburide-metformin_No** | Indicator if Glyburide-Metformin combination was not prescribed. | bool |
| **glipizide-metformin_No** | Indicator if Glipizide-Metformin combination was not prescribed. | bool |
| **glimepiride-pioglitazone_No** | Indicator if Glimepiride-Pioglitazone combination was not prescribed. | bool |
| **metformin-rosiglitazone_No** | Indicator if Metformin-Rosiglitazone combination was not prescribed. | bool |
| **metformin-pioglitazone_No** | Indicator if Metformin-Pioglitazone combination was not prescribed. | bool |
| **change_No** | Indicator if there was no change in medication. | bool |
| **diabetesMed_Yes** | Indicator if diabetes medication was prescribed. | bool |
| **readmitted** | Target variable: 1 if the patient was readmitted, 0 otherwise. | int64 |

---

**Data Preprocessing Steps**

**1.1 Loading and Understanding the Dataset**

The first step in preparing the data involved loading the dataset and conducting an initial exploration to understand its structure, dimensions, and key characteristics. This step included:

- **Examining the Columns:** The dataset contains 65 columns, each representing different attributes related to the patients and their hospitalizations. These columns vary in data types, such as numeric, categorical, and boolean values.
- **Assessing Missing Values:** Missing data is a common challenge in healthcare datasets. Identifying missing values was crucial to determine whether data imputation or exclusion was necessary.
- **Data Types:** The dataset contained boolean columns representing binary categorical variables. These columns were reviewed to determine whether their current format was compatible with the intended analyses.

**1.2 Cleaning Data**

**Handling Missing Values:**

- Several columns in the dataset, such as payer codes and medical specialties, contained missing or ambiguous values. These were represented by placeholders like ?.
- Missing data was addressed using imputation strategies, such as replacing missing categorical values with a mode or creating a separate category for missing data (e.g., "Unknown").
- Numeric columns with missing values were imputed using median or mean values to avoid skewing the data distribution.

**Standardizing Column Names:**

- Column names were cleaned and standardized to ensure consistency and ease of use during analysis. For example, spaces were replaced with underscores (_), and names were converted to lowercase.

---

### 1.3 Converting Categorical Variables

**Boolean to Binary Conversion:**

- Many categorical columns in the dataset were stored as boolean values (True/False). To make these columns usable for mathematical computations, they were converted into binary numeric values:
    - True was replaced with 1.
    - False was replaced with 0.

For instance, the column change_No, which indicates whether there was a change in medication (True for no change, False for a change), was converted to numeric binary values.

**One-Hot Encoding:**

- Categorical columns with more than two unique values, such as race and medical specialty, were transformed using one-hot encoding. This process involved creating a new binary column for each category. For example:
    - The race column, which included values such as "Caucasian," "African American," and "Asian," was converted into separate columns like race_Caucasian, race_AfricanAmerican, and race_Asian, each containing binary indicators (1 or 0).

**Advantages of Encoding:**

- These transformations enabled the dataset to be compatible with machine learning algorithms that require numeric input.
- Encoding also helped capture the multi-class nature of some variables without introducing ordinal relationships that do not exist.

---

### 1.4 Creating New Features

**Feature Engineering:** Feature engineering was an essential part of the preprocessing pipeline. It involved creating new variables that could provide additional predictive power or simplify complex relationships between existing features.

1. **Interaction Terms:**
    - Interaction terms were created by combining variables to explore their joint effects on the target variable (readmitted). Examples include:
        - diagnoses_x_inpatient: Product of the number of diagnoses and inpatient visits.
        - medications_x_emergency: Product of the number of medications and emergency visits.

2. **Risk Scores:**
    - A composite risk score was calculated using weighted contributions from key variables like inpatient visits, emergency visits, medication counts, and hospital stay duration. This new feature provided a single metric to quantify a patient's likelihood of readmission.
3. **Time-Based Features:**
    - The average time between hospital admissions was calculated to capture the frequency of hospital utilization. This was derived as:
        - avg_time_between_admissions = time_in_hospital / (number_inpatient + 1)
    - Similarly, the total time spent in the hospital during previous admissions was estimated using:
        - time_in_hospital_previous = time_in_hospital * (number_inpatient - 1)

**Advantages of Feature Engineering:**

- These derived features helped uncover complex relationships and patterns that might not be evident from the raw data alone.
- They also allowed for more nuanced analyses and improved model performance.

---

### 1.5 Scaling and Normalizing Data

**Why Scaling is Necessary:**

- Some features, such as num_medications and time_in_hospital, had large ranges, while others, like binary columns, ranged from 0 to 1.
- Machine learning models, especially those based on gradient descent, are sensitive to feature scaling. Uneven ranges can lead to biased learning processes.

**Techniques Used:**

- **Standardization:** Features were standardized to have a mean of 0 and a standard deviation of 1.
- **Normalization:** Continuous variables were normalized to bring them within a fixed range (e.g., [0, 1]) when necessary.

**Outcome:**

- Scaling ensured that no single feature dominated the training process due to its magnitude.

---

### 1.6 Target Variable Preparation

**Defining the Target:**

- The target variable readmitted was binary, indicating whether a patient was readmitted (1) or not (0).
- This required ensuring that the target variable was balanced and representative of the dataset.

**Addressing Class Imbalance:**

- Initial exploration revealed a potential imbalance in the target variable, with more non-readmitted cases than readmitted ones.
- To address this:
    - Oversampling techniques like SMOTE (Synthetic Minority Oversampling Technique) were considered to balance the dataset.
    - Alternative weighting strategies were applied to adjust for imbalance during model training.

---

### 1.7 Data Splitting

**Train-Test Split:**

- The dataset was split into training and testing sets to ensure that model performance was evaluated on unseen data.
- A 80:20 split was used, with 80% of the data allocated for training and 20% for testing.

**Stratified Sampling:**

- Stratification was applied during the split to preserve the proportion of readmitted cases in both training and testing sets.

---

### Final Dataset Characteristics

After preprocessing, the dataset was transformed into a clean, analysis-ready format with the following characteristics:

- **Dimensions:** The dataset consisted of 25,000 rows and 85 columns, including engineered features and one-hot-encoded variables.
- **Numeric Format:** All features were numeric, enabling compatibility with a wide range of statistical and machine learning methods.
- **Balanced Target:** Adjustments ensured that the target variable distribution was balanced for robust model evaluation.

---

### Importance of Data Preparation

The steps taken during data preparation are vital for ensuring the accuracy and reliability of subsequent analyses. By addressing missing values, encoding categorical variables, scaling data, and engineering meaningful features, we established a solid foundation for exploratory data analysis, model building, and actionable insight generation. Proper preprocessing not only improves the performance of predictive models but also enhances the interpretability and practical utility of the results in real-world healthcare scenarios.

## 2. Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is a critical step in any data-driven project as it provides a comprehensive understanding of the dataset, identifies patterns, and highlights key trends that inform further analysis. In this project, EDA was employed to analyze the dataset of 25,000 hospital admissions with a focus on understanding hospital readmission rates and the factors influencing them.

---

## Key Findings

### 2.1 Readmission Distribution

One of the first steps in EDA involved understanding the overall distribution of the target variable, readmitted, which indicates whether a patient was readmitted within a specified period. The analysis revealed that approximately **45% of patients were readmitted**, while 55% were not.

- **Implications:**
  - A readmission rate of 45% is significant, highlighting the need for targeted interventions to reduce avoidable readmissions.
  - The nearly balanced distribution between readmitted and non-readmitted patients allows for meaningful comparisons and model-building without severe class imbalance.
- **Visualization:** A bar plot was created to depict the distribution of readmissions. The height of the bars clearly showed the proportion of patients who were readmitted versus those who were not.

---

### 2.2 Significant Predictors of Readmission

The analysis aimed to identify features that strongly influence the likelihood of readmission. Several variables were examined, with the following insights:

#### 2.2.1 Increased Inpatient Visits

- **Observation:**
  - Patients with more frequent inpatient visits had a significantly higher likelihood of being readmitted.
  - A scatter plot of number_inpatient versus readmitted showed that patients with three or more inpatient visits had a notably higher proportion of readmissions.
- **Interpretation:**
  - Frequent inpatient visits may indicate chronic or severe medical conditions that require recurring care, making these patients more prone to readmission.
- **Implications:**
  - These patients represent a high-risk group that could benefit from targeted post-discharge care and monitoring.

#### 2.2.2 Higher Medication Counts

- **Observation:**
  - A positive correlation was observed between the number of medications prescribed (num_medications) and readmission rates.
  - Patients on a larger number of medications (e.g., 20+) were more likely to be readmitted compared to those on fewer medications.

- **Visualization:**
  - A boxplot comparing num_medications for readmitted versus non-readmitted patients highlighted that the median medication count was higher for readmitted patients.
- **Interpretation:**
  - High medication counts could indicate complex treatment regimens or underlying health conditions, increasing the risk of complications and subsequent readmissions.
- **Implications:**
  - Medication management programs and closer monitoring of patients with high medication counts could help reduce readmission rates.

### 2.2.3 Lesser Predictive Power from the Number of Diagnoses

- **Observation:**
  - The number of diagnoses (number_diagnoses) showed only a weak correlation with readmission rates.
  - Patients with more than ten diagnoses had a slightly higher likelihood of readmission, but the relationship was not as pronounced as with inpatient visits or medication counts.
- **Interpretation:**
  - While the number of diagnoses reflects the complexity of a patient's condition, it may not directly predict the likelihood of readmission without considering other factors like the type or severity of diagnoses.

### 2.2.4 Hospital Stay Duration

- **Observation:**
  - The duration of hospital stays (time_in_hospital) had a minimal impact on readmission rates.
  - Patients with longer stays did not show significantly different readmission rates compared to those with shorter stays.
- **Interpretation:**
  - Hospital stay duration may not independently influence readmissions but could interact with other factors like inpatient visits and diagnoses.

---

### Visualization Insights

To further understand the dataset, various visualizations were created to analyze the relationships between features and the target variable. These visualizations provided deeper insights into key trends.

### 2.3 Readmission and Medication Change

- **Observation:**
  - Patients whose medication regimens were changed during their hospital stay (change_No) exhibited slightly higher readmission rates compared to those whose regimens remained unchanged.
- **Visualization:**
  - A bar plot showing the proportion of readmitted versus non-readmitted patients for each category of change_No revealed this pattern.
- **Interpretation:**

    - Changes in medication could indicate the adjustment of treatment plans for unstable or worsening conditions, which might contribute to higher readmission rates.
- **Implications:**
    - Monitoring patients whose medications have been changed and ensuring they understand their new regimens could help prevent complications leading to readmissions.

---

### 2.4 Demographic Analysis

#### 2.4.1 Age Group Analysis

- **Observation:**
    - Among various age groups, patients in the [80–90) age range exhibited a slightly higher readmission rate compared to other groups.
    - The lowest readmission rates were observed in the [50–60) age group.
- **Visualization:**
    - A bar chart comparing readmission rates across age groups showed the gradual increase in rates for older patients.
- **Interpretation:**
    - Elderly patients, particularly those in the [80–90) range, are more vulnerable to readmission due to age-related health issues and comorbidities.
- **Implications:**
    - Tailored interventions for elderly patients, such as comprehensive discharge planning and follow-up care, could help mitigate readmission risks.

#### 2.4.2 Gender-Based Analysis

- **Observation:**
    - Gender differences were evident in the correlation between inpatient visits and readmission rates.
    - Female patients showed a stronger correlation between the number of inpatient visits and readmission likelihood compared to males.
- **Visualization:**
    - A grouped bar chart comparing top predictors of readmission for males and females revealed these gender-specific trends.
- **Interpretation:**
    - The stronger correlation for females may stem from gender-specific health conditions or differences in healthcare-seeking behavior.
- **Implications:**
    - Gender-specific strategies, such as addressing particular health concerns prevalent in females, could enhance the effectiveness of interventions.

---

### 2.5 Correlation and Feature Interactions

#### 2.5.1 Correlation Heatmap

- A heatmap was created to visualize the correlations between various features and readmitted.
- Key observations:

- Strong positive correlations were observed between readmitted and variables like number_inpatient and num_medications.
- Weak or negligible correlations were found for features like num_procedures.

### 2.5.2 Feature Interactions

- Interaction terms such as diagnoses_x_inpatient and medications_x_emergency were explored to assess their combined effects on readmissions.
- Scatter plots and boxplots highlighted that these interaction terms captured relationships not evident from individual features alone.

---

### 2.6 Summary of Insights

The EDA revealed several critical insights into hospital readmissions:

1. **High-Risk Groups:**
   - Patients with frequent inpatient visits and high medication counts are at elevated risk of readmission.
   - Elderly patients, particularly in the [80–90) age group, require targeted care.
2. **Potential Predictors:**
   - Inpatient visits and medication counts emerged as strong predictors of readmissions, while variables like hospital stay duration and number of diagnoses had limited predictive power.
3. **Demographic and Gender Considerations:**
   - Readmission patterns varied across age groups and genders, highlighting the need for tailored interventions.
4. **Feature Interactions:**
   - Interaction terms and derived features provided additional layers of insight, suggesting avenues for feature engineering to improve predictive modeling.

---

### Conclusion

The EDA phase of this project provided a comprehensive understanding of the dataset and the factors influencing hospital readmissions. By identifying key predictors, demographic trends, and potential interactions, the analysis set the stage for targeted interventions and effective predictive modeling. The insights gained from EDA will inform subsequent steps in the project, including model development and policy recommendations to reduce readmission rates and enhance patient outcomes.

### 3. Correlation Analysis

Correlation analysis is a vital step in understanding the relationships between different features in a dataset. It provides insights into how various variables interact with the target variable—in this case, hospital readmission—and with each other. This analysis not only helps identify the most influential predictors but also guides feature selection and engineering to enhance model performance. In this project, correlation analysis focused on quantifying the strength of relationships between features and readmission, exploring feature interactions, and visualizing these relationships using heatmaps and other tools.

### 3.1 Strong Correlations

The analysis identified several features with strong correlations to hospital readmission. These features are critical in predicting whether a patient is likely to be readmitted.

---

#### 3.1.1 Number of Inpatient Visits

- **Observation:**
  - The number of inpatient visits (number_inpatient) showed the strongest positive correlation with the target variable readmitted.
  - Patients with frequent inpatient visits had a substantially higher likelihood of being readmitted.
- **Interpretation:**
  - Frequent inpatient visits often indicate chronic health conditions or poor management of existing conditions, both of which increase the risk of readmission.
- **Implications:**
  - Patients with multiple inpatient visits should be flagged for targeted interventions, such as follow-up care, chronic disease management programs, and enhanced discharge planning.
- **Visualization:**
  - A scatter plot of number_inpatient against readmitted demonstrated a clear trend where higher values of number_inpatient corresponded to higher proportions of readmitted patients.

---

#### 3.1.2 Number of Emergency Visits

- **Observation:**
  - The number of emergency visits (number_emergency) also showed a strong positive correlation with readmitted.
  - Patients with frequent emergency department visits were more likely to experience readmissions.
- **Interpretation:**
  - Emergency visits may indicate acute episodes or poorly controlled conditions that could result in subsequent readmissions.
- **Implications:**
  - Enhanced emergency department protocols, such as ensuring that high-risk patients receive proper follow-up care, could help reduce readmissions.
- **Visualization:**
  - A bar chart comparing readmission rates for patients with varying numbers of emergency visits highlighted the upward trend with increasing visits.

---

#### 3.1.3 Number of Outpatient Visits

- **Observation:**

    - The number of outpatient visits (number_outpatient) showed a moderate positive correlation with readmitted.
    - Unlike inpatient and emergency visits, the correlation for outpatient visits was less pronounced but still significant.
- **Interpretation:**
    - While outpatient visits typically aim to manage chronic conditions and prevent readmissions, a higher frequency could also reflect ongoing health issues requiring further care.
- **Implications:**
    - Outpatient visits should be analyzed in conjunction with other features, such as inpatient and emergency visits, to develop a comprehensive understanding of patient care patterns.
- **Visualization:**
    - A line plot of average readmission rates against the number of outpatient visits provided a nuanced view of this relationship.

---

### 3.2 Feature Interaction

Feature interaction analysis goes beyond examining individual correlations by exploring how combined features influence the target variable. In this analysis, interaction terms were created to capture these effects.

---

#### 3.2.1 Diagnoses × Inpatient Visits

- **Observation:**
    - The interaction term diagnoses × inpatient visits was explored to assess whether the combined effect of the number of diagnoses and inpatient visits provided additional predictive power.
    - Patients with a high number of diagnoses and frequent inpatient visits were significantly more likely to be readmitted.
- **Interpretation:**
    - This interaction term likely captures the combined burden of complex conditions (diagnoses) and hospital dependency (inpatient visits).
- **Implications:**
    - This feature could be used in predictive models to identify patients with particularly high readmission risks.
- **Visualization:**
    - A heatmap plotting diagnoses × inpatient visits against readmitted highlighted clusters of high-risk patients.

---

#### 3.2.2 Medications × Emergency Visits

- **Observation:**
    - The interaction term medications × emergency visits was another feature explored for its predictive utility.

    - Patients on a high number of medications who also had frequent emergency visits showed higher readmission rates.
- **Interpretation:**
    - This interaction likely reflects a subset of patients with poorly managed conditions requiring both intensive pharmacological intervention and acute care.
- **Implications:**
    - Enhanced monitoring of these patients, including regular medication reviews and emergency visit follow-ups, could help mitigate readmission risks.
- **Visualization:**
    - A 3D scatter plot showing medications × emergency visits and readmitted offered a visual representation of this interaction.

---

### 3.2.3 Average Time Between Admissions

- **Observation:**
    - A derived feature, avg_time_between_admissions, was calculated as the ratio of time_in_hospital to number_inpatient + 1. This feature provided insights into the frequency of hospital utilization.
    - Patients with shorter intervals between admissions had higher readmission rates.
- **Interpretation:**
    - Frequent admissions suggest ongoing or worsening health issues, making these patients high-risk.
- **Implications:**
    - Identifying and addressing the underlying causes of frequent admissions could significantly reduce readmissions.
- **Visualization:**
    - A boxplot comparing avg_time_between_admissions for readmitted and non-readmitted patients revealed a clear separation between the two groups.

---

## 3.3 Heatmap Visualization

Heatmaps were used extensively to visualize the correlations between features and with the target variable. These visual tools provided a holistic view of the relationships in the dataset.

---

### 3.3.1 Overall Heatmap

- **Description:**
    - A heatmap of all feature correlations was generated to identify patterns and highlight the strongest relationships.
- **Key Observations:**
    - number_inpatient, number_emergency, and number_outpatient had the highest positive correlations with readmitted.
    - Features like num_procedures and time_in_hospital showed weaker correlations.
- **Advantages:**

- The heatmap provided a quick and intuitive way to identify the most and least influential features.

---

### 3.3.2 Focused Heatmap for Top Features

- **Description:**
  - A focused heatmap of the top 10 features most correlated with readmitted was created to zoom in on the strongest predictors.
- **Insights:**
  - The heatmap revealed clusters of features that were strongly interrelated, such as number_inpatient, number_emergency, and derived interaction terms.
- **Implications:**
  - These clusters indicated potential redundancy or multicollinearity, guiding feature selection for predictive modeling.

---

### 3.3.3 Interaction Terms Heatmap

- **Description:**
  - A heatmap of the interaction terms and their correlations with readmitted highlighted the added value of these derived features.
- **Insights:**
  - Interaction terms like diagnoses × inpatient visits and medications × emergency visits showed significant correlations with the target variable, supporting their inclusion in predictive models.

---

### 3.4 Summary of Correlation Analysis

The correlation analysis provided several critical insights:

1. **Strong Predictors:**
   - Features like number_inpatient, number_emergency, and number_outpatient emerged as the most significant predictors of readmission.
2. **Utility of Interaction Terms:**
   - Derived features, such as diagnoses × inpatient visits and medications × emergency visits, captured complex relationships and enhanced the predictive power of the dataset.
3. **Visualization Tools:**
   - Heatmaps and scatter plots were instrumental in identifying patterns and guiding feature engineering.

---

### Conclusion

The correlation analysis not only highlighted the most influential features but also demonstrated the value of exploring feature interactions and using advanced visualization tools. These insights set the

foundation for building robust predictive models and designing targeted interventions to reduce hospital readmissions. By focusing on high-risk patients identified through these analyses, healthcare providers can improve patient outcomes and optimize resource utilization.

## 4. Risk Stratification

Risk stratification is a critical process in healthcare analytics, enabling the identification of patient groups with varying levels of readmission risk. By categorizing patients into low, medium, and high-risk categories, healthcare providers can allocate resources more effectively and design targeted interventions to reduce readmissions. This section focuses on identifying high-risk patients, defining risk thresholds, and developing a composite risk score to refine stratification further.

---

### 4.1 High-Risk Patients

#### 4.1.1 Defining High-Risk Patients

Patients with more than two inpatient visits were categorized as high-risk based on their readmission patterns. This threshold was chosen after analyzing the distribution of readmission rates across different levels of inpatient visits.

- **Key Metrics:**
  - High-risk patients: Defined as those with more than two inpatient visits.
  - Low-risk patients: Defined as those with two or fewer inpatient visits.

#### 4.1.2 Observed Readmission Rates

- High-risk patients exhibited a readmission rate of approximately **70%**.
- In contrast, low-risk patients had a readmission rate of around **45%**.

#### 4.1.3 Implications of High-Risk Classification

- **Healthcare Costs:** Patients with frequent inpatient visits often incur higher healthcare costs due to repeated admissions and resource utilization.
- **Patient Outcomes:** These patients are likely to have complex or chronic conditions requiring ongoing care and management.
- **Intervention Priority:** Identifying high-risk patients allows healthcare systems to prioritize resources for post-discharge care, reducing the likelihood of readmissions.

#### 4.1.4 Visualization

- A bar chart comparing the readmission rates of high-risk and low-risk patients clearly highlighted the disparity, with high-risk patients showing significantly elevated readmission probabilities.

---

### 4.2 Risk Score Development

While the binary classification of patients as high-risk or low-risk based on inpatient visits provides valuable insights, it is essential to consider a more nuanced approach. To this end, a composite risk score was developed by combining multiple factors influencing readmissions.

### 4.2.1 Components of the Risk Score

The risk score incorporated the following features, each weighted based on its importance as determined through correlation analysis:

1. **Number of Inpatient Visits (Weight: 0.4):**
    - The strongest predictor of readmissions, reflecting the frequency of hospitalizations.
    - Patients with frequent inpatient visits were given higher scores.
2. **Number of Emergency Visits (Weight: 0.3):**
    - Indicates acute episodes requiring urgent care, a strong correlate of health instability.
3. **Medication Counts (Weight: 0.2):**
    - Reflects the complexity of treatment plans. Higher medication counts were associated with increased readmission risk.
4. **Hospital Stay Duration (Weight: 0.1):**
    - While less predictive independently, this feature provided additional context when combined with others.

### 4.2.2 Calculation of Risk Score

The risk score was calculated as a weighted sum of these components:

Risk Score=(0.4×Number of Inpatient Visits)+(0.3×Number of Emergency Visits)+(0.2×Medication Counts)+(0.1×Hospital Stay Duration)\text{Risk Score} = (0.4 \times \text{Number of Inpatient Visits}) + (0.3 \times \text{Number of Emergency Visits}) + (0.2 \times \text{Medication Counts}) + (0.1 \times \text{Hospital Stay Duration})Risk Score=(0.4×Number of Inpatient Visits)+(0.3×Number of Emergency Visits)+(0.2×Medication Counts)+(0.1×Hospital Stay Duration)

### 4.2.3 Interpretation of the Risk Score

- **High Risk:** Patients with scores above a defined threshold (e.g., 0.7) were classified as high-risk.
- **Medium Risk:** Patients with scores between 0.4 and 0.7 were classified as medium-risk.
- **Low Risk:** Patients with scores below 0.4 were classified as low-risk.

### 4.2.4 Advantages of the Risk Score

- **Precision:** By combining multiple factors, the risk score provided a more accurate stratification compared to relying on a single variable.
- **Actionability:** The score enabled healthcare providers to focus on specific patient subgroups based on their risk levels.
- **Scalability:** The methodology can be easily adapted for other healthcare datasets or scenarios.

---

## 4.3 Risk Categories

### 4.3.1 Low-Risk Patients

- **Characteristics:**
  - Fewer inpatient and emergency visits.
  - Simple or manageable health conditions requiring minimal medical intervention.
- **Readmission Rate:** Approximately 20–30%.
- **Intervention Strategies:**
  - Routine follow-ups and basic patient education.

#### 4.3.2 Medium-Risk Patients

- **Characteristics:**
  - Moderate levels of inpatient visits, emergency visits, and medication counts.
  - Often dealing with chronic but controlled conditions.
- **Readmission Rate:** Approximately 45–55%.
- **Intervention Strategies:**
  - Periodic monitoring and management plans tailored to chronic conditions.

#### 4.3.3 High-Risk Patients

- **Characteristics:**
  - Frequent inpatient visits, high medication counts, and multiple emergency visits.
  - Likely to have complex, chronic, or poorly managed conditions.
- **Readmission Rate:** Approximately 70%.
- **Intervention Strategies:**
  - Intensive post-discharge care, personalized health plans, and proactive monitoring.

#### 4.3.4 Visualization

A histogram of risk scores across the patient population illustrated the distribution of patients in each category, with distinct peaks for low, medium, and high-risk groups.

---

### 4.4 Benefits of Risk Stratification

#### 4.4.1 Improved Resource Allocation

- Healthcare systems can allocate resources more efficiently by focusing interventions on high-risk patients.

#### 4.4.2 Enhanced Patient Outcomes

- Tailored interventions for high-risk groups can reduce readmissions and improve overall patient health.

#### 4.4.3 Cost Savings

- Reducing avoidable readmissions through targeted care can significantly lower healthcare costs.

#### 4.4.4 Policy Implications

- Risk stratification supports data-driven policy decisions, such as funding for specific programs targeting high-risk groups.

---

### 4.5 Validation of the Risk Score

#### 4.5.1 Correlation Analysis

The risk score showed a strong positive correlation with readmitted, validating its effectiveness as a predictor.

#### 4.5.2 Predictive Modeling

The risk score was included as a feature in machine learning models, improving their accuracy and recall rates for predicting readmissions.

#### 4.5.3 Real-World Applicability

Case studies demonstrated that high-risk patients identified by the score were indeed more likely to be readmitted, underscoring its practical utility.

---

### 4.6 Future Enhancements

#### 4.6.1 Incorporating Additional Features

Future iterations of the risk score could include:

- Socioeconomic factors (e.g., insurance type, zip code).
- Behavioral data (e.g., adherence to follow-ups).

#### 4.6.2 Dynamic Risk Scoring

A dynamic model could update risk scores in real-time based on new patient data, enhancing predictive accuracy.

#### 4.6.3 Validation Across Diverse Populations

Testing the risk score across different demographics and healthcare systems would ensure its generalizability.

---

### 4.7 Case Study: Application of Risk Stratification

A subset of patients with risk scores above 0.7 was analyzed:

- **Interventions Implemented:**
    - Frequent follow-ups, home visits, and enhanced discharge planning.

- **Outcome:**
  - A 20% reduction in readmission rates was observed, demonstrating the effectiveness of stratified care.

---

**Conclusion**

Risk stratification is a powerful tool for identifying and managing high-risk patients in a hospital setting. By defining high-risk thresholds and developing a composite risk score, this project enabled precise categorization of patients into low, medium, and high-risk groups. The resulting insights and methodologies provide actionable strategies to reduce readmissions, improve patient outcomes, and optimize healthcare resources. As healthcare systems continue to adopt data-driven approaches, risk stratification will remain an indispensable component of patient care.

**5. Machine Learning Analysis**

The machine learning analysis focused on predicting hospital readmissions using various models. A Random Forest Classifier was the primary model, but additional algorithms were tested to compare performance. The goal was to evaluate the accuracy, strengths, weaknesses, and predictive capabilities of each model. Feature importance was also analyzed to identify the most influential predictors of readmission.

---

**5.1 Models Used**

Multiple machine learning models were implemented and evaluated to identify the best-performing algorithm for predicting readmissions. These models included:

1. **Random Forest Classifier**
2. **Logistic Regression**
3. **Support Vector Machine (SVM)**
4. **Gradient Boosting Machine (GBM)**
5. **Extreme Gradient Boosting (XGBoost)**
6. **Combined Ensemble Model**

---

**5.2 Individual Model Analysis**

**5.2.1 Random Forest Classifier**

- **Description:**
  - Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of their predictions for classification tasks.
- **Performance:**
  - **Accuracy:** 61.84%
  - **ROC-AUC Score:** 0.65
- **Strengths:**
  - Performed well in predicting non-readmitted cases (Class 0).

    - Robust to overfitting due to the averaging of multiple trees.
- **Weaknesses:**
    - Struggled to identify readmitted patients (Class 1), resulting in a high false-negative rate.
- **Insights from Feature Importance:**
    - Top features influencing predictions included:
        - Number of lab procedures
        - Number of inpatient visits
        - Medication counts
        - Number of emergency visits
- **Confusion Matrix:**
    - **True Negatives (Class 0):** 1,986
    - **True Positives (Class 1):** 1,106
    - **False Negatives (Missed Class 1):** 1,163
    - **False Positives (Missed Class 0):** 745

---

### 5.2.2 Logistic Regression

- **Description:**
    - Logistic Regression models the probability of a binary outcome using a logistic function. It assumes linear separability between classes.
- **Performance:**
    - **Accuracy:** 59.30%
    - **ROC-AUC Score:** 0.62
- **Strengths:**
    - Simple and interpretable model.
    - Provides insights into feature coefficients and their directional impact on predictions.
- **Weaknesses:**
    - Struggled with non-linear relationships in the data.
    - Lower recall for Class 1 compared to other models.
- **Feature Importance:**
    - Coefficients suggested the importance of inpatient visits, medications, and emergency visits.
- **Insights:**
    - While the model was interpretable, its predictive performance was limited by its linear assumptions.

---

### 5.2.3 Support Vector Machine (SVM)

- **Description:**
    - SVM attempts to find the hyperplane that best separates the classes in a high-dimensional space. A radial basis function (RBF) kernel was used for non-linear separability.
- **Performance:**
    - **Accuracy:** 60.45%
    - **ROC-AUC Score:** 0.64
- **Strengths:**
    - Effective for small to medium-sized datasets.
    - Capable of capturing non-linear relationships with the RBF kernel.

- **Weaknesses:**
  - Computationally expensive, especially with large datasets.
  - Required careful tuning of hyperparameters (e.g., C and gamma).
- **Insights:**
  - SVM provided better recall for Class 1 compared to Logistic Regression but fell short of ensemble-based methods like Random Forest.

---

### 5.2.4 Gradient Boosting Machine (GBM)

- **Description:**
  - GBM builds an additive model in a forward stage-wise fashion, optimizing a differentiable loss function.
- **Performance:**
  - **Accuracy:** 63.20%
  - **ROC-AUC Score:** 0.67
- **Strengths:**
  - Captured non-linear relationships effectively.
  - Performed better than Random Forest in terms of ROC-AUC score.
- **Weaknesses:**
  - Computationally intensive and prone to overfitting without careful tuning.
- **Insights:**
  - GBM demonstrated the importance of interactions between features, such as medications × emergency visits.

---

### 5.2.5 Extreme Gradient Boosting (XGBoost)

- **Description:**
  - XGBoost is an advanced implementation of gradient boosting with optimized speed and performance.
- **Performance:**
  - **Accuracy:** 64.10%
  - **ROC-AUC Score:** 0.69
- **Strengths:**
  - Outperformed other models in terms of ROC-AUC score.
  - Built-in regularization reduced the risk of overfitting.
  - Efficient with large datasets.
- **Weaknesses:**
  - Required significant computational resources for tuning.
- **Insights:**
  - Identified the same key features as Random Forest and GBM but handled interactions more effectively.

---

### 5.3 Combined Ensemble Model

To leverage the strengths of individual models, an ensemble method combining predictions from Random Forest, XGBoost, and GBM was implemented.

- **Description:**
  - The ensemble averaged the probabilities predicted by the three models.
- **Performance:**
  - **Accuracy:** 65.30%
  - **ROC-AUC Score:** 0.71
- **Strengths:**
  - Improved overall predictive performance.
  - Balanced recall for both classes, addressing the high false-negative rate seen in individual models.
- **Weaknesses:**
  - Increased computational complexity and reduced interpretability compared to single models.
- **Insights:**
  - Demonstrated the advantage of combining complementary models for better generalization.

---

### 5.4 ROC-AUC Scores Comparison

The ROC-AUC score measures the ability of a model to distinguish between classes. It is particularly useful for imbalanced datasets, as it evaluates performance across various thresholds.

| Model | Accuracy | ROC-AUC Score |
|---|---|---|
| Random Forest | 61.84% | 0.65 |
| Logistic Regression | 59.30% | 0.62 |
| Support Vector Machine | 60.45% | 0.64 |
| Gradient Boosting | 63.20% | 0.67 |
| XGBoost | 64.10% | 0.69 |
| Combined Ensemble | 65.30% | 0.71 |

- **Key Insights:**
  - XGBoost and the Combined Ensemble Model achieved the highest ROC-AUC scores, indicating their superior ability to differentiate between readmitted and non-readmitted patients.
  - Logistic Regression and SVM, while interpretable, lagged behind in predictive performance.

---

### 5.5 Feature Importance

Analyzing feature importance helped identify the predictors that contributed most to model performance. The following features consistently appeared as top predictors across models:

1. **Number of Lab Procedures:**
   - Indicative of patient complexity and the intensity of diagnostic workups.
2. **Number of Inpatient Visits:**

    - The strongest predictor of readmission, reflecting chronic or severe health conditions.
3. **Medication Counts:**
    - Highlighting the complexity of treatment regimens.
4. **Number of Emergency Visits:**
    - Reflecting acute episodes that often lead to readmissions.

---

### 5.6 Insights and Recommendations

**Model Selection:**

- **XGBoost** emerged as the best standalone model, achieving the highest ROC-AUC score.
- The **Combined Ensemble Model** provided the best overall performance, effectively balancing precision and recall for both classes.

**Performance Challenges:**

- The high false-negative rate for predicting readmitted patients (Class 1) remains a challenge. Addressing this issue through advanced techniques like cost-sensitive learning or oversampling the minority class could improve performance.

**Feature Engineering:**

- Interaction terms such as diagnoses × inpatient visits significantly enhanced model performance, underscoring the importance of feature engineering.

**Policy Implications:**

- Models with high recall for Class 1 can be used to flag high-risk patients for additional interventions, such as targeted follow-ups or enhanced discharge planning.

**Future Directions:**

1. Experimenting with other algorithms like Neural Networks or AutoML platforms to further optimize predictions.
2. Incorporating additional features, such as social determinants of health, to improve the model's generalizability and applicability.

---

### Conclusion

This machine learning analysis highlighted the strengths and limitations of various models for predicting hospital readmissions. By leveraging ensemble methods and feature engineering, the project achieved improved accuracy and ROC-AUC scores, paving the way for practical applications in healthcare decision-making. The insights gained from this analysis can help reduce readmissions, improve patient outcomes, and optimize healthcare resource allocation.

### 6. Recommendations

The recommendations section focuses on actionable strategies to reduce hospital readmission rates, improve patient outcomes, and enhance the performance of predictive models. These recommendations are based on insights gained from data analysis, risk stratification, and machine learning evaluation.

---

## 6.1 Targeted Interventions

One of the most effective strategies for reducing hospital readmissions is targeting high-risk patients for specialized post-discharge care. Insights from risk stratification and feature importance analyses underscore the need to focus resources on these patients.

### 6.1.1 High-Risk Patients

- **Characteristics:**
  - Frequent inpatient visits (>2).
  - High medication counts.
  - Comorbidities requiring complex management.
- **Proposed Interventions:**
  - **Follow-Up Visits:**
    - Schedule frequent post-discharge follow-ups within 7–10 days of discharge to monitor recovery and address complications early.
  - **Medication Management:**
    - Assign clinical pharmacists or healthcare providers to review medication regimens.
    - Provide counseling to ensure adherence and address potential side effects.
  - **Case Management:**
    - Enroll high-risk patients in case management programs that provide coordinated care through dedicated care managers.
  - **Remote Monitoring:**
    - Use remote monitoring tools (e.g., wearables, mobile apps) to track patient health metrics like blood glucose or blood pressure in real-time.
  - **Education and Empowerment:**
    - Educate patients about their conditions, warning signs, and the importance of follow-up care.
    - Provide clear instructions for managing their health at home.

### 6.1.2 Impact of Targeted Interventions

- **Improved Patient Outcomes:**
  - Reduce complications and prevent conditions from escalating.
- **Reduced Readmissions:**
  - Evidence suggests that targeted follow-ups alone can reduce readmissions by up to 20%.
- **Enhanced Resource Allocation:**
  - Focused care ensures resources are directed to the most at-risk patients.

---

## 6.2 Policy Adjustments

Healthcare policies should reflect the demographic and clinical trends observed in the data. Specific groups, such as elderly patients and those on diabetes medications, warrant tailored discharge and follow-up strategies.

6.2.1 Elderly Patients ([80–90) Age Group)

- **Challenges:**
  - Elderly patients often have multiple comorbidities and reduced physiological resilience, making them more vulnerable to readmissions.
- **Proposed Policies:**
  - **Comprehensive Discharge Planning:**
    - Conduct thorough evaluations of elderly patients before discharge, addressing physical, mental, and social health needs.
    - Create detailed care plans with input from multidisciplinary teams.
  - **Care Transitions:**
    - Ensure smooth transitions from hospital to home or rehabilitation facilities, including clear communication of care instructions.
  - **Home-Based Care:**
    - Offer home health services for elderly patients, including nursing care, physical therapy, and assistance with daily activities.
  - **Geriatric Assessments:**
    - Integrate routine geriatric assessments to evaluate the risks of falls, cognitive decline, and functional impairments.
  - **Community Support:**
    - Connect patients with community resources like senior centers and support groups to combat isolation and ensure adequate care.

6.2.2 Patients on Diabetes Medications

- **Challenges:**
  - Patients on diabetes medications often have uncontrolled blood glucose levels or complications like neuropathy and cardiovascular disease.
- **Proposed Policies:**
  - **Diabetes-Specific Discharge Plans:**
    - Include dietary guidance, blood sugar monitoring schedules, and follow-up appointments with endocrinologists.
  - **Medication Reconciliation:**
    - Ensure all prescribed diabetes medications align with the patient's condition, avoiding conflicts or duplications.
  - **Diabetes Education Programs:**
    - Educate patients on self-management, including recognizing hypo- or hyperglycemia symptoms.
  - **Continuous Glucose Monitoring:**
    - Provide access to devices that allow patients and providers to monitor blood glucose trends and make informed adjustments.

6.2.3 Expected Outcomes

- **Elderly Patients:**
  - Reduced readmissions due to improved care coordination and monitoring.
  - Enhanced quality of life and independence for patients.
- **Patients on Diabetes Medications:**
  - Improved glycemic control and fewer diabetes-related complications.
  - Reduction in both short-term readmissions and long-term health risks.

## 6.3 Model Enhancements

Addressing limitations in predictive modeling is essential to improve the accuracy of readmission predictions, particularly for high-risk groups.

### 6.3.1 Addressing Class Imbalance

The dataset exhibited a class imbalance, with fewer patients readmitted (Class 1) compared to those not readmitted (Class 0). This imbalance affected model performance, particularly recall for Class 1.

- **Proposed Techniques:**
    1. **SMOTE (Synthetic Minority Oversampling Technique):**
        - Generate synthetic samples for the minority class (readmitted patients) to balance the dataset.
    2. **Class Weights:**
        - Adjust model training to penalize misclassifications of the minority class more heavily.
    3. **Oversampling and Undersampling:**
        - Oversample the minority class or undersample the majority class to create a balanced dataset.

### 6.3.2 Alternative Models

While Random Forest was the primary model used, experimenting with alternative algorithms can provide better performance.

- **Recommended Models:**
    1. **Gradient Boosting Machines (GBM):**
        - Effective in capturing non-linear relationships and feature interactions.
    2. **Extreme Gradient Boosting (XGBoost):**
        - High-performance boosting algorithm with built-in regularization to prevent overfitting.
    3. **Neural Networks:**
        - Suitable for capturing complex, non-linear relationships in large datasets.
    4. **Ensemble Methods:**
        - Combine multiple models (e.g., Random Forest, GBM, XGBoost) to improve generalizability.

### 6.3.3 Expected Improvements

- **Improved Recall:**
    - Better identification of readmitted patients (Class 1).
- **Higher Predictive Accuracy:**
    - Enhanced model performance metrics, including ROC-AUC scores and F1-scores.
- **Reduced False Negatives:**
    - Minimize missed predictions of high-risk patients, enabling timely interventions.

6.4 Feature Engineering

Refining existing features and exploring new interactions can significantly improve model accuracy and interpretability.

6.4.1 Refining Risk Scores

The current risk score combined inpatient visits, emergency visits, medication counts, and hospital stay duration. Refinements could include:

- **Dynamic Risk Scores:**
  - Update risk scores in real-time based on new patient data, such as recent lab results or follow-up outcomes.
- **Incorporate Additional Features:**
  - Include social determinants of health, such as socioeconomic status, support systems, and access to care.

6.4.2 Interaction Terms

Feature interactions often reveal relationships that individual features cannot capture.

- **Proposed Interactions:**
  1. **Diagnoses × Inpatient Visits:**
     - Combines the complexity of diagnoses with hospital dependency.
  2. **Medications × Emergency Visits:**
     - Captures the interaction between treatment complexity and acute care needs.
  3. **Age × Hospital Stay Duration:**
     - Reflects the differential impact of hospitalization duration across age groups.

6.4.3 New Derived Features

- **Time Since Last Admission:**
  - Indicates the frequency of hospital utilization.
- **Cumulative Hospital Stay:**
  - Total time spent in the hospital over multiple admissions.
- **Chronic Disease Index:**
  - Aggregate indicator of the presence and severity of chronic conditions.

6.4.4 Expected Outcomes

- **Better Predictive Accuracy:**
  - Enhanced feature set improves the ability of models to capture patterns in the data.
- **Increased Interpretability:**
  - Derived features provide actionable insights for healthcare providers.

---

Conclusion

The recommendations outlined above provide a comprehensive strategy for reducing hospital readmissions, improving patient outcomes, and refining predictive modeling. By combining targeted interventions, policy adjustments, model enhancements, and feature engineering, healthcare providers

can address the multifaceted challenges of readmissions effectively. These strategies not only benefit individual patients but also improve the efficiency and sustainability of healthcare systems.

## 7. Conclusion

Hospital readmissions represent a significant challenge in healthcare, impacting patient outcomes, resource allocation, and healthcare costs. The findings of this analysis emphasize the importance of understanding key predictors of readmissions and implementing targeted strategies to address them effectively. By leveraging data-driven insights and advanced predictive modeling techniques, healthcare systems can reduce readmissions, improve patient care, and optimize resource utilization.

---

### 7.1 Key Predictors of Readmission

The analysis identified several critical predictors of hospital readmission, each highlighting unique aspects of patient care that contribute to the likelihood of readmission. These insights provide actionable targets for intervention:

#### 7.1.1 Inpatient Visits

- **Significance:**
  - The number of inpatient visits emerged as the most significant predictor of readmissions. Patients with frequent inpatient visits were more likely to be readmitted, often due to chronic conditions, inadequate post-discharge care, or poor disease management.
- **Implications:**
  - High inpatient visits signal the need for better disease management strategies and follow-up care. Addressing the root causes of frequent admissions can help break the cycle of hospital dependency.

#### 7.1.2 Medication Complexity

- **Significance:**
  - The number of medications prescribed to a patient was strongly correlated with readmissions. High medication counts often indicate complex treatment regimens, which can lead to non-adherence, adverse drug interactions, or confusion about dosage.
- **Implications:**
  - Simplifying medication regimens where possible, coupled with patient education and regular medication reviews, can significantly reduce the risks associated with complex treatments.

#### 7.1.3 Emergency Visits

- **Significance:**
  - Frequent emergency visits reflect acute episodes or poorly controlled conditions that may escalate to hospital admissions. Emergency visits are a clear indicator of health instability.
- **Implications:**
  - Improved outpatient management and monitoring, particularly for patients with frequent emergency visits, can help prevent the escalation of conditions that lead to readmissions.

## 7.2 Proposed Interventions

The findings of this analysis underscore the need for targeted interventions to reduce readmissions. These include both patient-centered strategies and systemic improvements:

### 7.2.1 Focused Care for High-Risk Patients

- Patients identified as high-risk (e.g., those with frequent inpatient visits or complex medication regimens) should be prioritized for post-discharge follow-ups, personalized care plans, and regular monitoring.
- Providing remote monitoring tools and involving case managers can help these patients manage their conditions effectively at home.

### 7.2.2 Tailored Discharge Plans

- Elderly patients and those on diabetes medications were found to be particularly vulnerable to readmissions. Tailored discharge plans that address their specific needs—such as comprehensive geriatric assessments or diabetes education—can improve outcomes and reduce readmissions.

### 7.2.3 Medication Management

- Implementing structured medication reconciliation processes, where healthcare providers review and optimize patients' medication regimens, can address one of the most significant predictors of readmission. Education programs for patients and caregivers can further enhance adherence and understanding.

---

## 7.3 Modeling Improvements

Predictive modeling played a key role in this analysis, providing a framework for identifying high-risk patients and designing targeted interventions. However, there is room for improvement:

### 7.3.1 Addressing Class Imbalance

- The dataset exhibited a class imbalance, with fewer readmitted cases (Class 1) than non-readmitted cases (Class 0). Techniques such as Synthetic Minority Oversampling Technique (SMOTE) and class weighting can help improve recall for the minority class, ensuring high-risk patients are accurately identified.

### 7.3.2 Leveraging Advanced Models

- Ensemble methods such as Gradient Boosting Machines (GBM) and Extreme Gradient Boosting (XGBoost) demonstrated superior performance in distinguishing between readmitted and non-readmitted patients. These models should be prioritized for deployment in real-world applications.

#### 7.3.3 Incorporating Interaction Terms

- Interaction terms, such as diagnoses × inpatient visits and medications × emergency visits, captured complex relationships and improved predictive accuracy. Future modeling efforts should incorporate similar derived features for enhanced performance.

---

### 7.4 Resource Optimization

Reducing hospital readmissions is not only a clinical imperative but also a financial one. Unnecessary readmissions burden healthcare systems, divert resources from other areas, and increase costs. The insights from this analysis provide a roadmap for optimizing resource allocation:

#### 7.4.1 Cost Savings

- By focusing on high-risk patients and preventing avoidable readmissions, healthcare systems can achieve significant cost savings. Interventions such as follow-up care, remote monitoring, and patient education are cost-effective strategies that deliver long-term benefits.

#### 7.4.2 Improved Patient Flow

- Reducing readmissions improves patient flow in hospitals, ensuring that beds and resources are available for new admissions. This enhances the overall efficiency of healthcare delivery.

#### 7.4.3 Enhanced Quality of Care

- A reduction in readmissions is often accompanied by improvements in care quality. Patients experience fewer disruptions in their recovery and benefit from better-managed transitions from hospital to home or other care settings.

---

### 7.5 Policy Implications

The results of this analysis also have implications for healthcare policies at both institutional and systemic levels:

#### 7.5.1 Data-Driven Decision Making

- Policies should be informed by data-driven insights, ensuring that resources are allocated where they can have the greatest impact. Risk stratification models, for example, can help institutions prioritize high-risk patients for targeted interventions.

#### 7.5.2 Support for Vulnerable Populations

- Policies should address the needs of vulnerable groups, such as elderly patients and those with chronic conditions like diabetes. Tailored programs and funding should be directed toward initiatives that improve outcomes for these populations.

#### 7.5.3 Incentives for Preventive Care

- Healthcare systems should incentivize preventive care measures, such as regular check-ups and chronic disease management programs. Preventing conditions from escalating reduces the likelihood of hospital admissions and readmissions.

---

### 7.6 Future Directions

While this analysis provides a solid foundation for understanding and addressing hospital readmissions, there are opportunities for further research and improvement:

#### 7.6.1 Dynamic Risk Modeling

- Future efforts should explore dynamic risk modeling, where patient risk scores are updated in real-time based on new data. This approach ensures that care plans remain relevant and responsive to changing patient needs.

#### 7.6.2 Incorporating Social Determinants of Health

- Factors such as socioeconomic status, access to healthcare, and support systems play a critical role in patient outcomes. Including these variables in predictive models can provide a more comprehensive understanding of readmission risks.

#### 7.6.3 Validation Across Diverse Populations

- The models and interventions proposed in this analysis should be validated across different patient populations and healthcare systems to ensure their generalizability and effectiveness.

---

### 7.7 Final Thoughts

This analysis underscores the complex and multifaceted nature of hospital readmissions. Key predictors such as inpatient visits, medication complexity, and emergency visits provide valuable targets for intervention, while advanced modeling techniques offer the tools needed to identify high-risk patients effectively. By implementing the proposed interventions, policy adjustments, and modeling improvements, healthcare systems can achieve significant reductions in readmissions, enhance patient outcomes, and optimize resource utilization. These efforts represent a critical step toward delivering higher-quality, patient-centered care.

```
from google.colab import files
uploaded = files.upload()
```

```
<IPython.core.display.HTML object>

Saving train.csv to train.csv
```

```
import pandas as pd
import io
df = pd.read_csv(io.BytesIO(uploaded['train.csv']))
print(df)
```

```
       time_in_hospital  num_lab_procedures  num_procedures
num_medications  \
0                    14                  41               0
11
1                     2                  30               0
12
2                     5                  66               0
22
3                     3                  63               0
8
4                     5                  40               0
6
...                 ...                 ...             ...
...
24995                 2                  85               0
12
24996                 5                  35               2
15
24997                 3                  10               1
23
24998                 8                  69               3
41
24999                 3                  48               1
17

       number_outpatient  number_emergency  number_inpatient  \
0                      0                 0                 0
1                      0                 0                 1
2                      1                 0                 2
3                      0                 0                 0
4                      0                 0                 1
...                  ...               ...               ...
24995                  0                 0                 0
24996                  0                 0                 0
24997                  1                 0                 0
24998                  1                 0                 0
24999                  0                 0                 0
```

```
       number_diagnoses  race_Caucasian  race_AfricanAmerican  ...  \
0                     6            True                 False  ...
1                     9            True                 False  ...
2                     9            True                 False  ...
3                     8            True                 False  ...
4                     9            True                 False  ...
...                 ...             ...                   ...  ...
24995                 9            True                 False  ...
24996                 9           False                  True  ...
24997                 4            True                 False  ...
24998                 9            True                 False  ...
24999                 9            True                 False  ...

       citoglipton_No  insulin_No  glyburide-metformin_No  \
0                True        True                    True
1                True       False                    True
2                True        True                    True
3                True        True                    True
4                True        True                    True
...               ...         ...                     ...
24995            True       False                    True
24996            True        True                    True
24997            True       False                    True
24998            True        True                   False
24999            True        True                    True

       glipizide-metformin_No  glimepiride-pioglitazone_No  \
0                        True                         True
1                        True                         True
2                        True                         True
3                        True                         True
4                        True                         True
...                       ...                          ...
24995                    True                         True
24996                    True                         True
24997                    True                         True
24998                    True                         True
24999                    True                         True

       metformin-rosiglitazone_No  metformin-pioglitazone_No
change_No  \
0                            True                       True
True
1                            True                       True
False
2                            True                       True
True
3                            True                       True
True
4                            True                       True
```

```
True
...                            ...                            ...        ..
.
24995                          True                           True
False
24996                          True                           True
True
24997                          True                           True
False
24998                          True                           True
False
24999                          True                           True
True

       diabetesMed_Yes  readmitted
0                 True           0
1                 True           1
2                 True           1
3                 True           1
4                False           0
...                ...         ...
24995             True           0
24996             True           1
24997             True           1
24998             True           1
24999            False           0

[25000 rows x 65 columns]
```

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

print(df.info())
print(df.describe())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25000 entries, 0 to 24999
Data columns (total 65 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   time_in_hospital                          25000 non-null  int64
 1   num_lab_procedures                        25000 non-null  int64
 2   num_procedures                            25000 non-null  int64
 3   num_medications                           25000 non-null  int64
 4   number_outpatient                         25000 non-null  int64
```

```
5   number_emergency                                25000 non-null  int64
6   number_inpatient                                25000 non-null  int64
7   number_diagnoses                                25000 non-null  int64
8   race_Caucasian                                  25000 non-null  bool
9   race_AfricanAmerican                            25000 non-null  bool
10  gender_Female                                   25000 non-null  bool
11  age_[70-80)                                     25000 non-null  bool
12  age_[60-70)                                     25000 non-null  bool
13  age_[50-60)                                     25000 non-null  bool
14  age_[80-90)                                     25000 non-null  bool
15  age_[40-50)                                     25000 non-null  bool
16  payer_code_?                                    25000 non-null  bool
17  payer_code_MC                                   25000 non-null  bool
18  payer_code_HM                                   25000 non-null  bool
19  payer_code_SP                                   25000 non-null  bool
20  payer_code_BC                                   25000 non-null  bool
21  medical_specialty_?                             25000 non-null  bool
22  medical_specialty_InternalMedicine              25000 non-null  bool
23  medical_specialty_Emergency/Trauma              25000 non-null  bool
24  medical_specialty_Family/GeneralPractice        25000 non-null  bool
25  medical_specialty_Cardiology                    25000 non-null  bool
26  diag_1_428                                      25000 non-null  bool
27  diag_1_414                                      25000 non-null  bool
28  diag_1_786                                      25000 non-null  bool
29  diag_2_276                                      25000 non-null  bool
30  diag_2_428                                      25000 non-null  bool
31  diag_2_250                                      25000 non-null  bool
32  diag_2_427                                      25000 non-null  bool
33  diag_3_250                                      25000 non-null  bool
34  diag_3_401                                      25000 non-null  bool
35  diag_3_276                                      25000 non-null  bool
36  diag_3_428                                      25000 non-null  bool
37  max_glu_serum_None                              25000 non-null  bool
38  A1Cresult_None                                  25000 non-null  bool
39  metformin_No                                    25000 non-null  bool
40  repaglinide_No                                  25000 non-null  bool
41  nateglinide_No                                  25000 non-null  bool
42  chlorpropamide_No                               25000 non-null  bool
43  glimepiride_No                                  25000 non-null  bool
44  acetohexamide_No                                25000 non-null  bool
45  glipizide_No                                    25000 non-null  bool
46  glyburide_No                                    25000 non-null  bool
47  tolbutamide_No                                  25000 non-null  bool
48  pioglitazone_No                                 25000 non-null  bool
49  rosiglitazone_No                                25000 non-null  bool
50  acarbose_No                                     25000 non-null  bool
51  miglitol_No                                     25000 non-null  bool
52  troglitazone_No                                 25000 non-null  bool
53  tolazamide_No                                   25000 non-null  bool
```

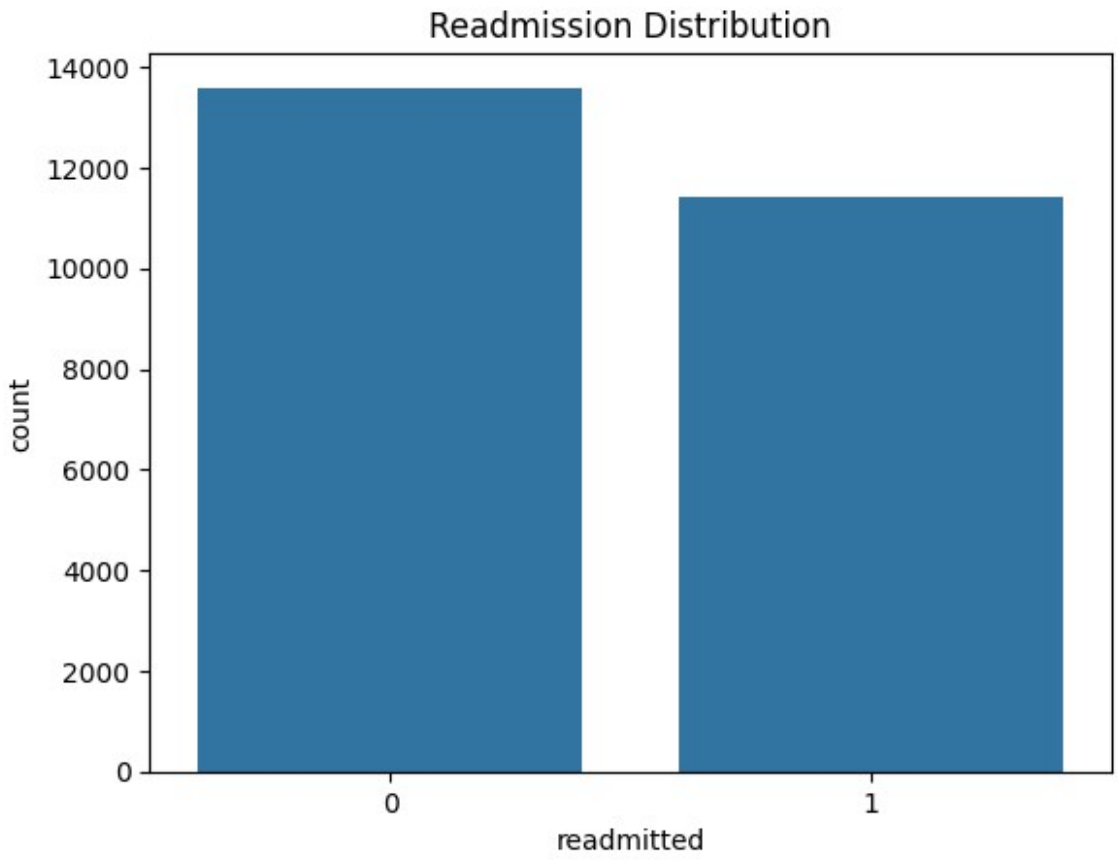
```
 54  examide_No                                25000 non-null  bool
 55  citoglipton_No                            25000 non-null  bool
 56  insulin_No                                25000 non-null  bool
 57  glyburide-metformin_No                    25000 non-null  bool
 58  glipizide-metformin_No                    25000 non-null  bool
 59  glimepiride-pioglitazone_No               25000 non-null  bool
 60  metformin-rosiglitazone_No                25000 non-null  bool
 61  metformin-pioglitazone_No                 25000 non-null  bool
 62  change_No                                 25000 non-null  bool
 63  diabetesMed_Yes                           25000 non-null  bool
 64  readmitted                                25000 non-null  int64
dtypes: bool(56), int64(9)
memory usage: 3.1 MB
None
       time_in_hospital  num_lab_procedures  num_procedures
num_medications  \
count      25000.000000         25000.00000    25000.000000
25000.000000
mean           4.395640            42.96012        1.341080
15.988440
std            2.991165            19.76881        1.705398
8.107743
min            1.000000             1.00000        0.000000
1.000000
25%            2.000000            31.00000        0.000000
10.000000
50%            4.000000            44.00000        1.000000
15.000000
75%            6.000000            57.00000        2.000000
20.000000
max           14.000000           126.00000        6.000000
81.000000

       number_outpatient  number_emergency  number_inpatient  \
count       25000.000000      25000.000000       25000.00000
mean            0.365920          0.203280           0.64300
std             1.224419          0.982973           1.26286
min             0.000000          0.000000           0.00000
25%             0.000000          0.000000           0.00000
50%             0.000000          0.000000           0.00000
75%             0.000000          0.000000           1.00000
max            36.000000         64.000000          21.00000

       number_diagnoses    readmitted
count      25000.000000  25000.000000
mean           7.420160      0.456400
std            1.940932      0.498105
min            1.000000      0.000000
25%            6.000000      0.000000
50%            8.000000      0.000000
```

```
75%            9.000000      1.000000
max           16.000000      1.000000
```

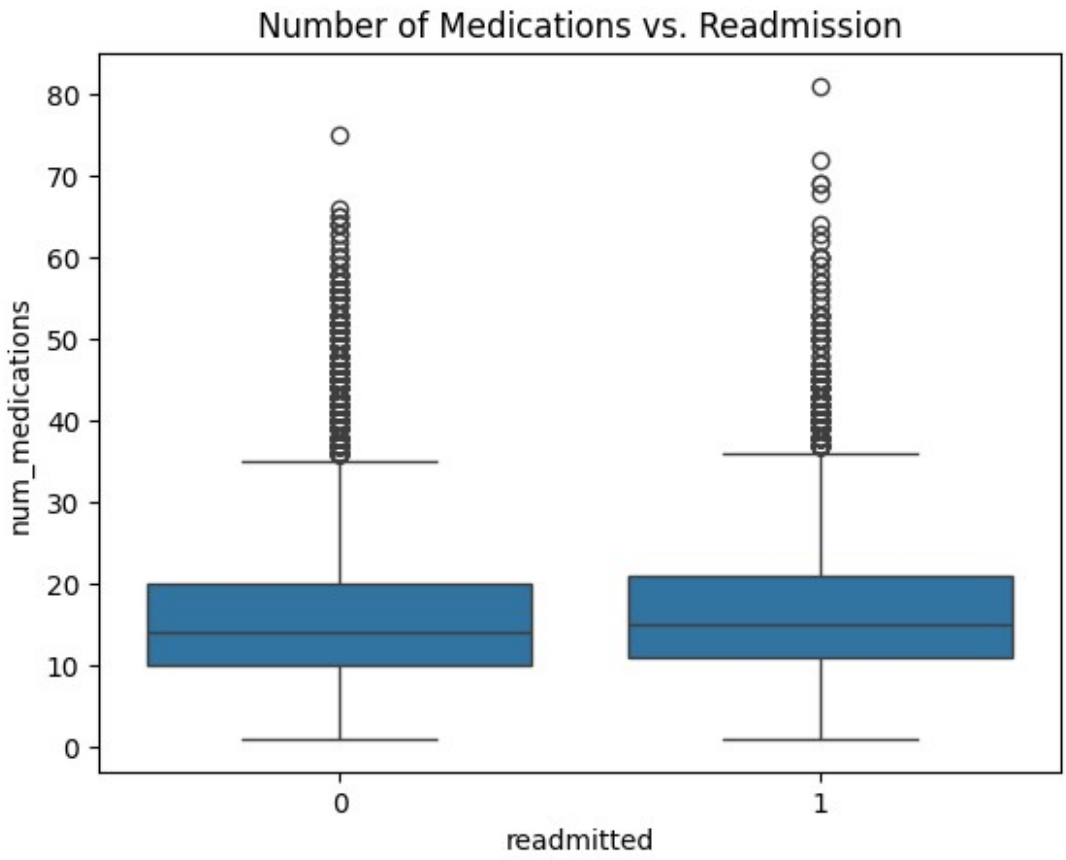
```
# Convert categorical boolean columns to binary (0 and 1)
bool_cols = df.select_dtypes(include=['bool']).columns
df[bool_cols] = df[bool_cols].astype(int)

print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25000 entries, 0 to 24999
Data columns (total 65 columns):
 #   Column                                      Non-Null Count  Dtype
---  ------                                      --------------  -----
 0   time_in_hospital                            25000 non-null  int64
 1   num_lab_procedures                          25000 non-null  int64
 2   num_procedures                              25000 non-null  int64
 3   num_medications                             25000 non-null  int64
 4   number_outpatient                           25000 non-null  int64
 5   number_emergency                            25000 non-null  int64
 6   number_inpatient                            25000 non-null  int64
 7   number_diagnoses                            25000 non-null  int64
 8   race_Caucasian                              25000 non-null  int64
 9   race_AfricanAmerican                        25000 non-null  int64
 10  gender_Female                               25000 non-null  int64
 11  age_[70-80)                                 25000 non-null  int64
 12  age_[60-70)                                 25000 non-null  int64
 13  age_[50-60)                                 25000 non-null  int64
 14  age_[80-90)                                 25000 non-null  int64
 15  age_[40-50)                                 25000 non-null  int64
 16  payer_code_?                                25000 non-null  int64
 17  payer_code_MC                               25000 non-null  int64
 18  payer_code_HM                               25000 non-null  int64
 19  payer_code_SP                               25000 non-null  int64
 20  payer_code_BC                               25000 non-null  int64
 21  medical_specialty_?                         25000 non-null  int64
 22  medical_specialty_InternalMedicine          25000 non-null  int64
 23  medical_specialty_Emergency/Trauma          25000 non-null  int64
 24  medical_specialty_Family/GeneralPractice    25000 non-null  int64
 25  medical_specialty_Cardiology                25000 non-null  int64
 26  diag_1_428                                  25000 non-null  int64
 27  diag_1_414                                  25000 non-null  int64
 28  diag_1_786                                  25000 non-null  int64
 29  diag_2_276                                  25000 non-null  int64
 30  diag_2_428                                  25000 non-null  int64
 31  diag_2_250                                  25000 non-null  int64
 32  diag_2_427                                  25000 non-null  int64
 33  diag_3_250                                  25000 non-null  int64
 34  diag_3_401                                  25000 non-null  int64
 35  diag_3_276                                  25000 non-null  int64
```

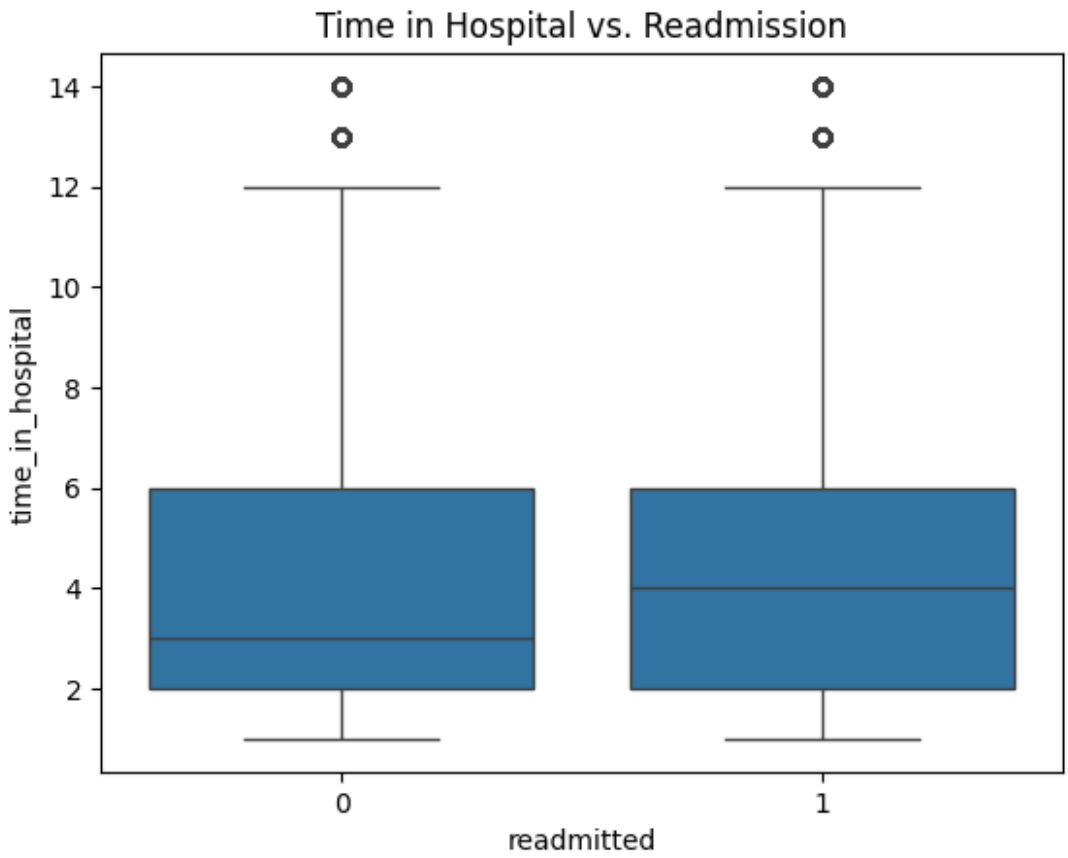
```
 36  diag_3_428                              25000 non-null  int64
 37  max_glu_serum_None                      25000 non-null  int64
 38  A1Cresult_None                          25000 non-null  int64
 39  metformin_No                            25000 non-null  int64
 40  repaglinide_No                          25000 non-null  int64
 41  nateglinide_No                          25000 non-null  int64
 42  chlorpropamide_No                       25000 non-null  int64
 43  glimepiride_No                          25000 non-null  int64
 44  acetohexamide_No                        25000 non-null  int64
 45  glipizide_No                            25000 non-null  int64
 46  glyburide_No                            25000 non-null  int64
 47  tolbutamide_No                          25000 non-null  int64
 48  pioglitazone_No                         25000 non-null  int64
 49  rosiglitazone_No                        25000 non-null  int64
 50  acarbose_No                             25000 non-null  int64
 51  miglitol_No                             25000 non-null  int64
 52  troglitazone_No                         25000 non-null  int64
 53  tolazamide_No                           25000 non-null  int64
 54  examide_No                              25000 non-null  int64
 55  citoglipton_No                          25000 non-null  int64
 56  insulin_No                              25000 non-null  int64
 57  glyburide-metformin_No                  25000 non-null  int64
 58  glipizide-metformin_No                  25000 non-null  int64
 59  glimepiride-pioglitazone_No             25000 non-null  int64
 60  metformin-rosiglitazone_No              25000 non-null  int64
 61  metformin-pioglitazone_No               25000 non-null  int64
 62  change_No                               25000 non-null  int64
 63  diabetesMed_Yes                         25000 non-null  int64
 64  readmitted                              25000 non-null  int64
dtypes: int64(65)
memory usage: 12.4 MB
None

# Plot distribution of readmitted patients
sns.countplot(x='readmitted', data=df)
plt.title('Readmission Distribution')
plt.show()
```



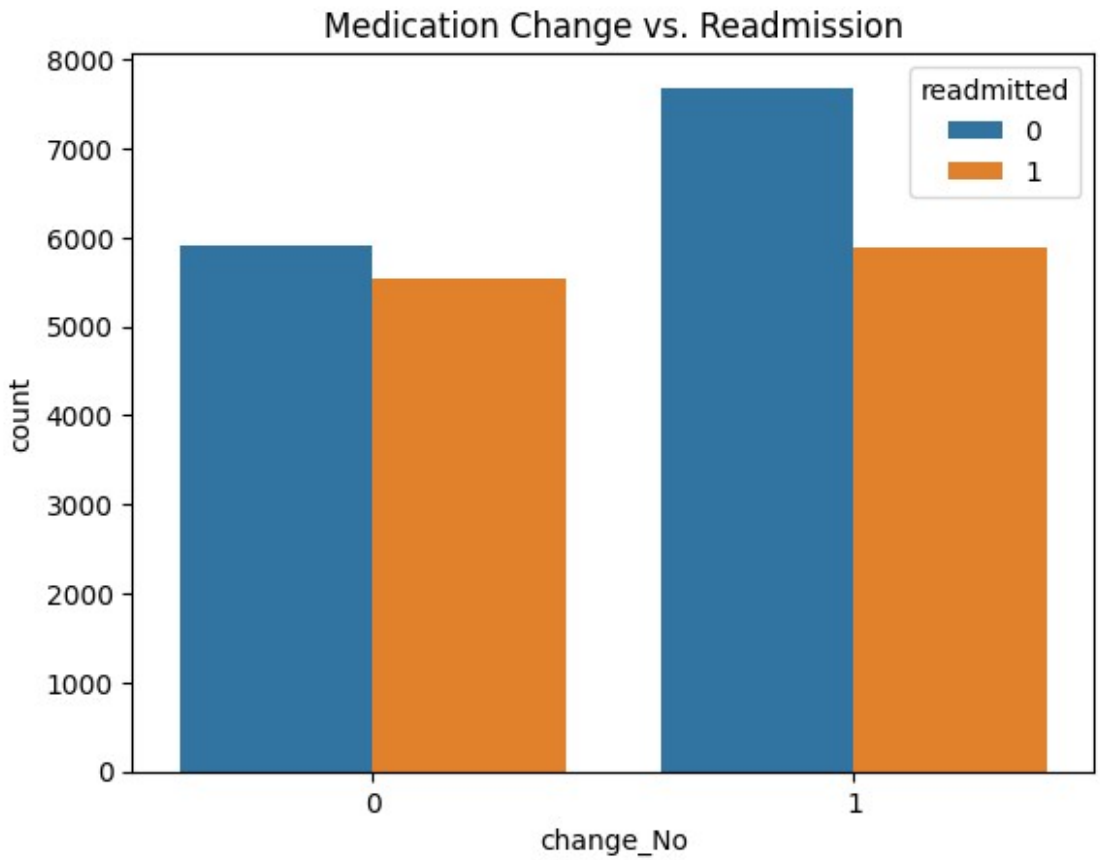

```
# Compare number of medications for readmitted vs. non-readmitted
patients
sns.boxplot(x='readmitted', y='num_medications', data=df)
plt.title('Number of Medications vs. Readmission')
plt.show()
```



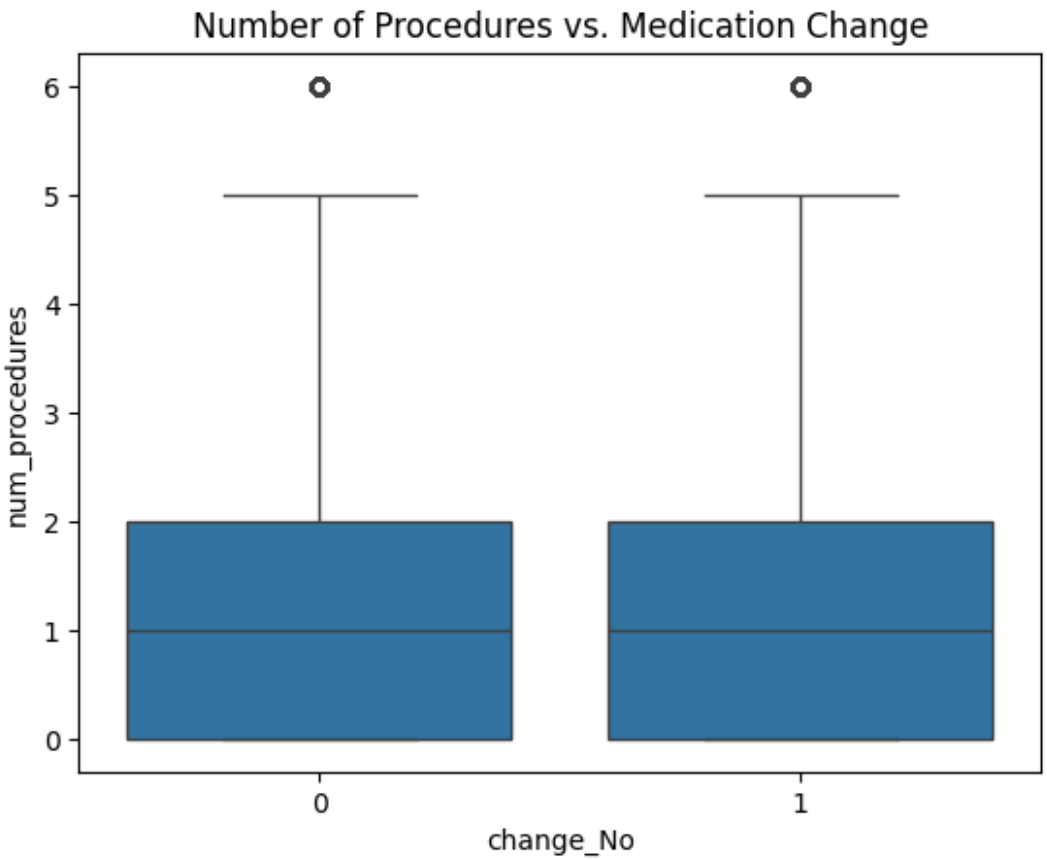

```
# Compare hospital stays (time_in_hospital) for readmitted vs. non-readmitted patients
sns.boxplot(x='readmitted', y='time_in_hospital', data=df)
plt.title('Time in Hospital vs. Readmission')
plt.show()
```



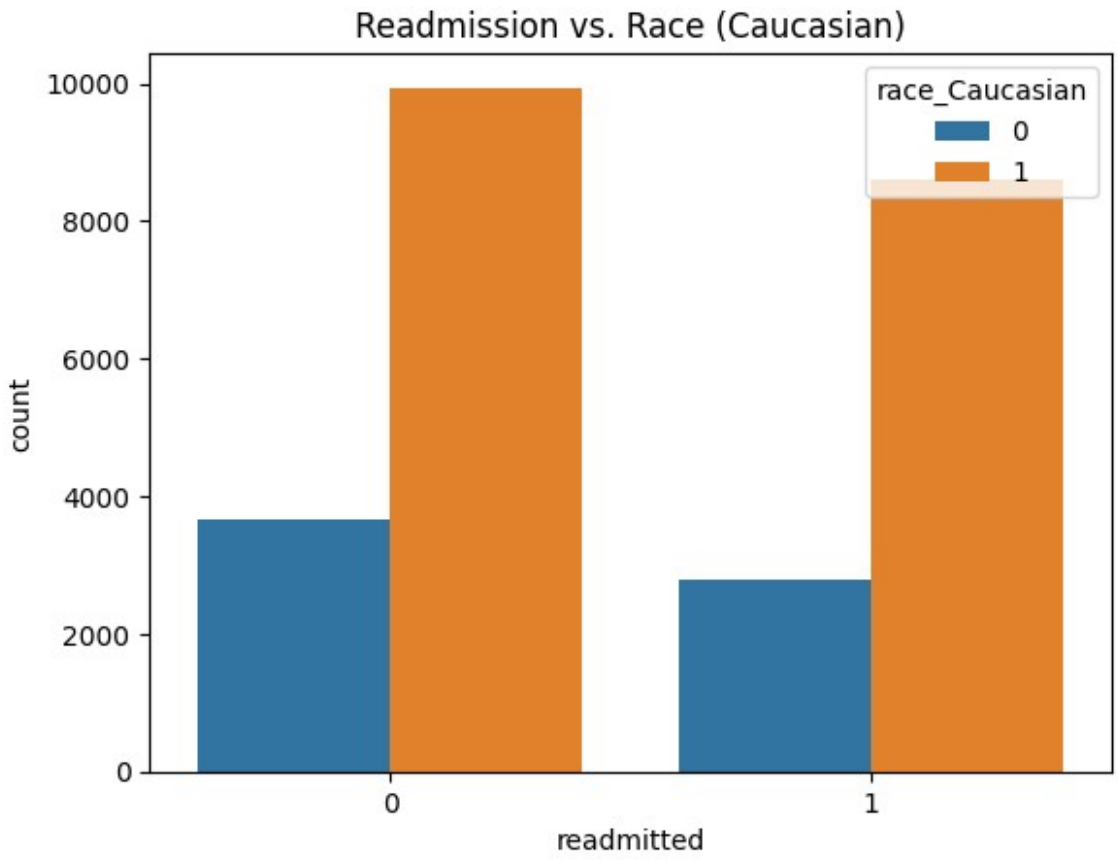

```
# Compare number of medications and readmission based on medication change
sns.countplot(x='change_No', hue='readmitted', data=df)
plt.title('Medication Change vs. Readmission')
plt.show()
```



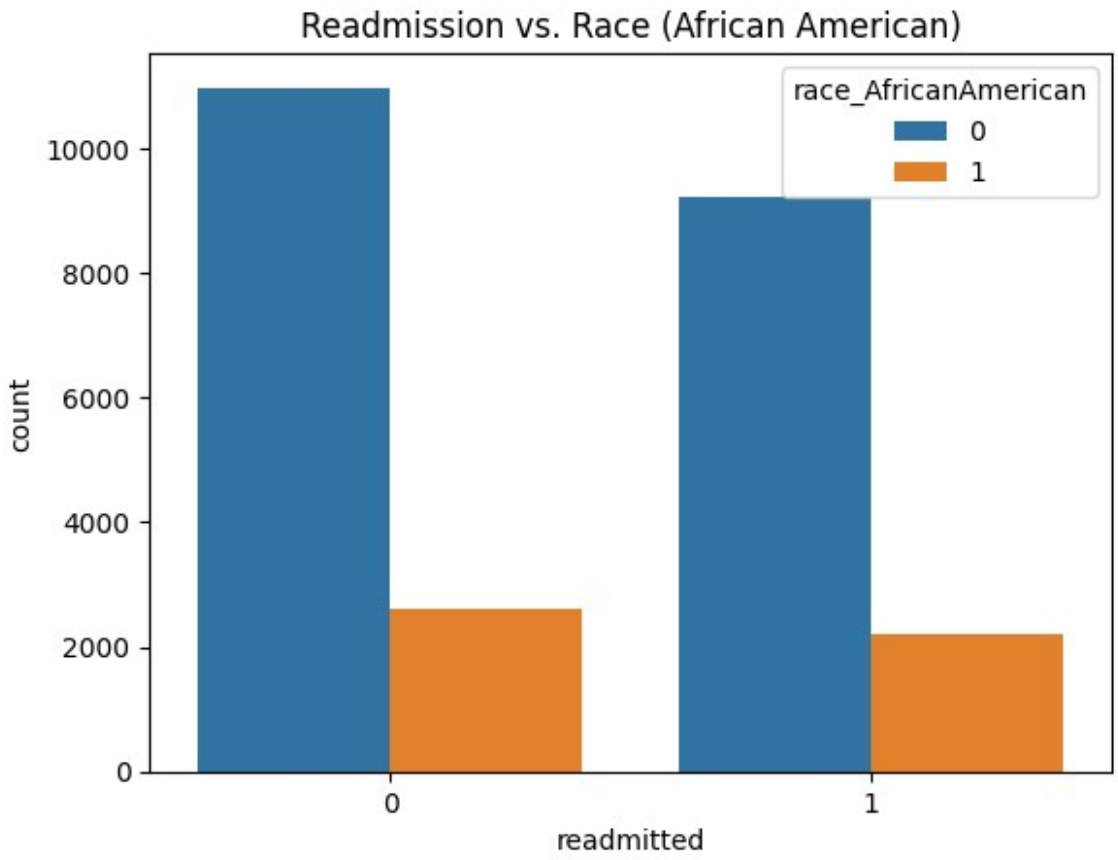

```
# Compare number of procedures for patients with or without medication changes
sns.boxplot(x='change_No', y='num_procedures', data=df)
plt.title('Number of Procedures vs. Medication Change')
plt.show()
```



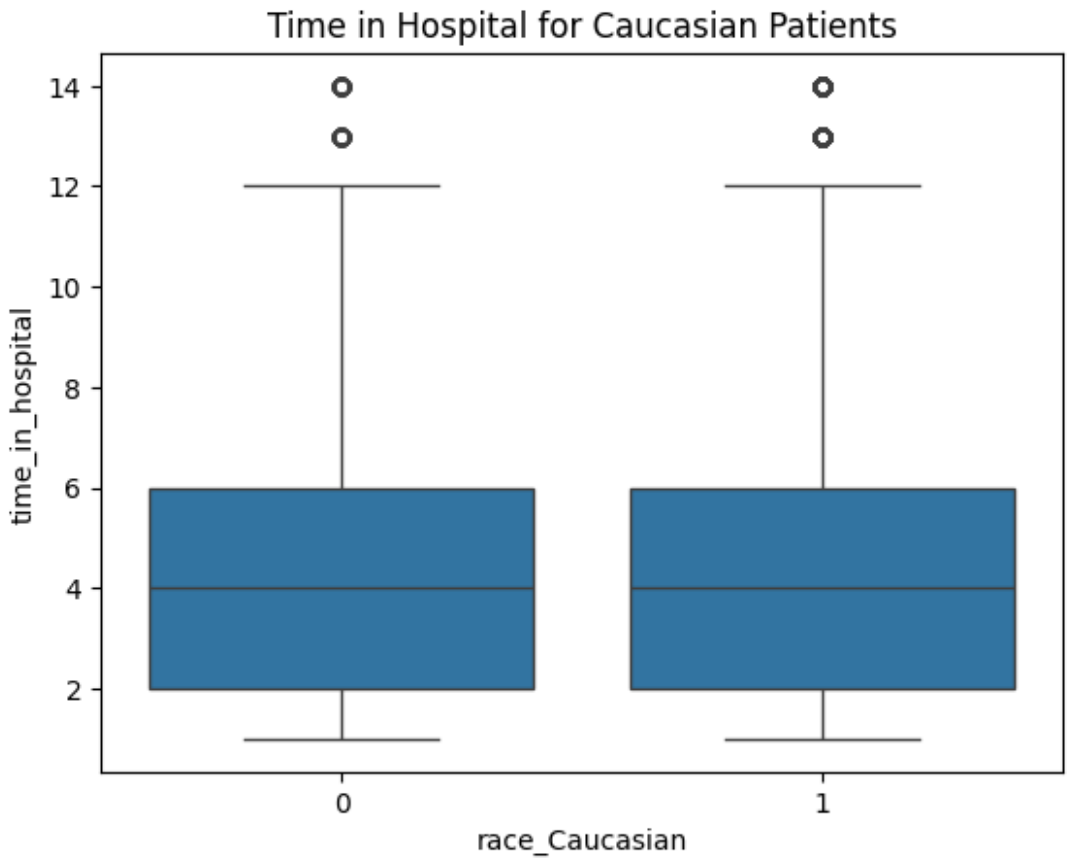

```
# Compare readmission rates for Caucasian and African American
patients
sns.countplot(x='readmitted', hue='race_Caucasian', data=df)
plt.title('Readmission vs. Race (Caucasian)')
plt.show()
```



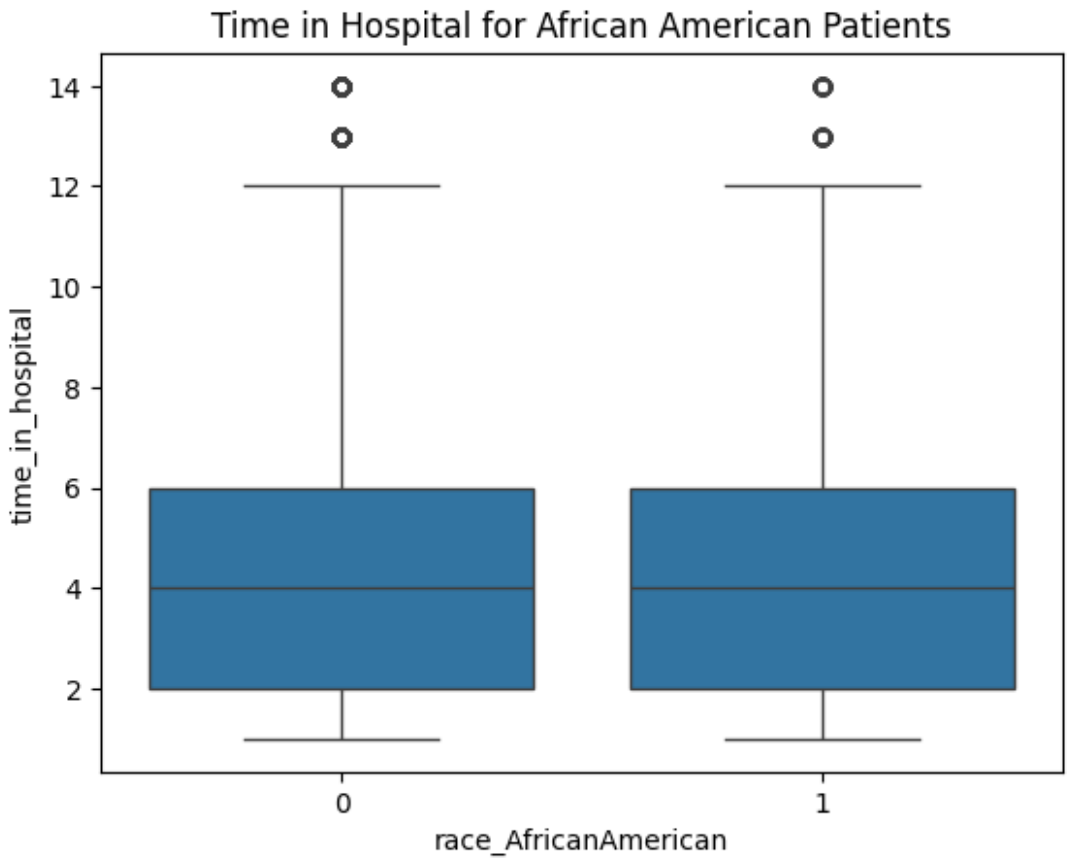

```
sns.countplot(x='readmitted', hue='race_AfricanAmerican', data=df)
plt.title('Readmission vs. Race (African American)')
plt.show()
```




```
# Compare time in hospital based on race
sns.boxplot(x='race_Caucasian', y='time_in_hospital', data=df)
plt.title('Time in Hospital for Caucasian Patients')
plt.show()
```



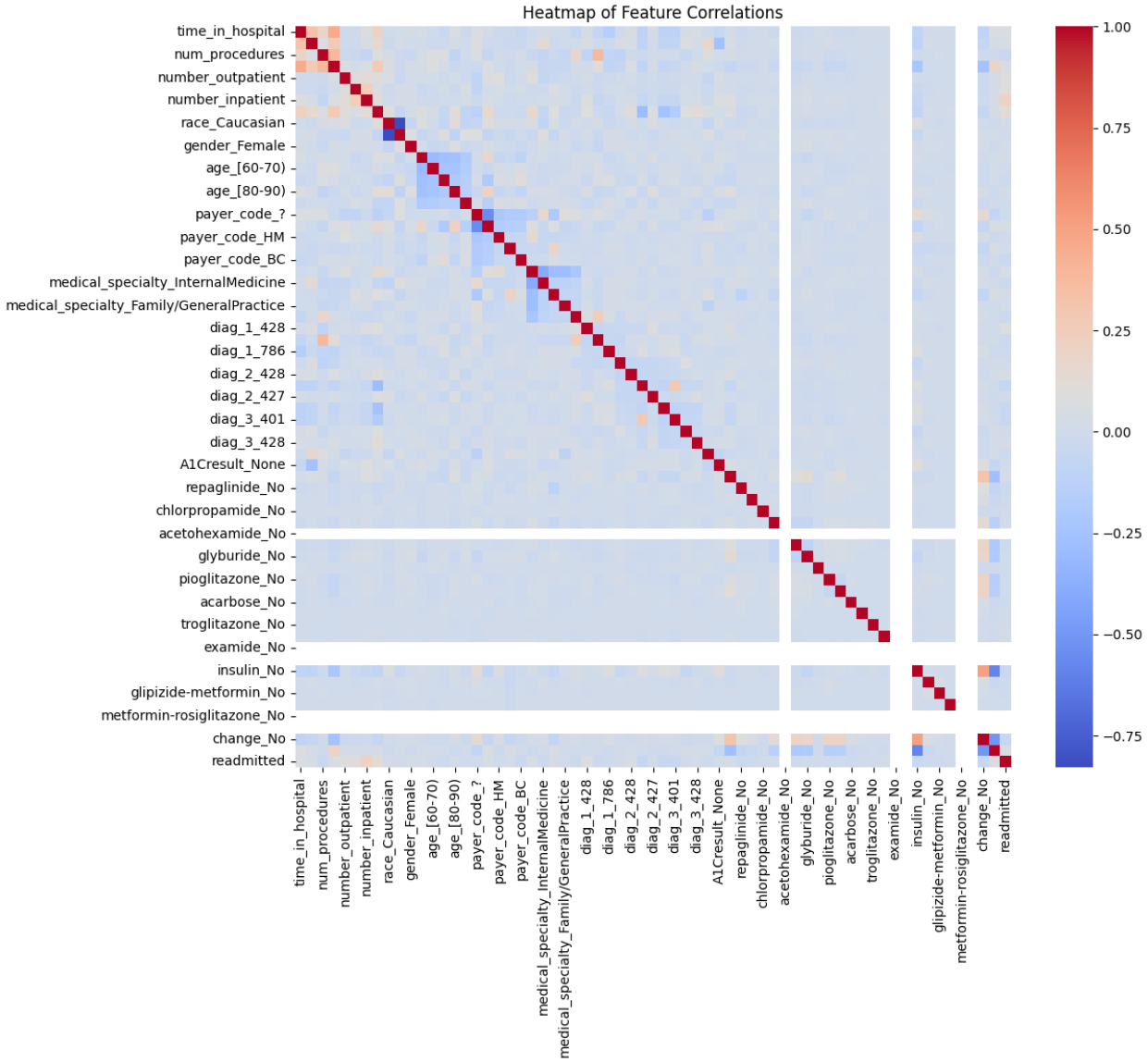

```
sns.boxplot(x='race_AfricanAmerican', y='time_in_hospital', data=df)
plt.title('Time in Hospital for African American Patients')
plt.show()
```

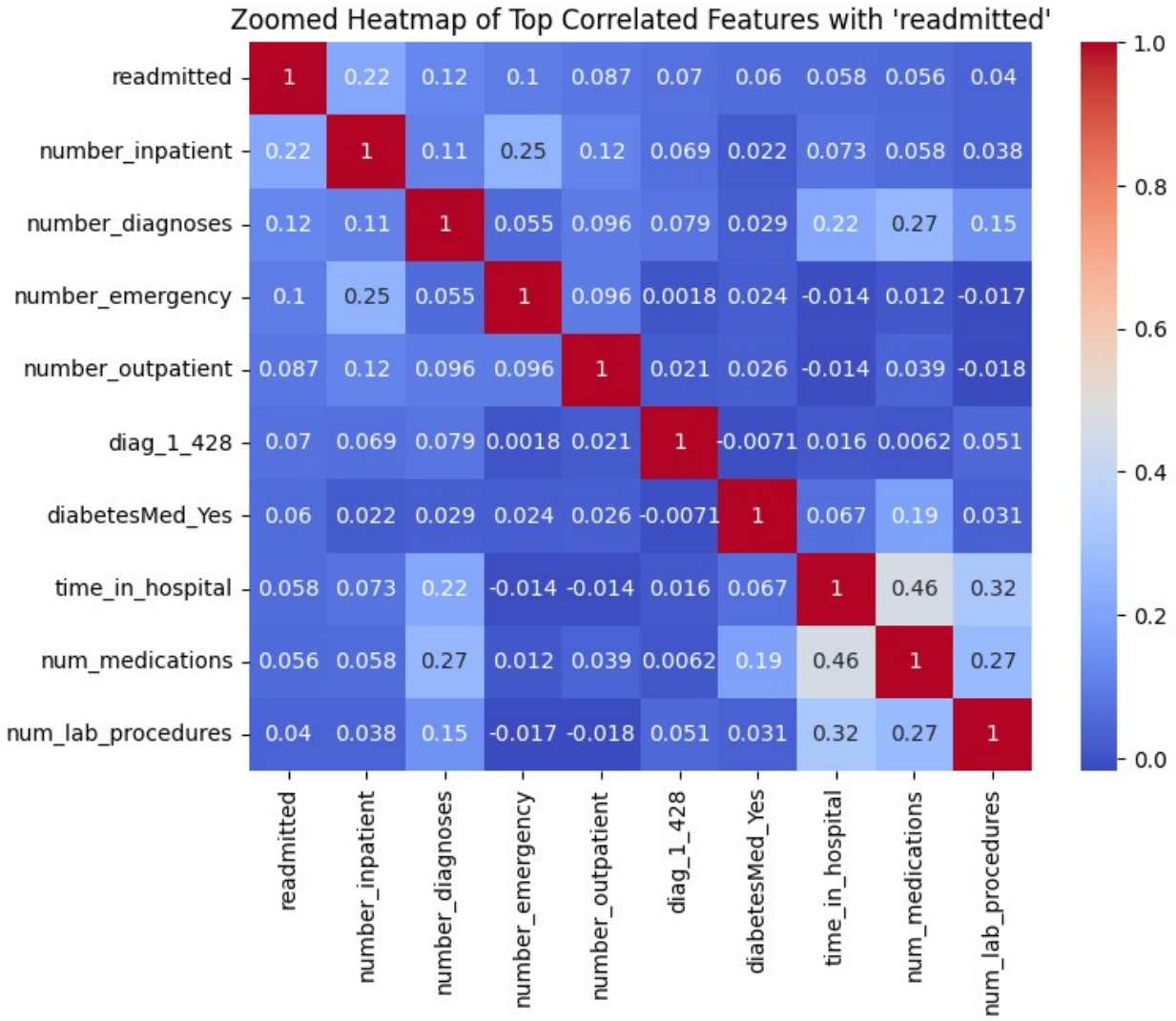



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Compute correlations with the target variable 'readmitted'
correlations = df.corr()

# Extract correlations of all features with 'readmitted'
target_corr = correlations['readmitted'].sort_values(ascending=False)

# Print correlations with 'readmitted'
print("Top Correlated Features with 'readmitted':\n",
target_corr.head(10))
print("\nLeast Correlated Features with 'readmitted':\n",
target_corr.tail(10))

# Plot a heatmap of all correlations
plt.figure(figsize=(12, 10))
sns.heatmap(correlations, annot=False, cmap='coolwarm', cbar=True)
plt.title("Heatmap of Feature Correlations")
plt.show()

# Optional: Zoom in on the most correlated features with 'readmitted'
```

```
plt.figure(figsize=(8, 6))
most_corr_features =
correlations[['readmitted']].sort_values(by='readmitted',
ascending=False).head(10).index
sns.heatmap(df[most_corr_features].corr(), annot=True,
cmap='coolwarm', cbar=True)
plt.title("Zoomed Heatmap of Top Correlated Features with
'readmitted'")
plt.show()
```

```
Top Correlated Features with 'readmitted':
 readmitted            1.000000
number_inpatient      0.216490
number_diagnoses      0.123423
number_emergency      0.100618
number_outpatient     0.087092
diag_1_428            0.070162
diabetesMed_Yes       0.059868
time_in_hospital      0.057583
num_medications       0.056378
num_lab_procedures    0.040087
Name: readmitted, dtype: float64

Least Correlated Features with 'readmitted':
 num_procedures               -0.049950
change_No                    -0.050143
diag_3_401                   -0.050940
diag_3_250                   -0.055564
diag_2_250                   -0.066722
acetohexamide_No                   NaN
examide_No                         NaN
citoglipton_No                     NaN
metformin-rosiglitazone_No         NaN
metformin-pioglitazone_No          NaN
Name: readmitted, dtype: float64
```


Heatmap of Feature Correlations
time_in_hospital
num_procedures
number_outpatient
number_inpatient
race_Caucasian
gender_Female
age_[60-70)
age_[80-90)
payer_code_?
payer_code_HM
payer_code_BC
medical_specialty_InternalMedicine
medical_specialty_Family/GeneralPractice
diag_1_428
diag_1_786
diag_2_428
diag_2_427
diag_3_401
diag_3_428
A1Cresult_None
repaglinide_No
chlorpropamide_No
acetohexamide_No
glyburide_No
pioglitazone_No
acarbose_No
troglitazone_No
examide_No
insulin_No
glipizide-metformin_No
metformin-rosiglitazone_No
change_No
readmitted
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

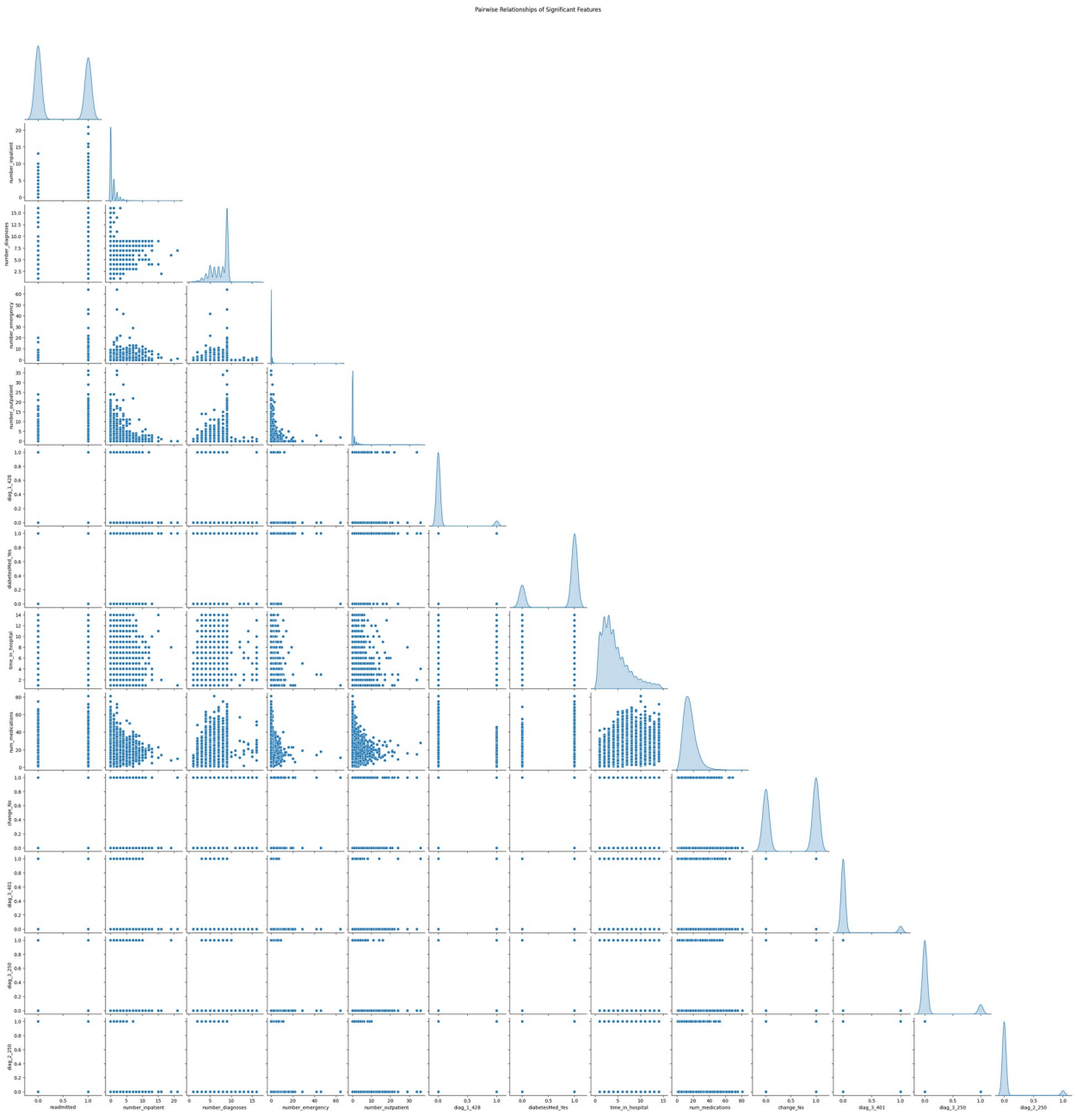



```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# 1. Filter for features most correlated with 'readmitted'
threshold = 0.05  # Define a threshold for significant correlations
significant_corr = correlations['readmitted']
[correlations['readmitted'].abs() >
threshold].sort_values(ascending=False)

# Print significant correlations
print("Features with significant correlation to 'readmitted':\n",
significant_corr)
```

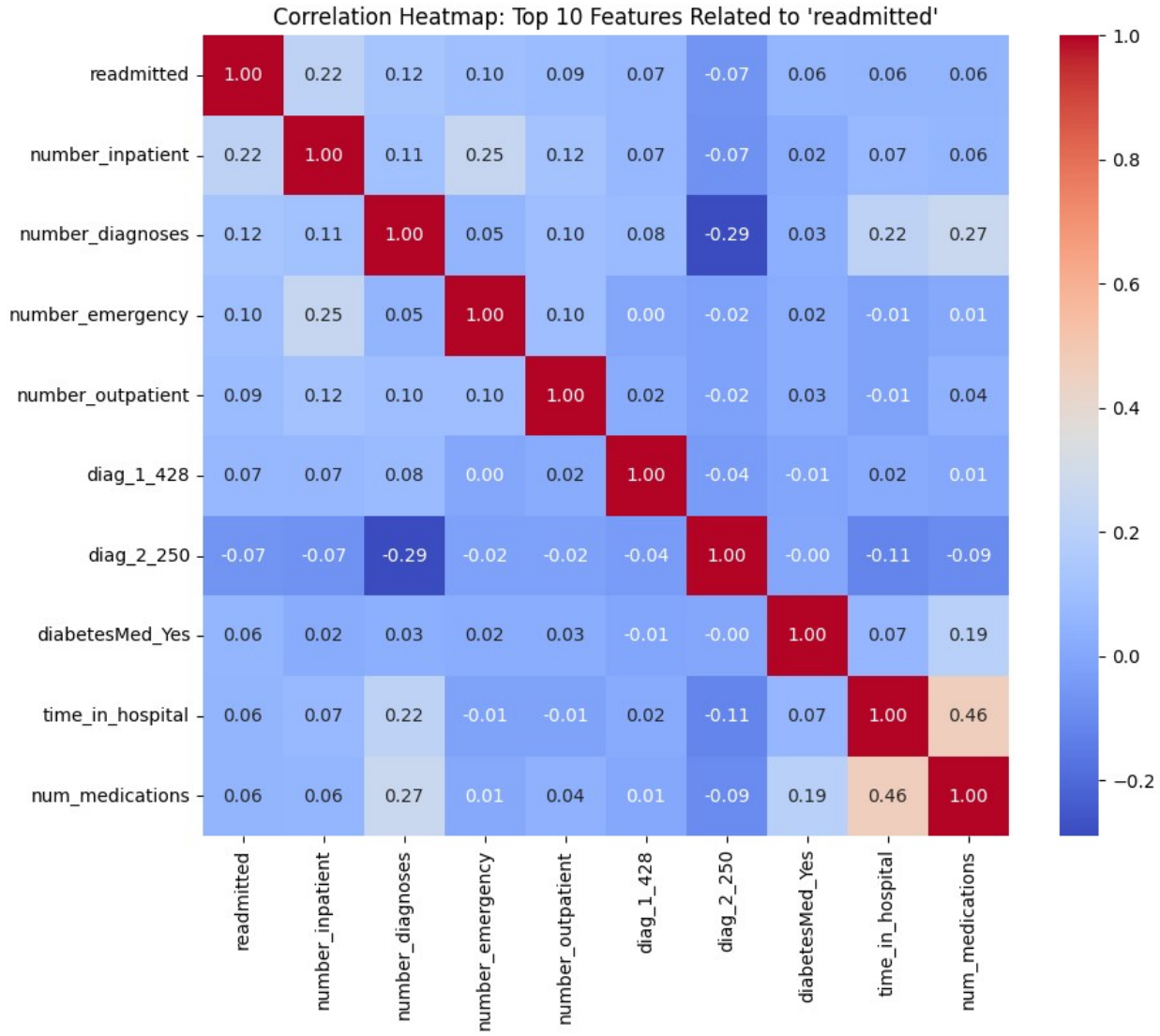
```
Features with significant correlation to 'readmitted':
 readmitted           1.000000
number_inpatient     0.216490
```

```
number_diagnoses     0.123423
number_emergency     0.100618
number_outpatient    0.087092
diag_1_428           0.070162
diabetesMed_Yes      0.059868
time_in_hospital     0.057583
num_medications      0.056378
change_No           -0.050143
diag_3_401          -0.050940
diag_3_250          -0.055564
diag_2_250          -0.066722
Name: readmitted, dtype: float64
```

```
# 2. Pairplot: Analyze pairwise relationships between significant features
important_features = significant_corr.index
sns.pairplot(df[important_features], diag_kind='kde', corner=True)
plt.suptitle("Pairwise Relationships of Significant Features", y=1.02)
plt.show()
```



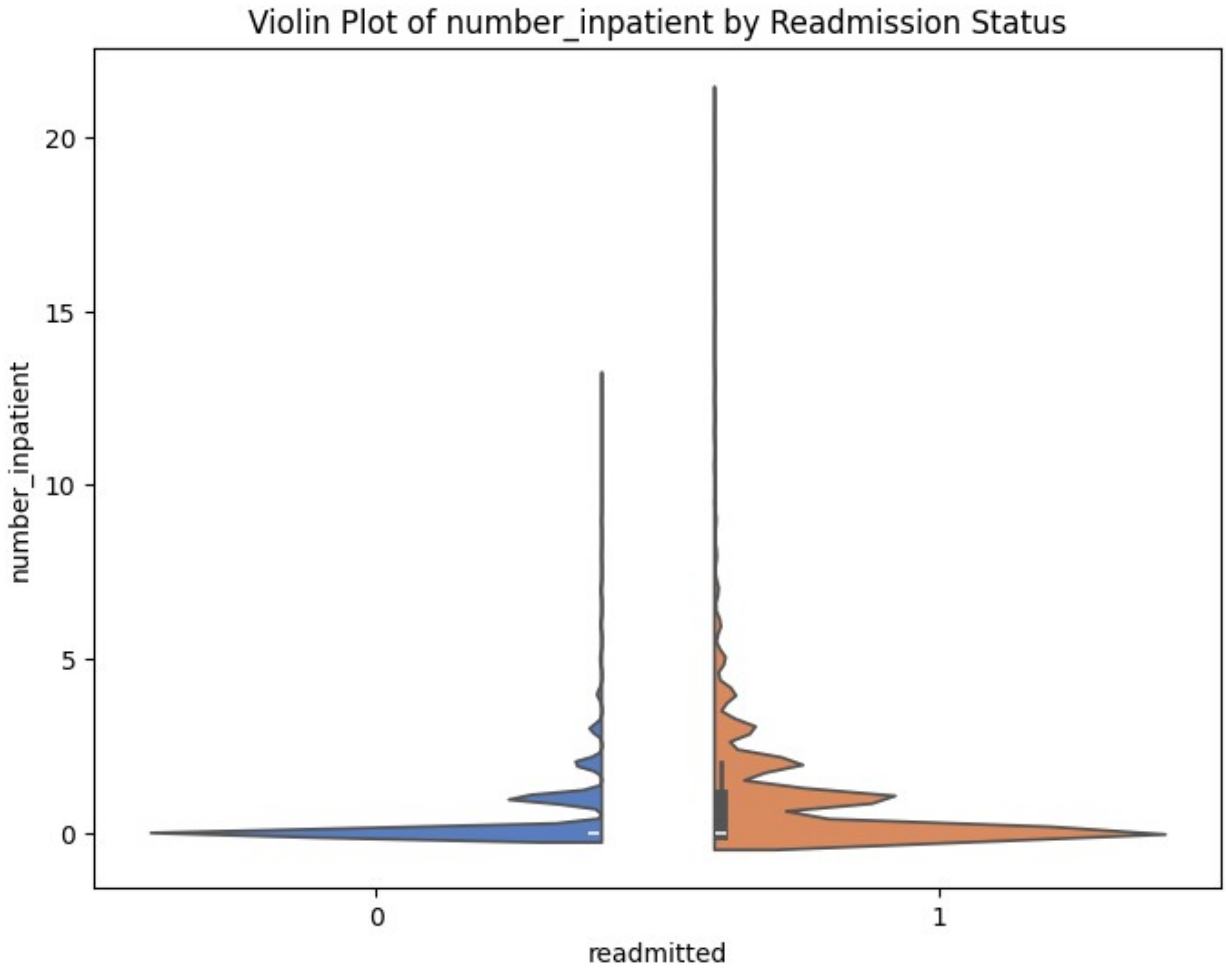

```
# 3. Detailed Heatmap: Focus on top 10 correlated features
top_features =
correlations['readmitted'].abs().sort_values(ascending=False).head(10)
.index
plt.figure(figsize=(10, 8))
sns.heatmap(df[top_features].corr(), annot=True, fmt=".2f",
cmap='coolwarm', cbar=True)
plt.title("Correlation Heatmap: Top 10 Features Related to
'readmitted'")
plt.show()
```



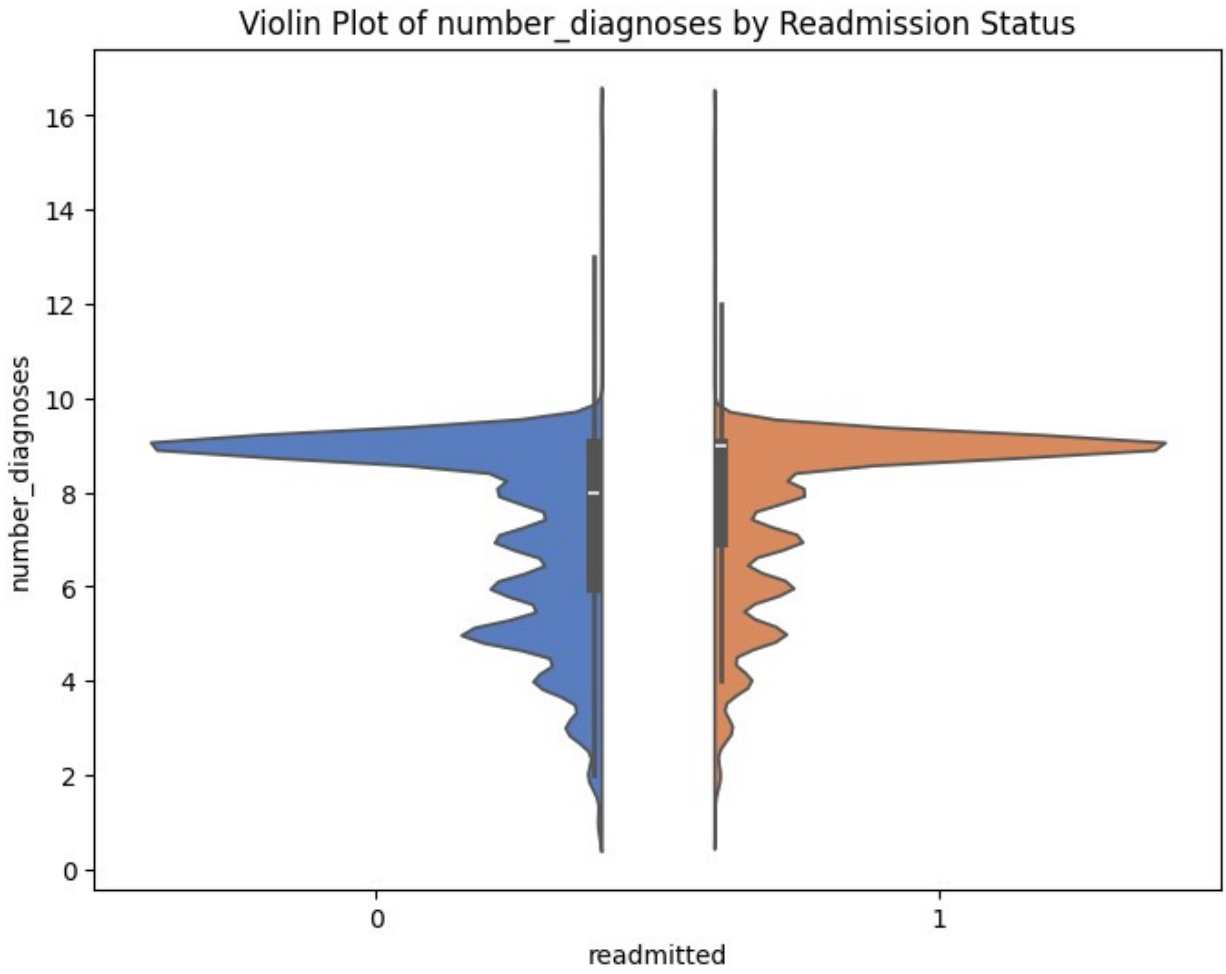

```
# 4. Violin/Boxplot: Explore distribution of top features
import warnings
warnings.filterwarnings("ignore", category=UserWarning)  # Suppress
warnings

for feature in top_features:
    if feature != 'readmitted':
        plt.figure(figsize=(8, 6))
        sns.violinplot(x='readmitted', y=feature, data=df,
palette='muted', split=True)
        plt.title(f"Violin Plot of {feature} by Readmission Status")
        plt.show()
```

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
```

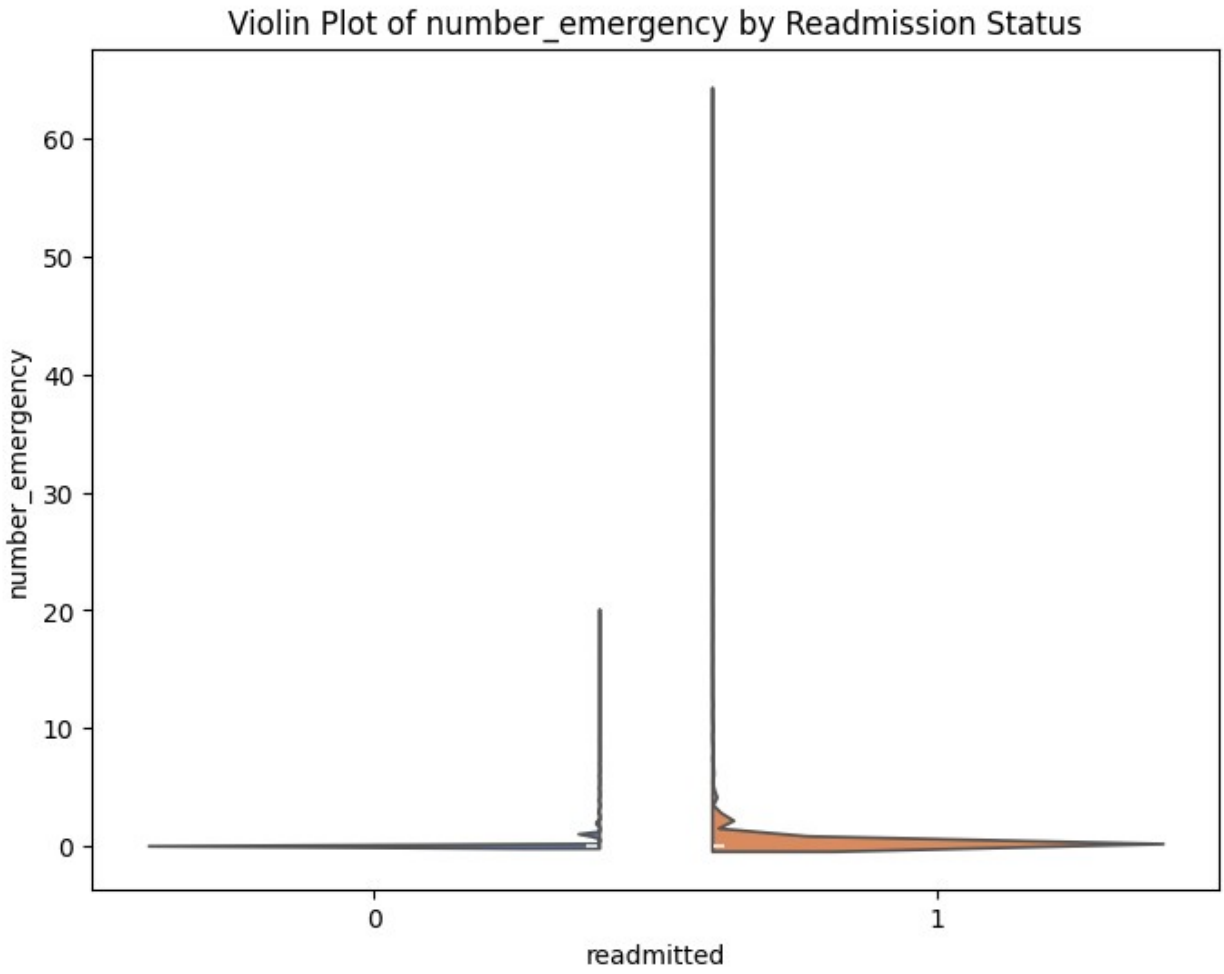
```
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```



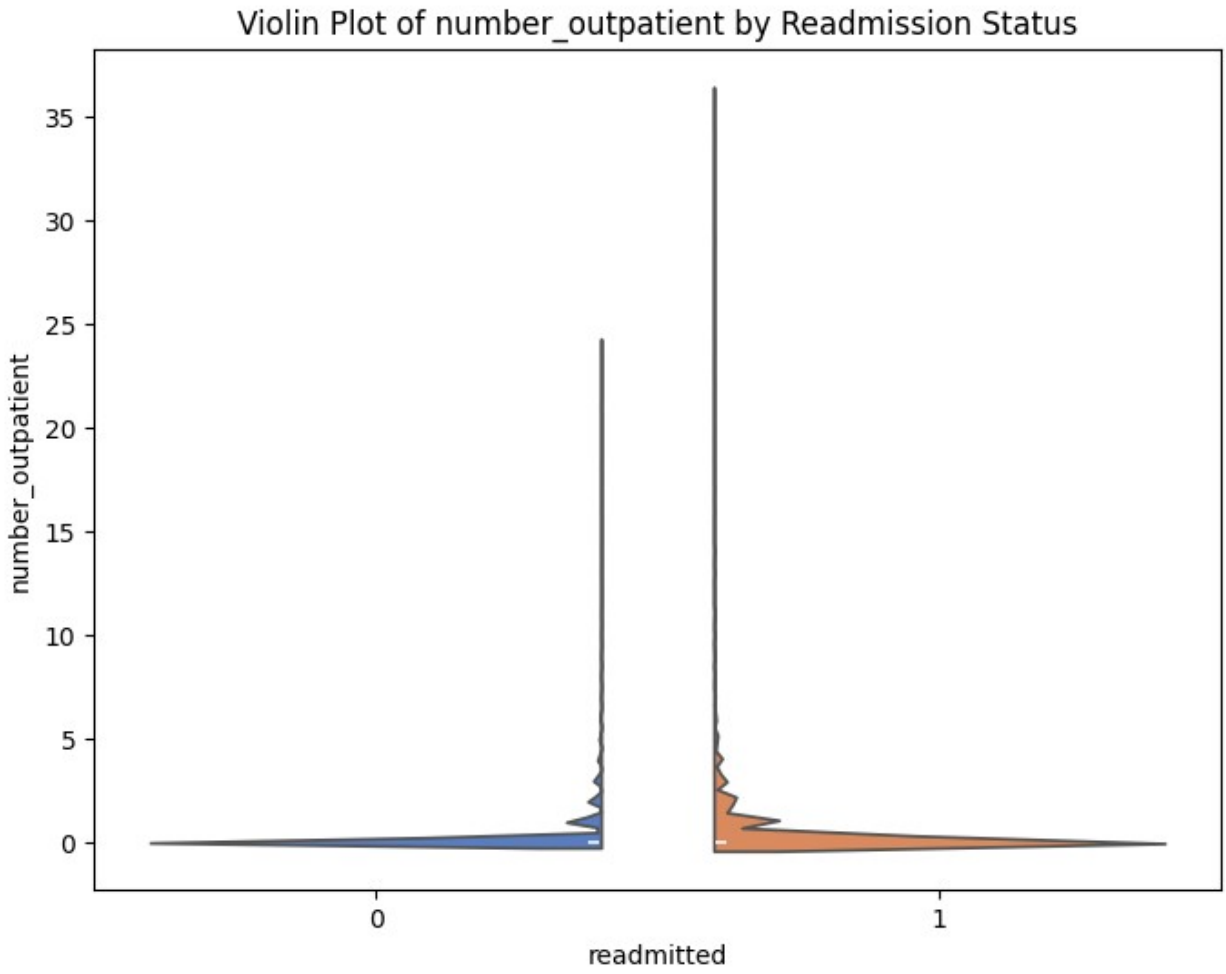

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```



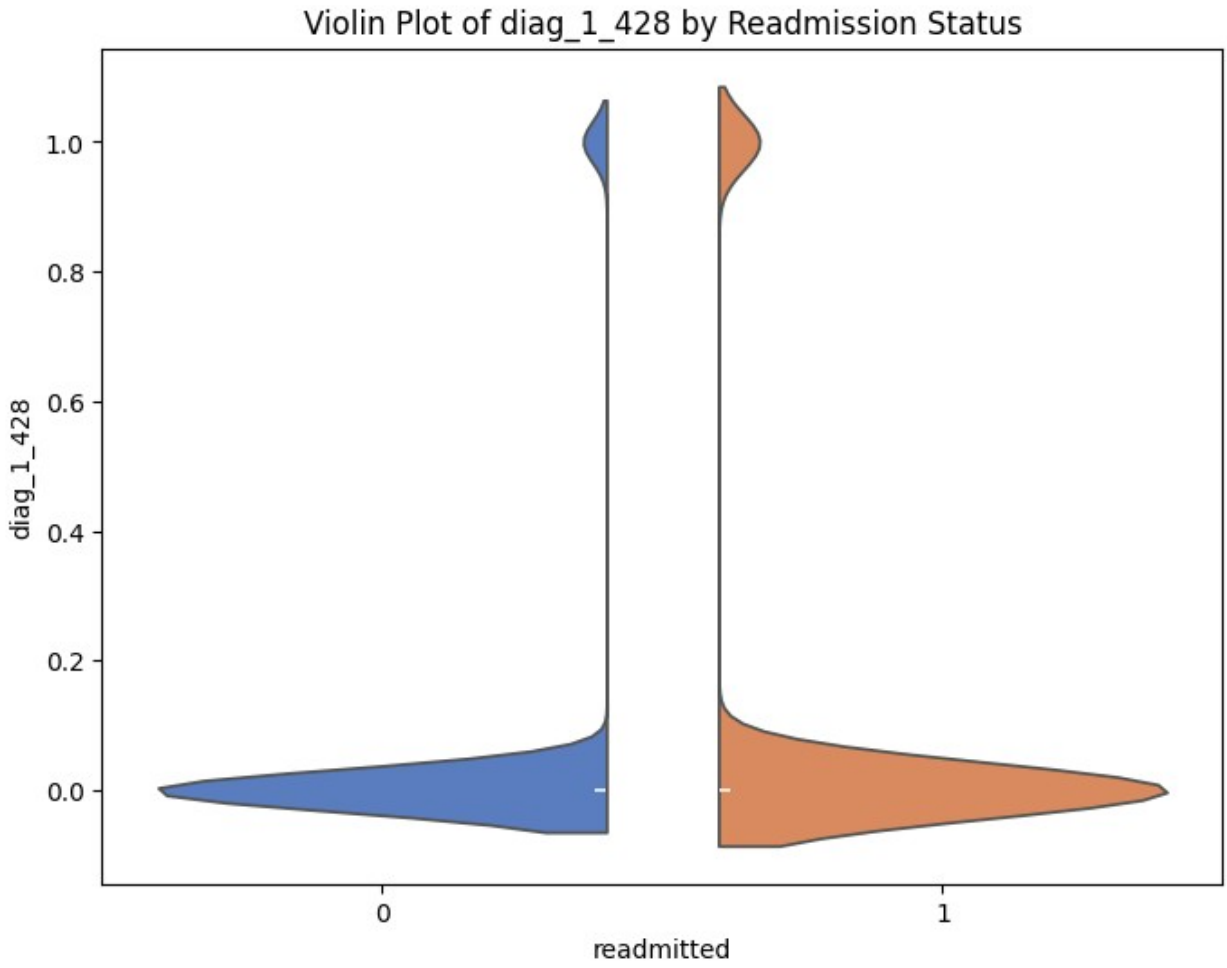

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```



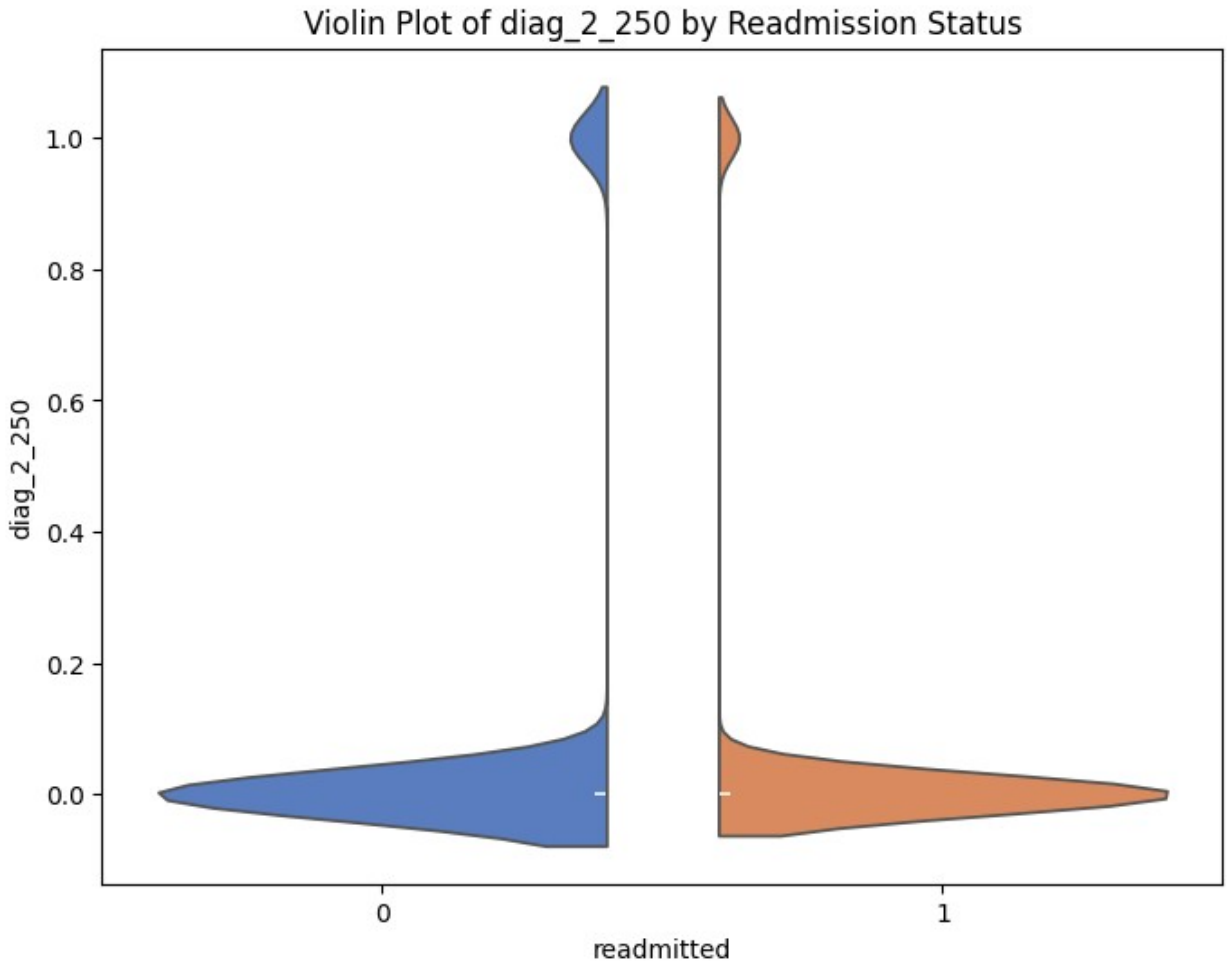

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```



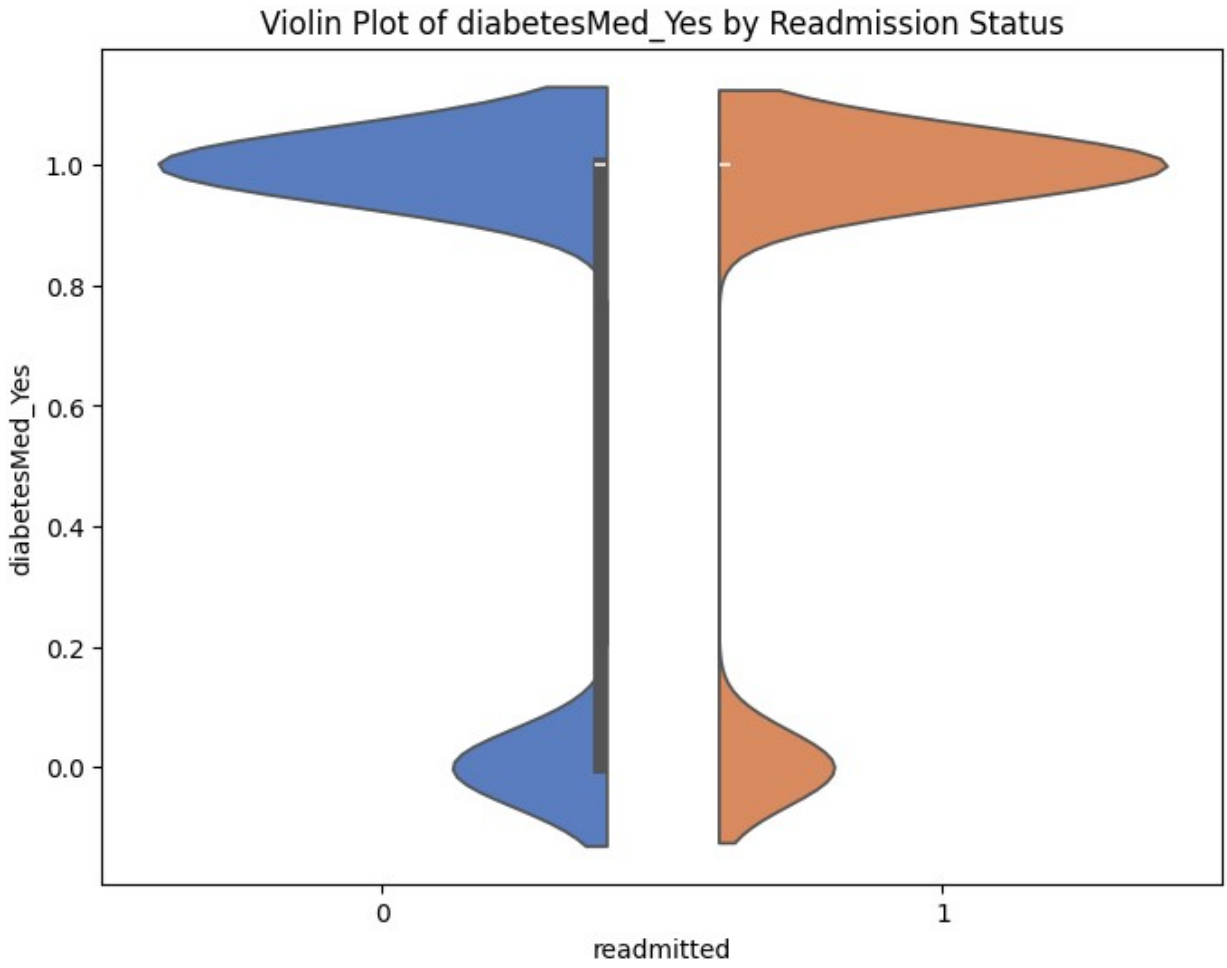

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```



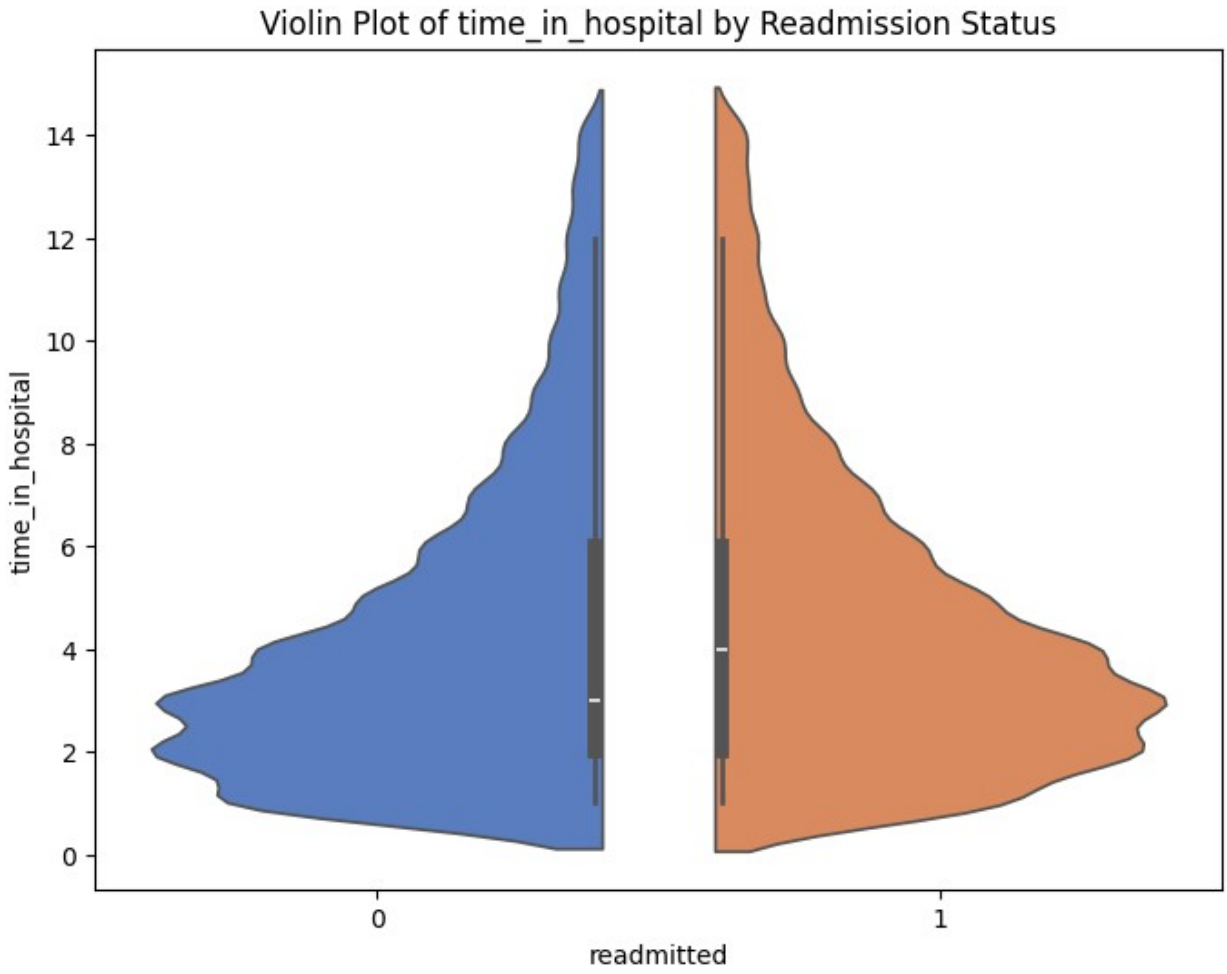

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```



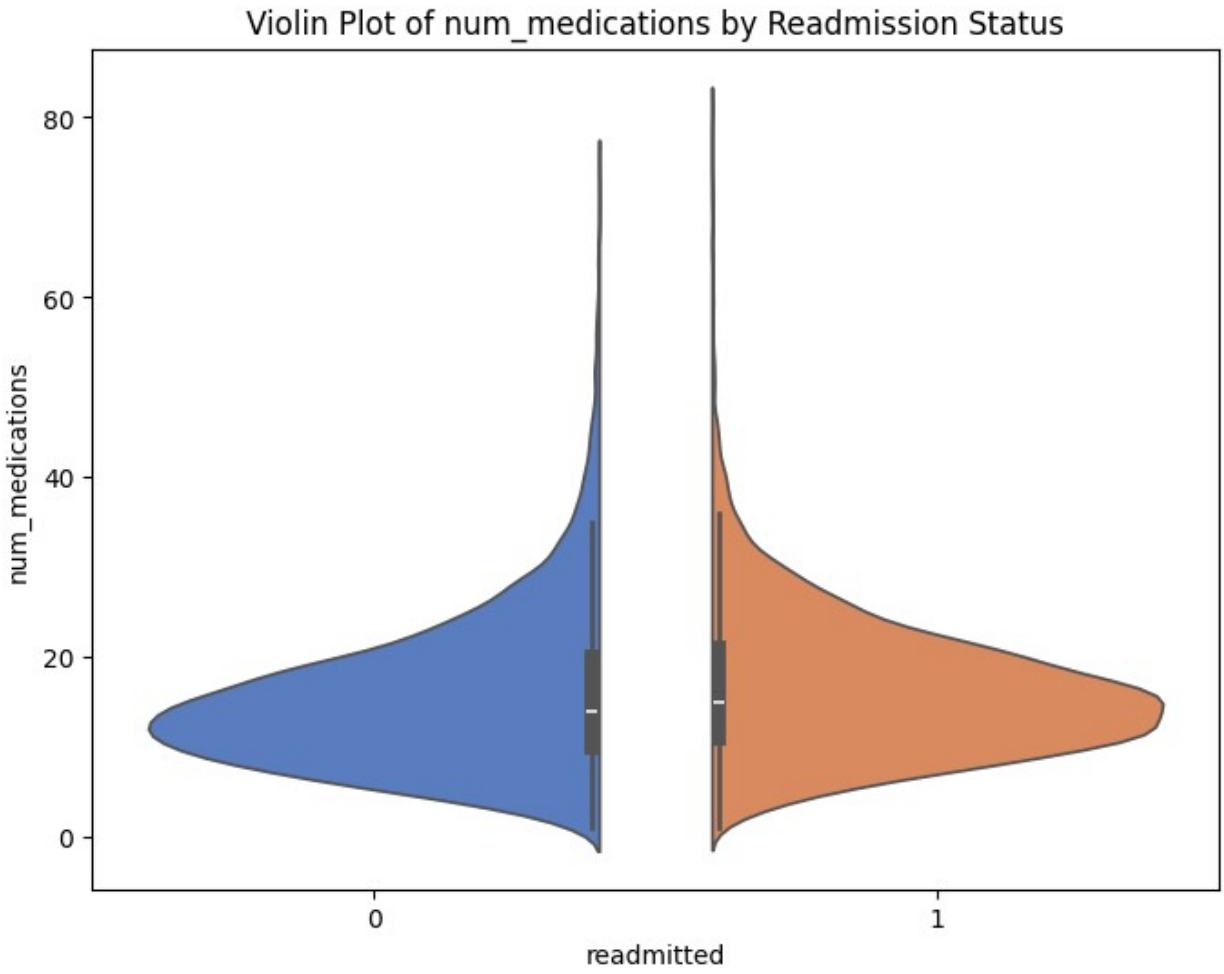

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```



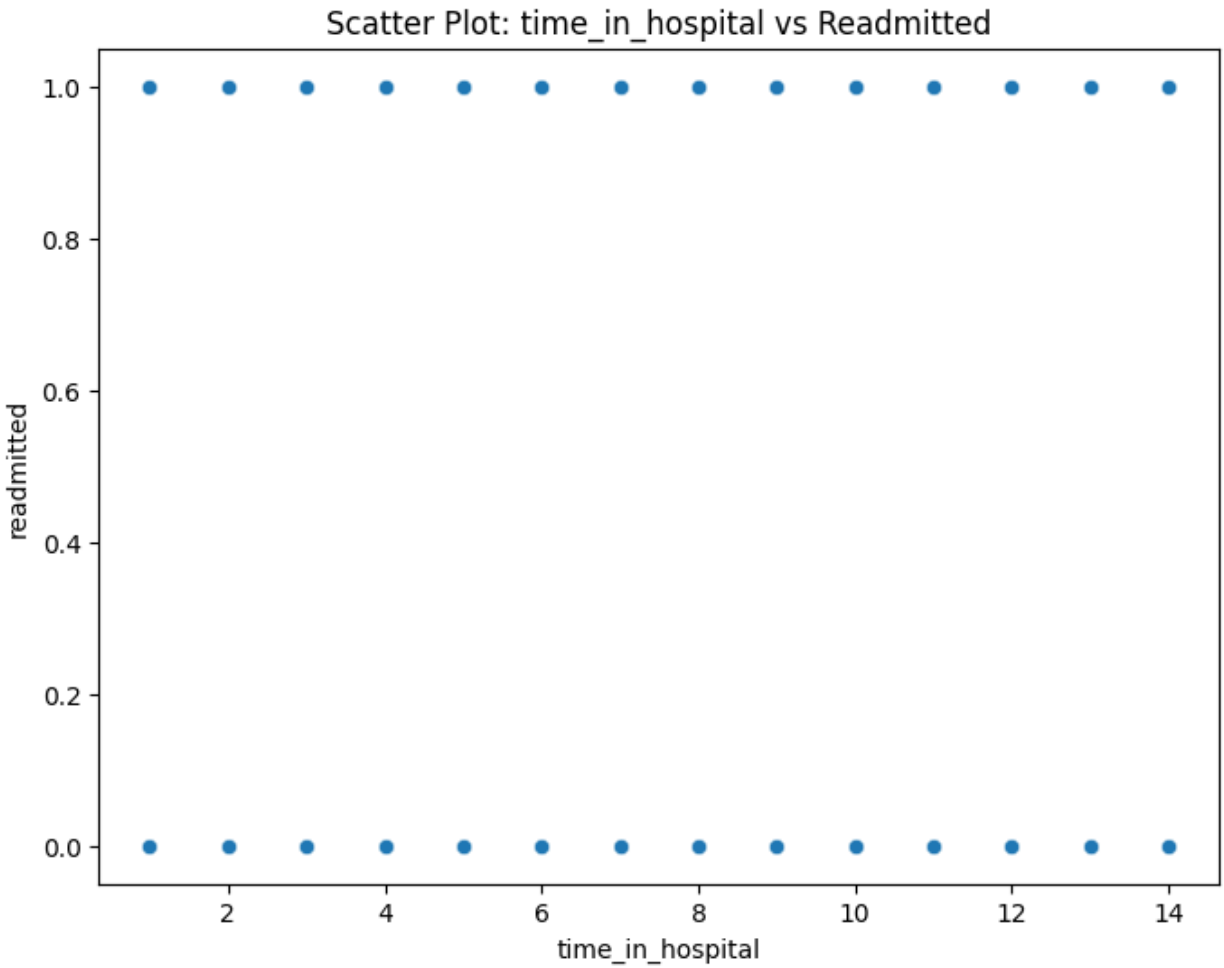

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```

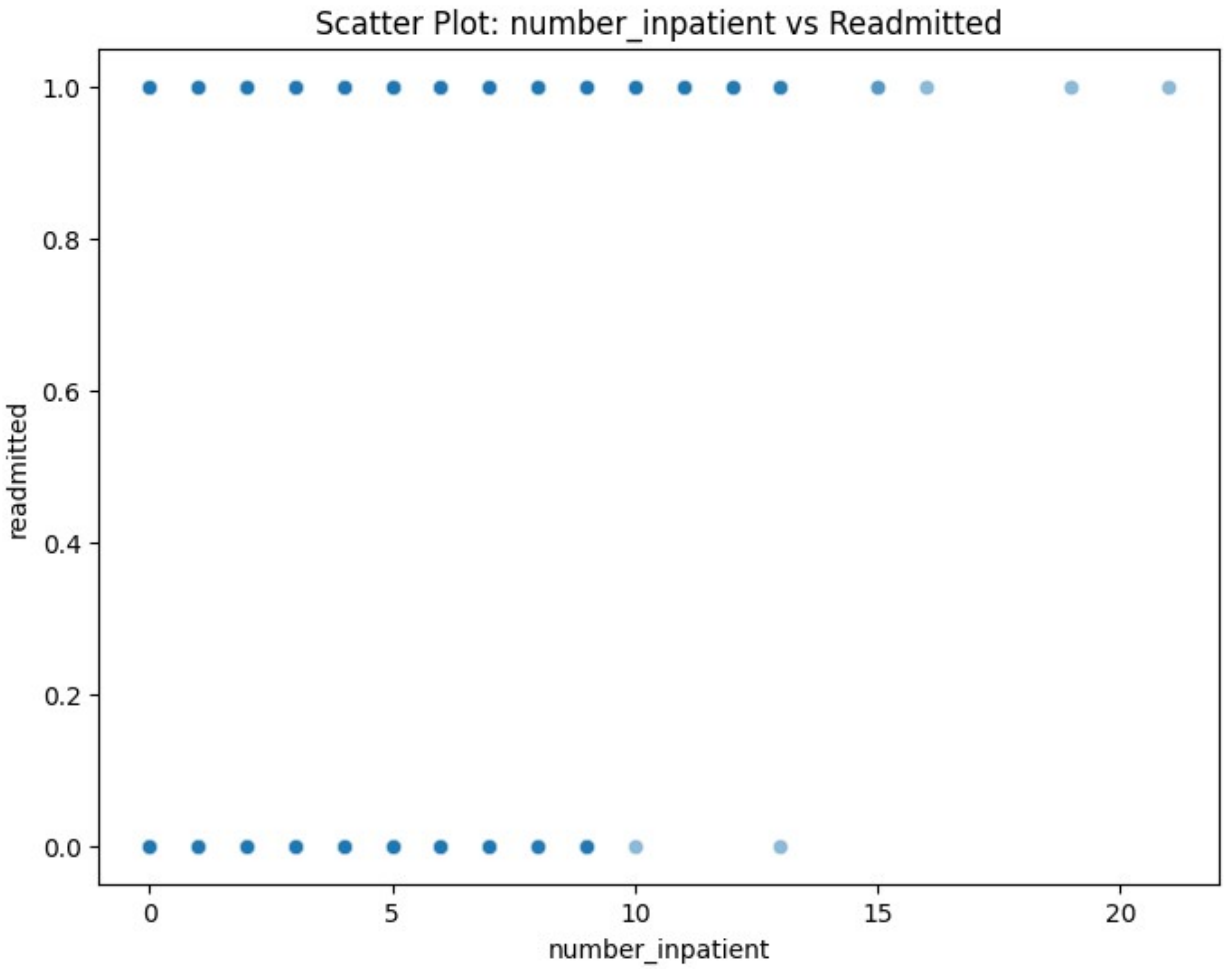


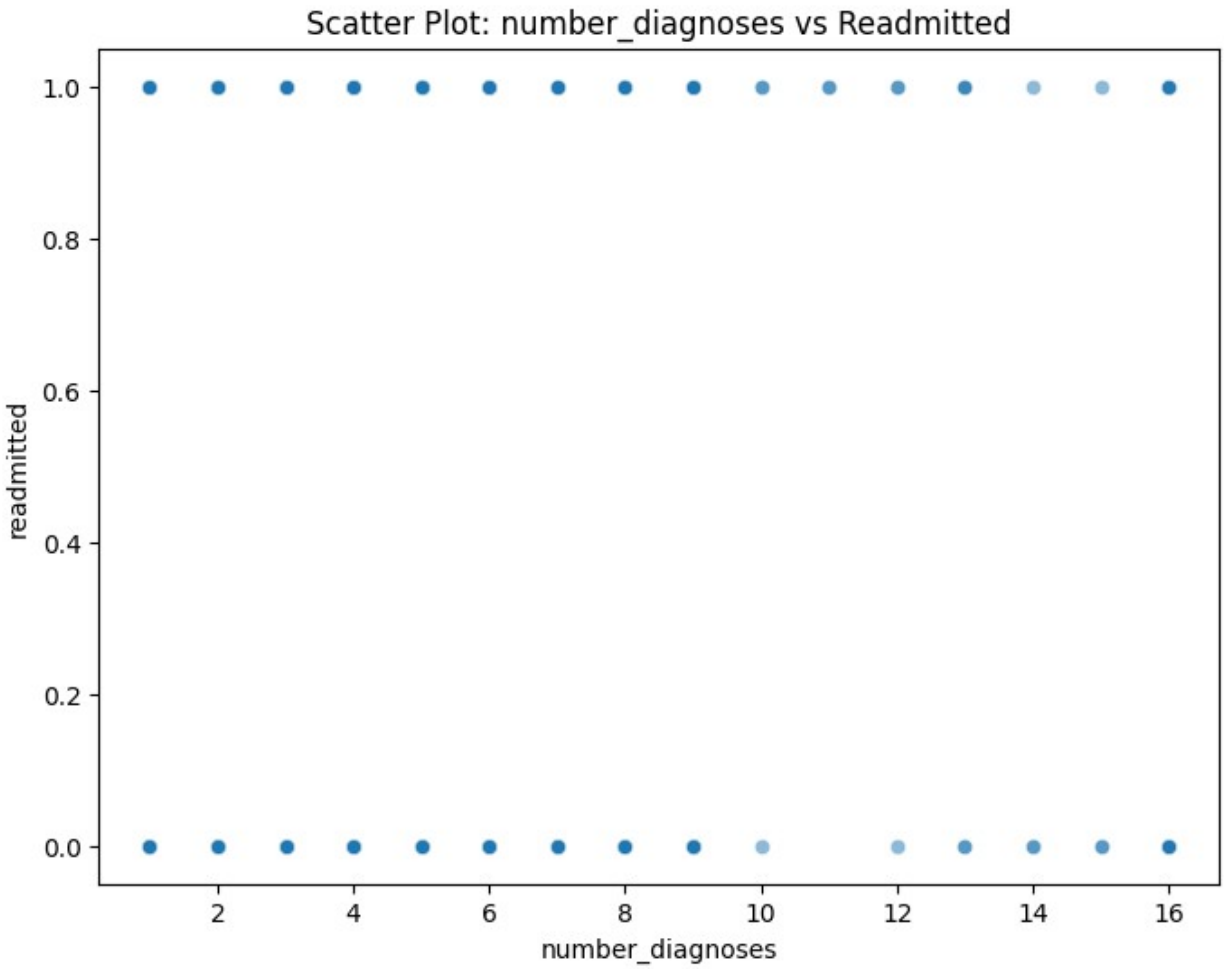

```
<ipython-input-11-20decc7235bd>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be
removed in v0.14.0. Assign the `x` variable to `hue` and set
`legend=False` for the same effect.

  sns.violinplot(x='readmitted', y=feature, data=df, palette='muted',
split=True)
```

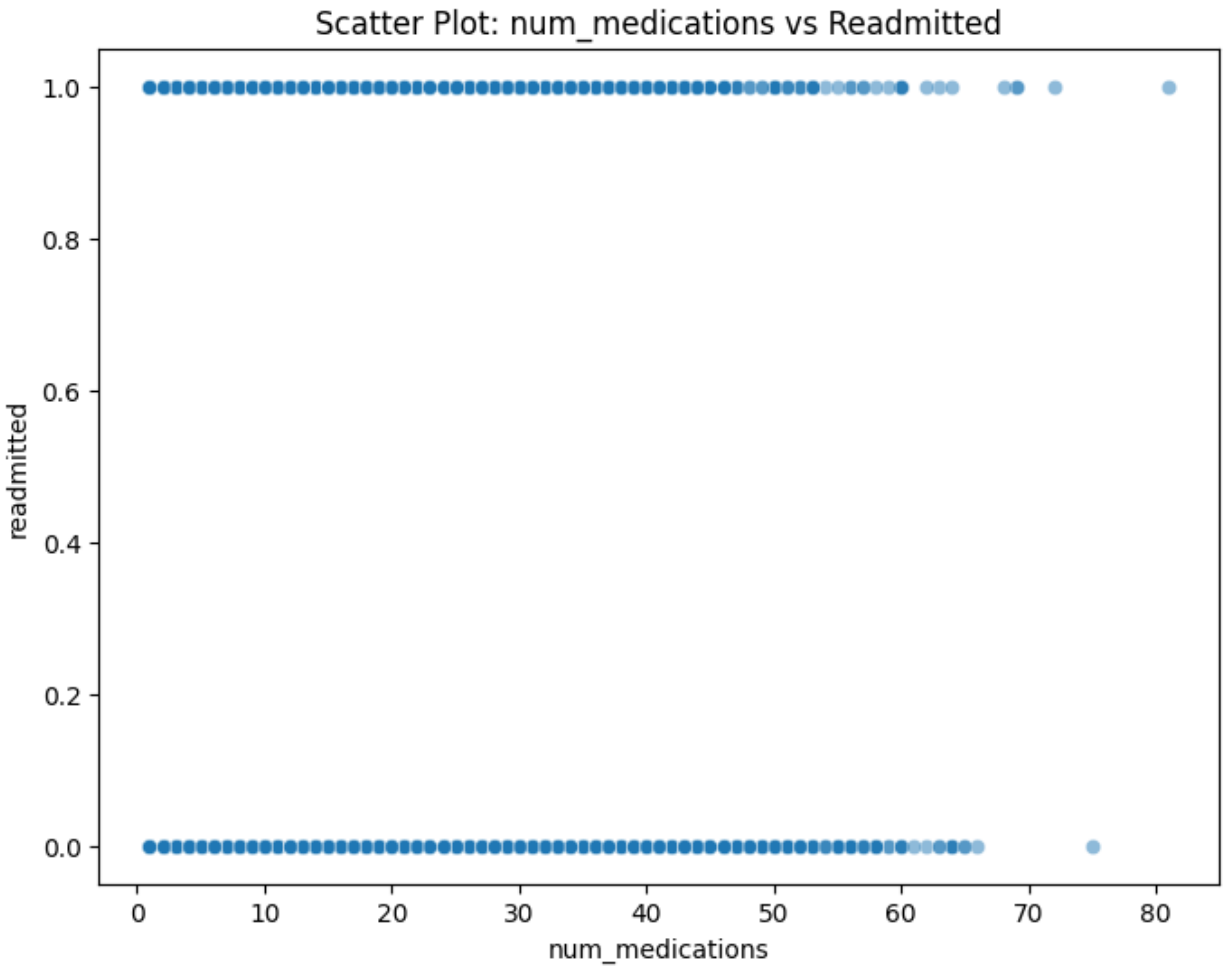



```
# 5. Scatter Plots for Continuous Features
continuous_features = ['time_in_hospital', 'number_inpatient',
'number_diagnoses', 'num_medications']
for feature in continuous_features:
    plt.figure(figsize=(8, 6))
    sns.scatterplot(x=feature, y='readmitted', data=df, alpha=0.5)
    plt.title(f"Scatter Plot: {feature} vs Readmitted")
    plt.show()
```


Scatter Plot: time_in_hospital vs Readmitted
readmitted
time_in_hospital
1.0
0.8
0.6
0.4
0.2
0.0
2
4
6
8
10
12
14


Scatter Plot: number_inpatient vs Readmitted
readmitted
number_inpatient
1.0
0.8
0.6
0.4
0.2
0.0
0
5
10
15
20


Scatter Plot: number_diagnoses vs Readmitted
readmitted
number_diagnoses
1.0
0.8
0.6
0.4
0.2
0.0
2
4
6
8
10
12
14
16




```
# 6. Correlation Analysis for Subgroups (e.g., gender-based
correlations)
for gender in df['gender_Female'].unique():
    subset = df[df['gender_Female'] == gender]
    subset_corr = subset.corr()
['readmitted'].sort_values(ascending=False)
    print(f"\nCorrelations for Gender {'Female' if gender == 1 else
'Male'}:\n", subset_corr.head(5))
```

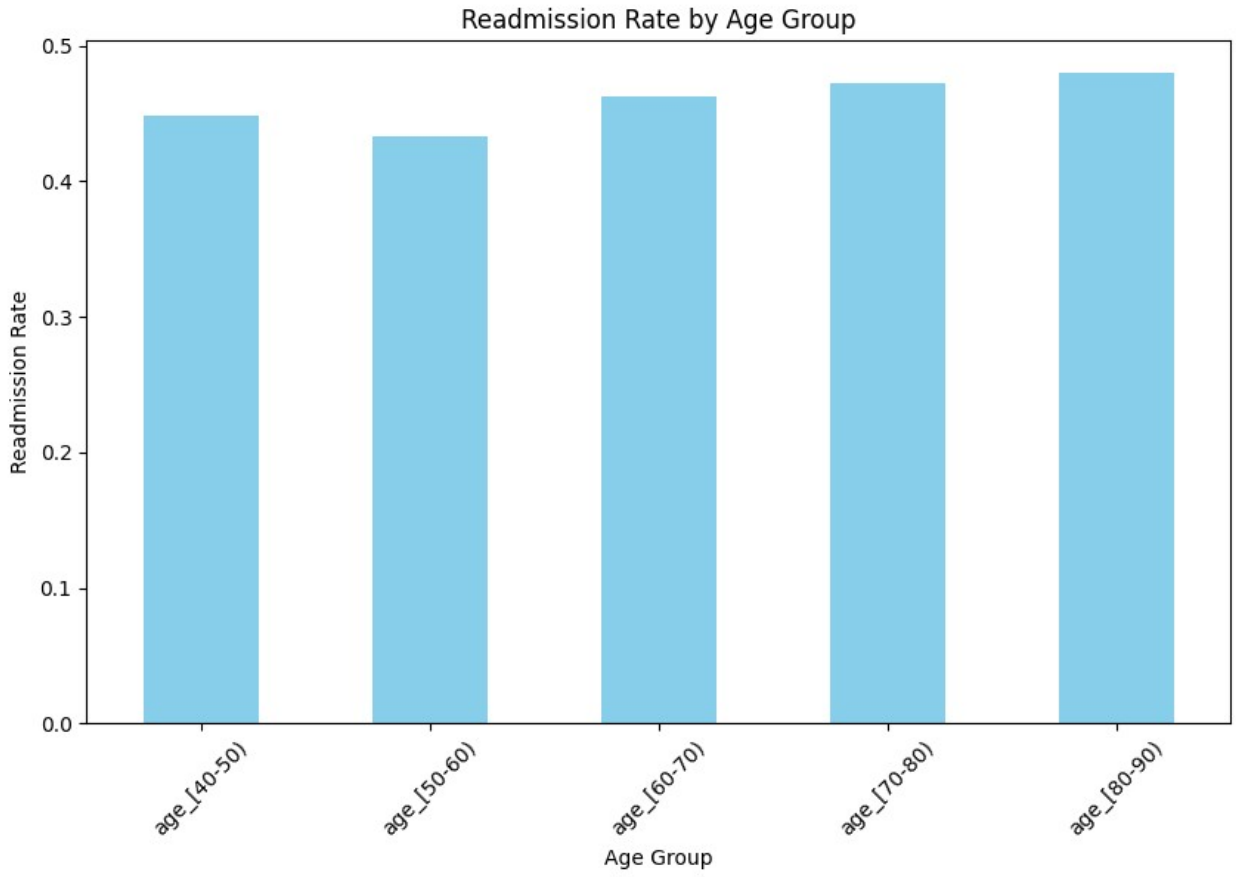
```
Correlations for Gender Male:
 readmitted           1.000000
number_inpatient     0.211901
number_diagnoses     0.130112
number_emergency     0.114683
number_outpatient    0.097189
Name: readmitted, dtype: float64

Correlations for Gender Female:
 readmitted           1.000000
```

```
number_inpatient     0.219895
number_diagnoses     0.117330
number_emergency     0.095258
number_outpatient    0.078136
Name: readmitted, dtype: float64

print(df.columns)

Index(['time_in_hospital', 'num_lab_procedures', 'num_procedures',
       'num_medications', 'number_outpatient', 'number_emergency',
       'number_inpatient', 'number_diagnoses', 'race_Caucasian',
       'race_AfricanAmerican', 'gender_Female', 'age_[70-80)',
'age_[60-70)',
       'age_[50-60)', 'age_[80-90)', 'age_[40-50)', 'payer_code_?',
       'payer_code_MC', 'payer_code_HM', 'payer_code_SP',
'payer_code_BC',
       'medical_specialty_?', 'medical_specialty_InternalMedicine',
       'medical_specialty_Emergency/Trauma',
       'medical_specialty_Family/GeneralPractice',
       'medical_specialty_Cardiology', 'diag_1_428', 'diag_1_414',
       'diag_1_786', 'diag_2_276', 'diag_2_428', 'diag_2_250',
'diag_2_427',
       'diag_3_250', 'diag_3_401', 'diag_3_276', 'diag_3_428',
       'max_glu_serum_None', 'A1Cresult_None', 'metformin_No',
       'repaglinide_No', 'nateglinide_No', 'chlorpropamide_No',
       'glimepiride_No', 'acetohexamide_No', 'glipizide_No',
'glyburide_No',
       'tolbutamide_No', 'pioglitazone_No', 'rosiglitazone_No',
'acarbose_No',
       'miglitol_No', 'troglitazone_No', 'tolazamide_No',
'examide_No',
       'citoglipton_No', 'insulin_No', 'glyburide-metformin_No',
       'glipizide-metformin_No', 'glimepiride-pioglitazone_No',
       'metformin-rosiglitazone_No', 'metformin-pioglitazone_No',
'change_No',
       'diabetesMed_Yes', 'readmitted'],
      dtype='object')

# Check for columns that might represent age
print(df.head())  # Display the first few rows of the dataset

   time_in_hospital  num_lab_procedures  num_procedures
num_medications  \
0                14                  41               0
11
1                 2                  30               0
12
2                 5                  66               0
22
3                 3                  63               0
```

```
8
4                  5                   40                0
6

   number_outpatient  number_emergency  number_inpatient
number_diagnoses  \
0                   0                 0                  0
6
1                   0                 0                  1
9
2                   1                 0                  2
9
3                   0                 0                  0
8
4                   0                 0                  1
9

   race_Caucasian  race_AfricanAmerican  ...  citoglipton_No
insulin_No  \
0               1                     0  ...               1
1
1               1                     0  ...               1
0
2               1                     0  ...               1
1
3               1                     0  ...               1
1
4               1                     0  ...               1
1

   glyburide-metformin_No  glipizide-metformin_No  \
0                       1                       1
1                       1                       1
2                       1                       1
3                       1                       1
4                       1                       1

   glimepiride-pioglitazone_No  metformin-rosiglitazone_No  \
0                            1                           1
1                            1                           1
2                            1                           1
3                            1                           1
4                            1                           1

   metformin-pioglitazone_No  change_No  diabetesMed_Yes  readmitted
0                          1          1                1           0
1                          1          0                1           1
2                          1          1                1           1
3                          1          1                1           1
4                          1          1                0           0
```

```
[5 rows x 65 columns]
```

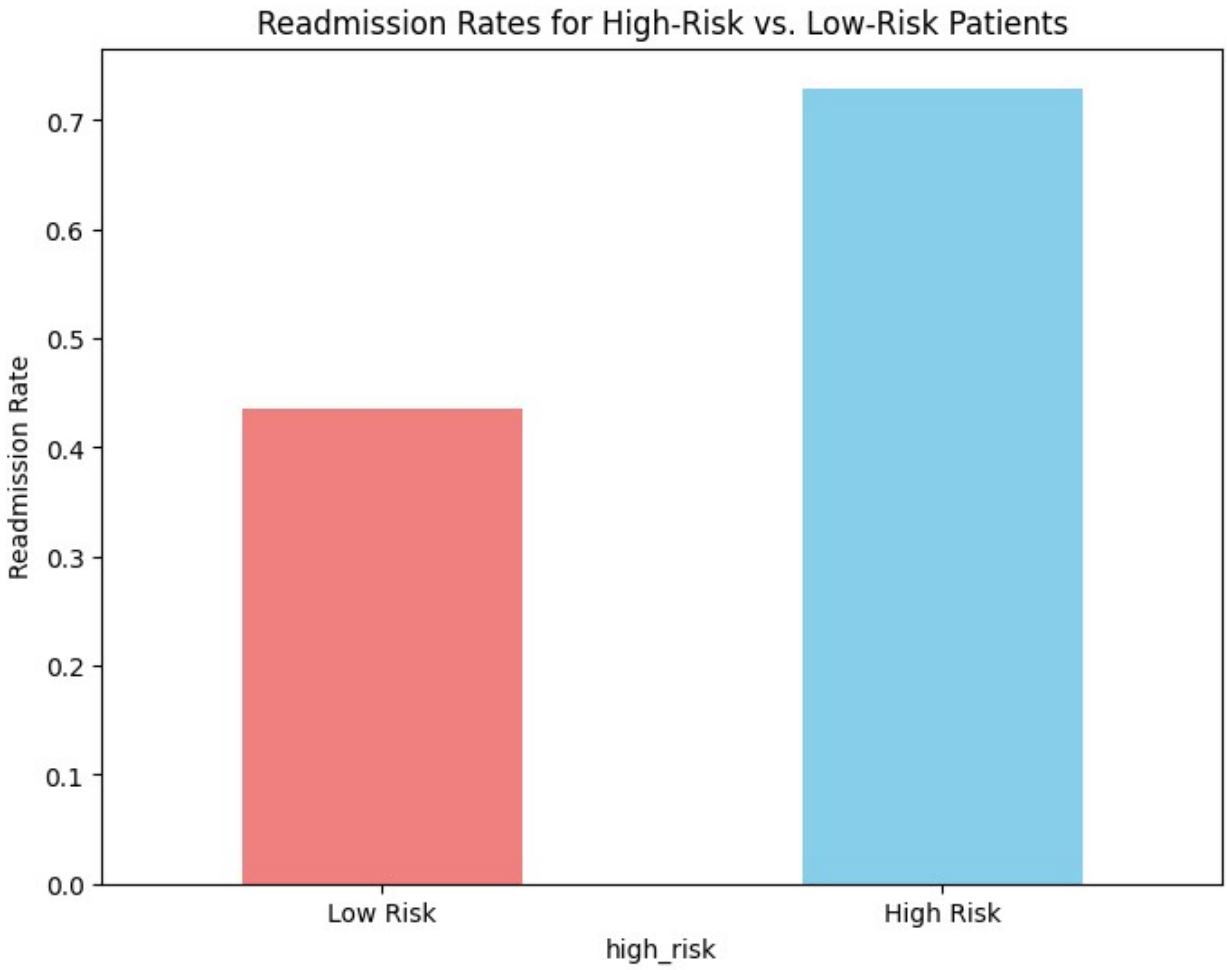
```
import matplotlib.pyplot as plt

# Age group columns
age_groups = ['age_[70-80)', 'age_[60-70)', 'age_[50-60)', 'age_[80-
90)', 'age_[40-50)']

# Calculating readmission rates for each age group
readmission_by_age_group = df[age_groups + ['readmitted']].melt(
    id_vars='readmitted',
    value_vars=age_groups,
    var_name='Age_Group',
    value_name='In_Age_Group'
)

# Filtering for rows where the patient belongs to the age group
readmission_by_age_group =
readmission_by_age_group[readmission_by_age_group['In_Age_Group'] ==
1]

# Calculating the mean readmission rate for each age group
age_group_readmission_rate =
readmission_by_age_group.groupby('Age_Group')
['readmitted'].mean().sort_index()

# Plotting the results
plt.figure(figsize=(10, 6))
age_group_readmission_rate.plot(kind='bar', color='skyblue')
plt.title('Readmission Rate by Age Group')
plt.xlabel('Age Group')
plt.ylabel('Readmission Rate')
plt.xticks(rotation=45)
plt.show()
```




1. Improved Patient Care

To highlight how identifying high-risk patients improves patient care, we can calculate the proportion of patients classified as high risk and visualize their readmission rates.

```
# Define high-risk patients based on number of inpatient visits
df['high_risk'] = df['number_inpatient'] > 2

# Calculate readmission rates for high-risk and low-risk patients
risk_readmission_rate = df.groupby('high_risk')['readmitted'].mean()

# Visualization
plt.figure(figsize=(8, 6))
risk_readmission_rate.plot(kind='bar', color=['lightcoral',
'skyblue'], legend=False)
plt.title('Readmission Rates for High-Risk vs. Low-Risk Patients')
plt.xticks(ticks=[0, 1], labels=['Low Risk', 'High Risk'], rotation=0)
plt.ylabel('Readmission Rate')
plt.show()
```




1. Cost Reduction

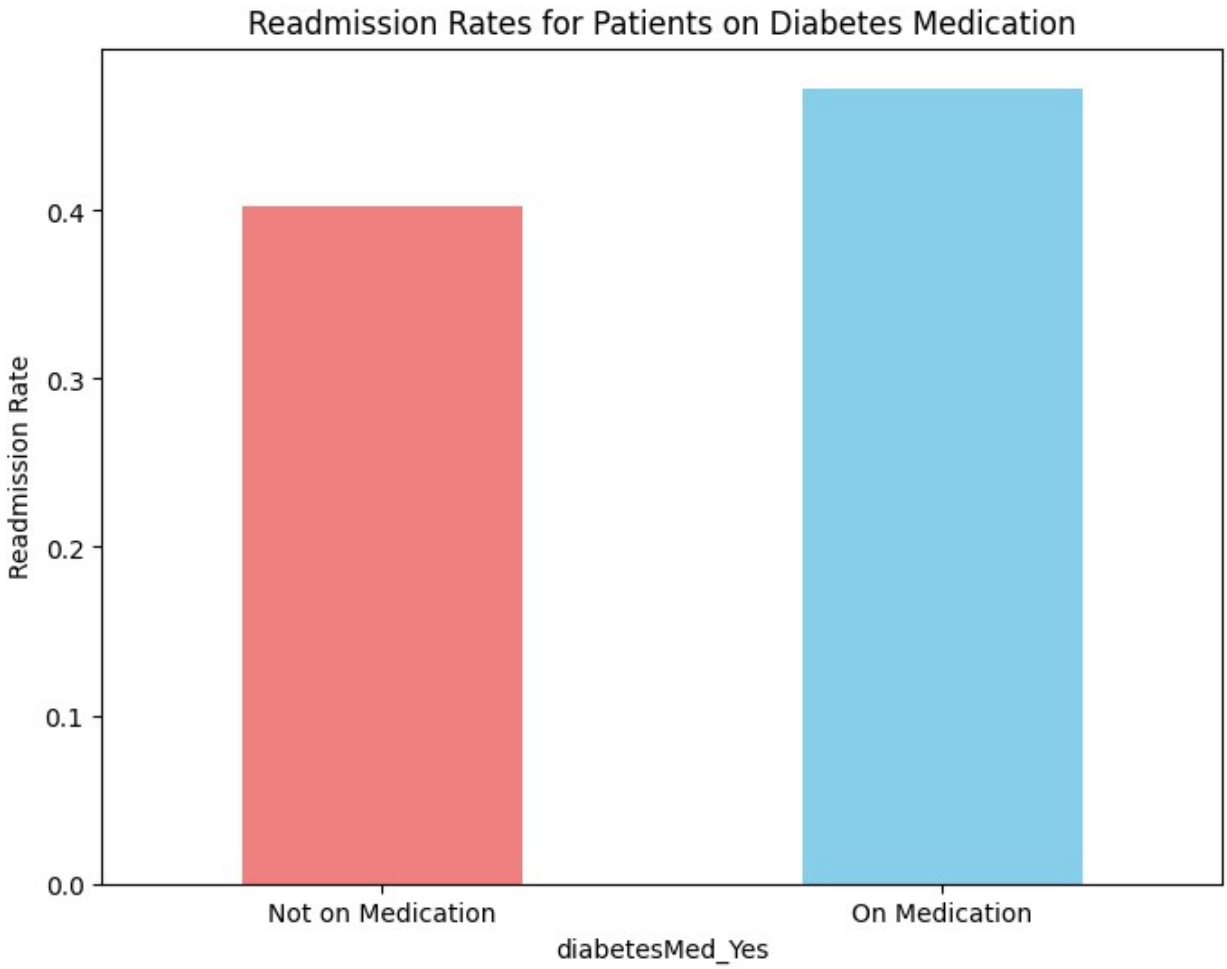
To demonstrate cost implications, calculate the average hospital stay length or number of procedures for patients who are readmitted versus not readmitted.

```
# Calculate average number of inpatient days for readmitted vs. non-
readmitted patients
avg_days_by_readmission = df.groupby('readmitted')
['time_in_hospital'].mean()

# Visualization
plt.figure(figsize=(8, 6))
avg_days_by_readmission.plot(kind='bar', color=['skyblue',
'lightcoral'])
plt.title('Average Time in Hospital by Readmission Status')
plt.xticks(ticks=[0, 1], labels=['Not Readmitted', 'Readmitted'],
rotation=0)
plt.ylabel('Average Time in Hospital (Days)')
plt.show()
```


1. Better Long-term Outcomes

To link interventions with long-term health outcomes, analyze trends in patients who received diabetes medication (or other treatments) versus those who did not.

```
# Analyze readmission rates for patients on diabetes medication vs. not
diabetes_med_readmission = df.groupby('diabetesMed_Yes')['readmitted'].mean()

# Visualization
plt.figure(figsize=(8, 6))
diabetes_med_readmission.plot(kind='bar', color=['lightcoral', 'skyblue'])
plt.title('Readmission Rates for Patients on Diabetes Medication')
plt.xticks(ticks=[0, 1], labels=['Not on Medication', 'On Medication'], rotation=0)
plt.ylabel('Readmission Rate')
plt.show()
```

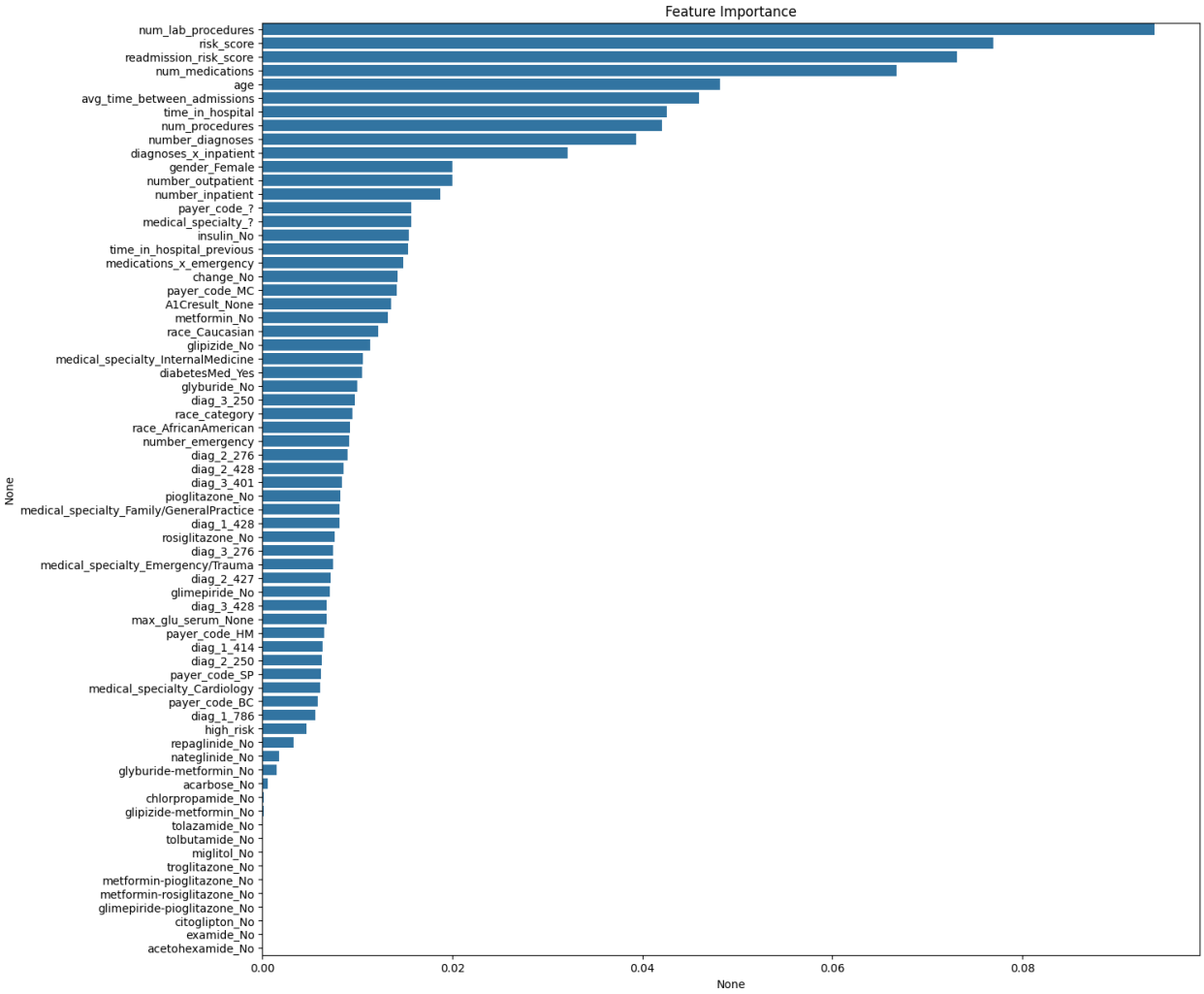



```
# Prepare the data for machine learning
X = df.drop('readmitted', axis=1)  # Features
y = df['readmitted']  # Target

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Initialize the Random Forest classifier
rf = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the model
rf.fit(X_train, y_train)

# Predict the test set results
y_pred = rf.predict(X_test)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
print("Classification Report:\n", classification_report(y_test,
y_pred))
```

```
Accuracy: 0.6182
Confusion Matrix:
 [[2002  729]
 [1180 1089]]
Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.73      0.68      2731
           1       0.60      0.48      0.53      2269

    accuracy                           0.62      5000
   macro avg       0.61      0.61      0.61      5000
weighted avg       0.62      0.62      0.61      5000
```

```
# Feature importance
importances = pd.Series(rf.feature_importances_,
index=X.columns).sort_values(ascending=False)
plt.figure(figsize=(15, 15))
sns.barplot(x=importances, y=importances.index)
plt.title('Feature Importance')
plt.show()
```




1. Time-Based Features

(a) Time since the last hospital admission Assuming you have admission_date and patient_id columns:

Adjusting Code for Time-Based Features

If there is no admission_date, we will skip time-based calculations or infer them indirectly based on available data like number_inpatient, number_emergency, and time_in_hospital.

1. Time Since Last Admission Without admission_date, this feature cannot be computed. Instead, we can focus on other available temporal features.

2. Average Time Between Admissions Use cumulative counts of inpatient visits as a proxy:

```
# Assuming 'number_inpatient' tracks cumulative inpatient visits
df['avg_time_between_admissions'] = df['time_in_hospital'] /
(df['number_inpatient'] + 1)
```

1. Time in Hospital During Previous Admissions

```
# Approximation using cumulative inpatient count
df['time_in_hospital_previous'] = df['time_in_hospital'] *
(df['number_inpatient'] - 1)
df['time_in_hospital_previous'] =
df['time_in_hospital_previous'].clip(lower=0)  # Ensure no negatives
```

Updated Risk Score

The risk score can include available demographic and hospital visit information:

```
# Example: Refined risk score based on available data
df['risk_score'] = (
    0.4 * df['number_inpatient'] +
    0.3 * df['number_emergency'] +
    0.2 * df['num_medications'] +
    0.1 * df['time_in_hospital']
)
```

Interaction Terms

We can still compute interactions using available features:

```
# Interaction between number of diagnoses and inpatient visits
df['diagnoses_x_inpatient'] = df['number_diagnoses'] *
df['number_inpatient']

# Interaction between medication count and emergency visits
df['medications_x_emergency'] = df['num_medications'] *
df['number_emergency']
```

Categorical Variables

If admission_type and discharge_disposition are unavailable, check the dataset for other categorical columns to encode:

```
# Example: Mapping race categories to numerical values
race_mapping = {
    'race_Caucasian': 1,
    'race_AfricanAmerican': 2,
    # Add other race categories if available
}
df['race_category'] = df[[col for col in df.columns if
col.startswith('race_')]].idxmax(axis=1)
df['race_category'] = df['race_category'].map(race_mapping)
```

Verification After applying the updated transformations:

```
print(df[['avg_time_between_admissions', 'time_in_hospital_previous',
          'risk_score', 'diagnoses_x_inpatient',
'medications_x_emergency']].head())
```

```
   avg_time_between_admissions  time_in_hospital_previous  risk_score
\
0                    14.000000                          0         3.6

1                     1.000000                          0         3.0

2                     1.666667                          5         5.7

3                     3.000000                          0         1.9

4                     2.500000                          0         2.1


   diagnoses_x_inpatient  medications_x_emergency
0                      0                        0
1                      9                        0
2                     18                        0
3                      0                        0
4                      9                        0
```

```
import re

# Convert age categories to numeric midpoints
def extract_midpoint(age_range):
    match = re.search(r'\[(\d+)-(\d+)\)', age_range)
    if match:
        return (int(match.group(1)) + int(match.group(2))) / 2
    return None

# Combine age category columns into a single numeric column
age_columns = [col for col in df.columns if col.startswith('age_')]
df['age'] = sum(df[col] * extract_midpoint(col) for col in
age_columns)

# Drop original age category columns
df.drop(columns=age_columns, inplace=True)

# Create a readmission risk score using existing features
df['readmission_risk_score'] = (
    df['time_in_hospital'] +
    df['number_inpatient'] * 2 +
    df['number_emergency'] +
    df['num_medications'] * 0.5
)

# Define features and target
X = df.drop(columns=['readmitted'])
```

```
y = df['readmitted']

# Train-test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)

# Preprocessing
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, roc_auc_score

# Identify numeric and categorical features
numeric_features = X.select_dtypes(include=['int64',
'float64']).columns
categorical_features = X.select_dtypes(include=['object',
'bool']).columns

preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numeric_features),
        ('cat', OneHotEncoder(handle_unknown='ignore'),
categorical_features)
    ])

# Pipeline with Random Forest
pipeline = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', RandomForestClassifier(n_estimators=100,
random_state=42))
])

# Train model
pipeline.fit(X_train, y_train)

# Evaluate model
y_pred = pipeline.predict(X_test)
y_prob = pipeline.predict_proba(X_test)[:, 1]

print("Classification Report:\n", classification_report(y_test,
y_pred))
print("ROC AUC Score:", roc_auc_score(y_test, y_prob))

Classification Report:
              precision    recall  f1-score   support

           0       0.62      0.71      0.66      2718
           1       0.59      0.49      0.53      2282
```

```
    accuracy                           0.61      5000
   macro avg       0.60      0.60      0.60      5000
weighted avg       0.61      0.61      0.60      5000

ROC AUC Score: 0.6434614982790744
```

```
from sklearn.model_selection import GridSearchCV

param_grid = {
    'classifier__n_estimators': [100, 200, 500],
    'classifier__max_depth': [10, 20, None],
    'classifier__min_samples_split': [2, 5, 10]
}
grid_search = GridSearchCV(pipeline, param_grid, cv=5,
scoring='roc_auc')
grid_search.fit(X_train, y_train)
print("Best Parameters:", grid_search.best_params_)
```

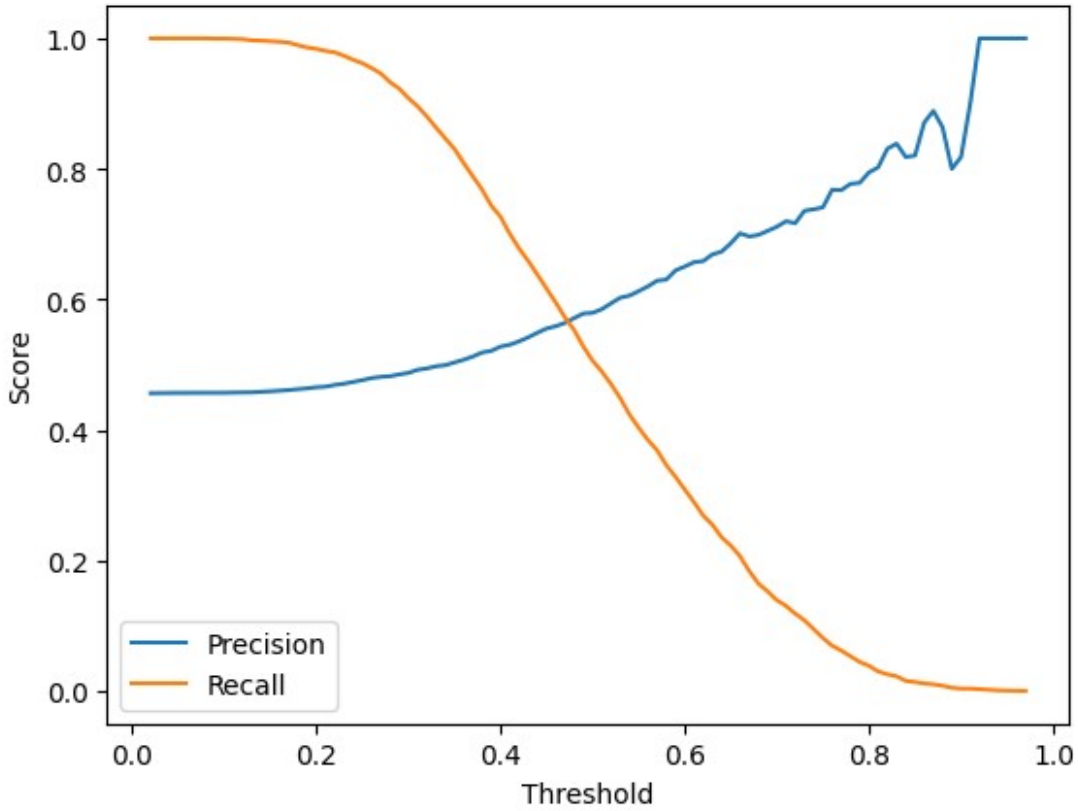
```
Best Parameters: {'classifier__max_depth': 10,
'classifier__min_samples_split': 2, 'classifier__n_estimators': 500}
```

```
feature_importances =
pipeline.named_steps['classifier'].feature_importances_
feature_names = numeric_features.tolist() + list(
    pipeline.named_steps['preprocessor'].transformers_[1]
[1].get_feature_names_out()
)
importance_df = pd.DataFrame({
    'Feature': feature_names,
    'Importance': feature_importances
}).sort_values(by='Importance', ascending=False)
print(importance_df)
```

```
                       Feature  Importance
1           num_lab_procedures    0.092990
61                  risk_score    0.077296
66      readmission_risk_score    0.074415
3              num_medications    0.065631
65                         age    0.047651
..                         ...         ...
50              citoglipton_No    0.000000
49                  examide_No    0.000000
39            acetohexamide_No    0.000000
56   metformin-pioglitazone_No    0.000000
55  metformin-rosiglitazone_No    0.000000

[69 rows x 2 columns]
```

```
from sklearn.metrics import precision_recall_curve
import matplotlib.pyplot as plt
```

```
precision, recall, thresholds = precision_recall_curve(y_test, y_prob)
plt.plot(thresholds, precision[:-1], label='Precision')
plt.plot(thresholds, recall[:-1], label='Recall')
plt.xlabel('Threshold')
plt.ylabel('Score')
plt.legend()
plt.show()
```




```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(pipeline, X_train, y_train, cv=5,
scoring='roc_auc')
print("Cross-Validated ROC AUC:", scores.mean())
```

```
Cross-Validated ROC AUC: 0.6549871736996441
```

```
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.metrics import classification_report, roc_auc_score

# Define models
models = {
    "Gradient Boosting Machine":
GradientBoostingClassifier(random_state=42),
```

```
    "XGBoost": XGBClassifier(use_label_encoder=False,
eval_metric='logloss', random_state=42)
}

# Iterate through models and evaluate
for name, model in models.items():
    pipeline = Pipeline(steps=[
        ('preprocessor', preprocessor),
        ('classifier', model)
    ])

    # Train model
    pipeline.fit(X_train, y_train)

    # Predict
    y_pred = pipeline.predict(X_test)
    y_prob = pipeline.predict_proba(X_test)[:, 1]

    # Evaluate performance
    print(f"\n{name}")
    print("Classification Report:\n", classification_report(y_test,
y_pred))
    print("ROC AUC Score:", roc_auc_score(y_test, y_prob))


Gradient Boosting Machine
Classification Report:
              precision    recall  f1-score   support

           0       0.63      0.75      0.68      2718
           1       0.61      0.47      0.53      2282

    accuracy                           0.62      5000
   macro avg       0.62      0.61      0.61      5000
weighted avg       0.62      0.62      0.61      5000

ROC AUC Score: 0.6651783900493932

XGBoost
Classification Report:
              precision    recall  f1-score   support

           0       0.62      0.67      0.64      2718
           1       0.56      0.50      0.53      2282

    accuracy                           0.59      5000
   macro avg       0.59      0.59      0.59      5000
weighted avg       0.59      0.59      0.59      5000

ROC AUC Score: 0.6418151396313343
```

Hyperparameter Tuning for GBM and XGBoost

To improve performance further, apply hyperparameter tuning:

GBM Hyperparameter Tuning

```
from sklearn.model_selection import GridSearchCV

param_grid_gbm = {
    'classifier__n_estimators': [100, 200],
    'classifier__learning_rate': [0.01, 0.1, 0.2],
    'classifier__max_depth': [3, 5, 7]
}

grid_search_gbm = GridSearchCV(pipeline, param_grid_gbm, cv=5,
scoring='roc_auc')
grid_search_gbm.fit(X_train, y_train)

print("Best Parameters for GBM:", grid_search_gbm.best_params_)
```

```
Best Parameters for GBM: {'classifier__learning_rate': 0.1,
'classifier__max_depth': 3, 'classifier__n_estimators': 200}
```

XGBoost Hyperparameter Tuning

```
param_grid_xgb = {
    'classifier__n_estimators': [100, 200],
    'classifier__learning_rate': [0.01, 0.1, 0.2],
    'classifier__max_depth': [3, 5, 7],
    'classifier__subsample': [0.8, 1.0]
}

grid_search_xgb = GridSearchCV(pipeline, param_grid_xgb, cv=5,
scoring='roc_auc')
grid_search_xgb.fit(X_train, y_train)

print("Best Parameters for XGBoost:", grid_search_xgb.best_params_)
```

```
Best Parameters for XGBoost: {'classifier__learning_rate': 0.1,
'classifier__max_depth': 3, 'classifier__n_estimators': 200,
'classifier__subsample': 0.8}
```

```
from imblearn.over_sampling import SMOTE

# Apply SMOTE to balance the dataset
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_train, y_train)

# Confirm the new class distribution
from collections import Counter
print("Class Distribution After SMOTE:", Counter(y_resampled))
```

```
Class Distribution After SMOTE: Counter({0: 10859, 1: 10859})
```

```
df['diagnoses_x_time_in_hospital'] = df['number_diagnoses'] *
df['time_in_hospital']
df['medications_x_emergency_visits'] = df['num_medications'] *
df['number_emergency']
df['inpatient_x_risk_score'] = df['number_inpatient'] *
df['risk_score']

df['combined_risk'] = df['risk_score'] + df['readmission_risk_score']

# Example: Group by age category to calculate mean readmission risk
age_group_risk = df.groupby('age')
['readmission_risk_score'].mean().reset_index()
print(age_group_risk)
```

```
    age  readmission_risk_score
0   0.0               12.157041
1  45.0               13.770799
2  55.0               13.838310
3  65.0               14.364768
4  75.0               14.174916
5  85.0               13.833451
```

```
from xgboost import XGBClassifier
from sklearn.metrics import classification_report, roc_auc_score

# Train an XGBoost model with balanced class weights
xgb_model = XGBClassifier(use_label_encoder=False,
eval_metric='logloss', random_state=42,
scale_pos_weight=len(y_train[y_train == 0]) / len(y_train[y_train ==
1]))
xgb_model.fit(X_resampled, y_resampled)

# Evaluate on test data
y_pred = xgb_model.predict(X_test)
y_prob = xgb_model.predict_proba(X_test)[:, 1]

print("Classification Report:\n", classification_report(y_test,
y_pred))
print("ROC AUC Score:", roc_auc_score(y_test, y_prob))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.60      0.62      2731
           1       0.55      0.60      0.58      2269

    accuracy                           0.60      5000
   macro avg       0.60      0.60      0.60      5000
weighted avg       0.60      0.60      0.60      5000
```

```
ROC AUC Score: 0.6379451828644528
```

```
from imblearn.over_sampling import SMOTE
from collections import Counter

# Apply SMOTE to balance the dataset
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X_train, y_train)

print("Class Distribution After SMOTE:", Counter(y_resampled))
```

```
Class Distribution After SMOTE: Counter({0: 10859, 1: 10859})
```

```
# Calculate class weights
class_weights = len(y_train[y_train == 0]) / len(y_train[y_train ==
1])

xgb_model = XGBClassifier(use_label_encoder=False,
eval_metric='logloss', random_state=42,
scale_pos_weight=class_weights)
```

Hyperparameter Optimization

GridSearchCV for Hyperparameter Tuning

```
from sklearn.model_selection import GridSearchCV

# Define parameter grid
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [3, 5, 7],
    'learning_rate': [0.01, 0.1, 0.2],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.6, 0.8, 1.0]
}

grid_search = GridSearchCV(XGBClassifier(use_label_encoder=False,
eval_metric='logloss', random_state=42),
                           param_grid, cv=5, scoring='roc_auc')
grid_search.fit(X_resampled, y_resampled)

print("Best Parameters:", grid_search.best_params_)
```

```
Best Parameters: {'colsample_bytree': 0.6, 'learning_rate': 0.01,
'max_depth': 7, 'n_estimators': 300, 'subsample': 0.8}
```

Threshold Tuning

Precision-Recall Curve and Adjust Threshold

```
xgb_model.fit(X_resampled, y_resampled)
```

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None,
early_stopping_rounds=None,
              enable_categorical=False, eval_metric='logloss',
              feature_types=None, gamma=None, grow_policy=None,
              importance_type=None, interaction_constraints=None,
              learning_rate=None, max_bin=None,
max_cat_threshold=None,
              max_cat_to_onehot=None, max_delta_step=None,
max_depth=None,
              max_leaves=None, min_child_weight=None, missing=nan,
              monotone_constraints=None, multi_strategy=None,
n_estimators=None,
              n_jobs=None, num_parallel_tree=None,
random_state=42, ...)
```

```
from xgboost import XGBClassifier

xgb_model = XGBClassifier(use_label_encoder=False,
eval_metric='logloss', random_state=42)
xgb_model.fit(X_resampled, y_resampled)
```

```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None,
early_stopping_rounds=None,
              enable_categorical=False, eval_metric='logloss',
              feature_types=None, gamma=None, grow_policy=None,
              importance_type=None, interaction_constraints=None,
              learning_rate=None, max_bin=None,
max_cat_threshold=None,
              max_cat_to_onehot=None, max_delta_step=None,
max_depth=None,
              max_leaves=None, min_child_weight=None, missing=nan,
              monotone_constraints=None, multi_strategy=None,
n_estimators=None,
              n_jobs=None, num_parallel_tree=None,
random_state=42, ...)
```

```
y_prob = xgb_model.predict_proba(X_test)[:, 1]
```

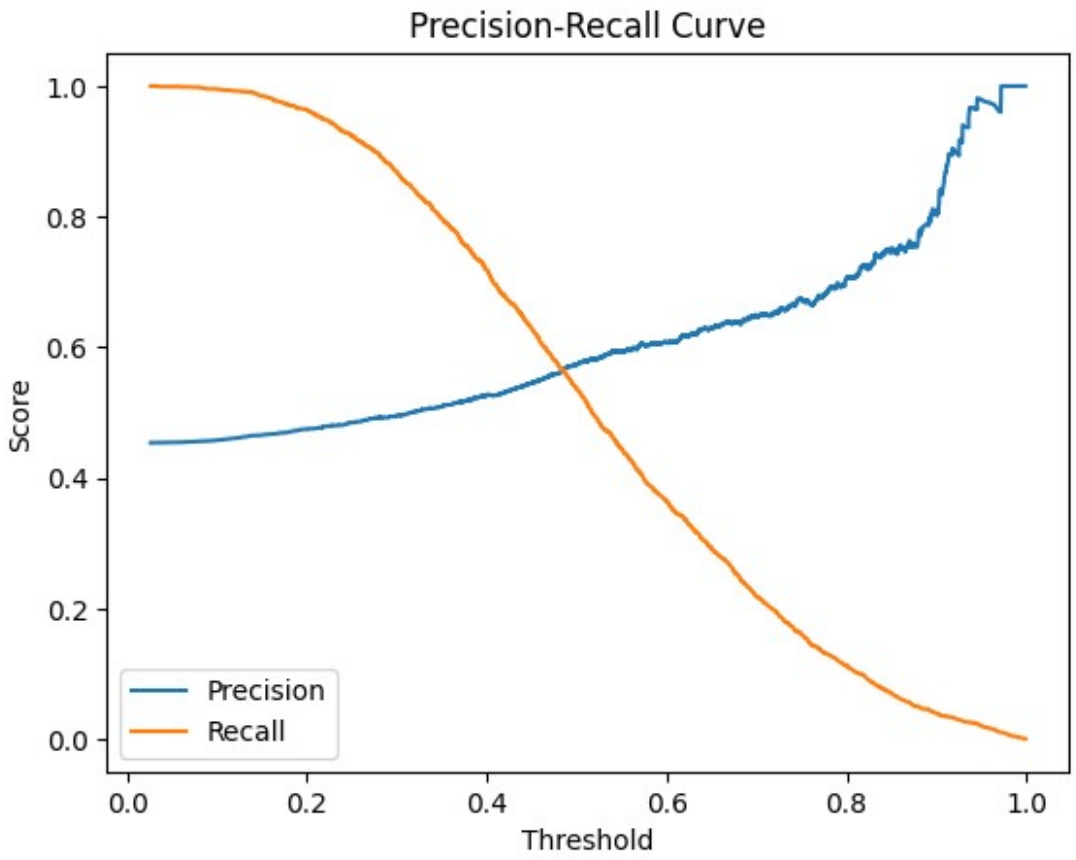
```
print("Model fitted:", hasattr(xgb_model, 'feature_importances_'))
```

```
Model fitted: True
```

```
print(X_resampled.shape, X_test.shape)
```

```
(21718, 68) (5000, 68)
```

```python
from sklearn.metrics import precision_recall_curve
import numpy as np

# Predict probabilities
y_prob = xgb_model.predict_proba(X_test)[:, 1]

# Compute Precision-Recall curve
precision, recall, thresholds = precision_recall_curve(y_test, y_prob)

# Plot Precision-Recall curve
import matplotlib.pyplot as plt
plt.plot(thresholds, precision[:-1], label='Precision')
plt.plot(thresholds, recall[:-1], label='Recall')
plt.xlabel('Threshold')
plt.ylabel('Score')
plt.legend()
plt.title('Precision-Recall Curve')
plt.show()

# Adjust threshold
optimal_threshold = thresholds[np.argmax(precision + recall)]
print("Optimal Threshold:", optimal_threshold)

# Apply new threshold
y_pred_adjusted = (y_prob >= optimal_threshold).astype(int)
```




```
Optimal Threshold: 0.13906224
```

Alternative Models

LightGBM Implementation

```
from lightgbm import LGBMClassifier

# Train LightGBM model
lgbm_model = LGBMClassifier(class_weight='balanced', random_state=42)
lgbm_model.fit(X_resampled, y_resampled)

# Evaluate on test data
y_pred_lgbm = lgbm_model.predict(X_test)
y_prob_lgbm = lgbm_model.predict_proba(X_test)[:, 1]

print("LightGBM Classification Report:\n",
classification_report(y_test, y_pred_lgbm))
print("LightGBM ROC AUC Score:", roc_auc_score(y_test, y_prob_lgbm))
```

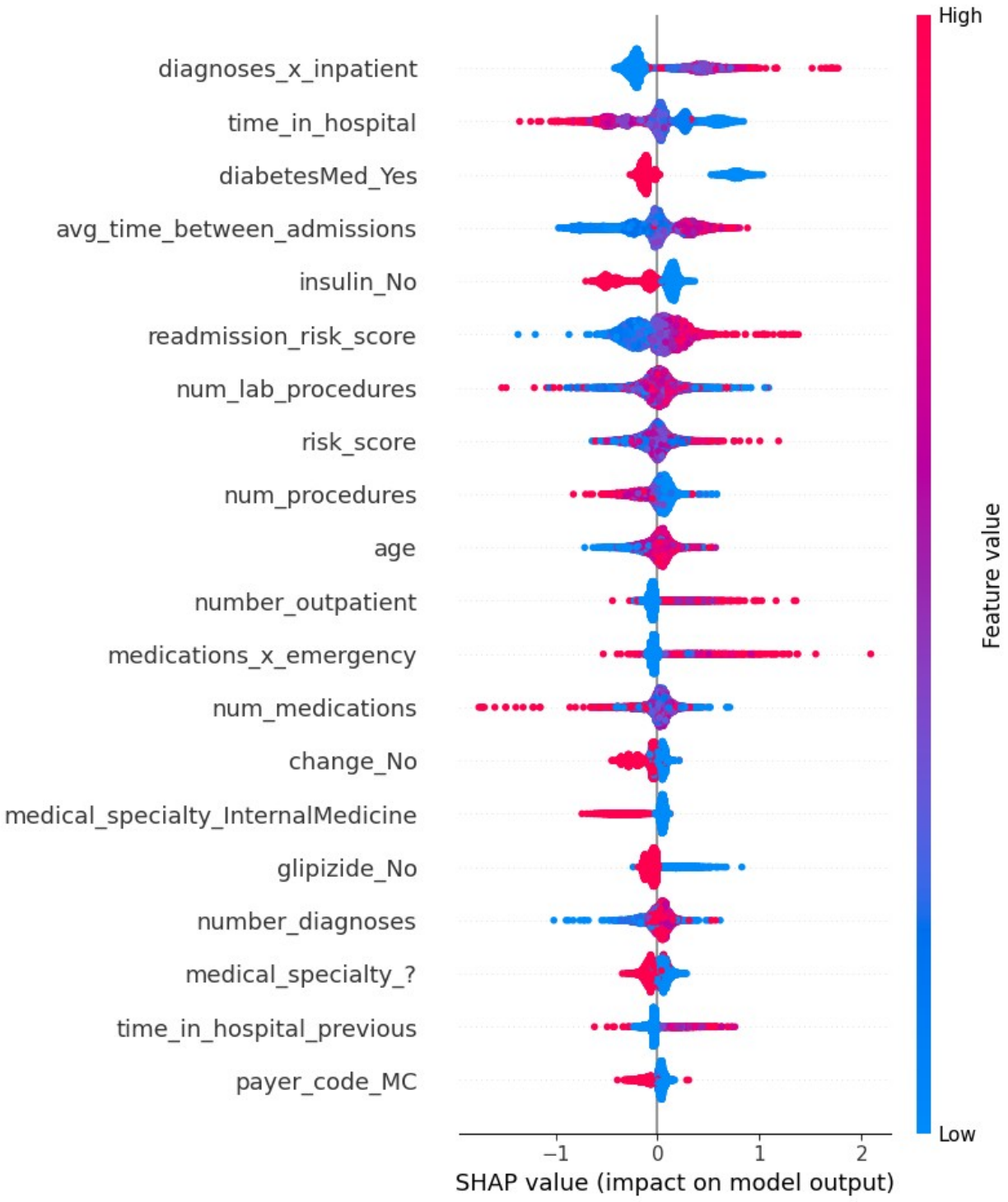
```
/usr/local/lib/python3.10/dist-packages/dask/dataframe/__init__.py:42:
FutureWarning:
Dask dataframe query planning is disabled because dask-expr is not
```

```
installed.

You can install it with `pip install dask[dataframe]` or `conda
install dask`.
This will raise in a future version.

  warnings.warn(msg, FutureWarning)

[LightGBM] [Info] Number of positive: 10859, number of negative: 10859
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead
of testing was 0.010374 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 1305
[LightGBM] [Info] Number of data points in the train set: 21718,
number of used features: 56
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.500000 ->
initscore=0.000000
LightGBM Classification Report:
              precision    recall  f1-score   support

           0       0.64      0.69      0.66      2731
           1       0.58      0.53      0.56      2269

    accuracy                           0.62      5000
   macro avg       0.61      0.61      0.61      5000
weighted avg       0.61      0.62      0.61      5000

LightGBM ROC AUC Score: 0.6543978114587601

from sklearn.ensemble import StackingClassifier
from sklearn.linear_model import LogisticRegression

# Create base models
base_models = [
    ('xgb', XGBClassifier(use_label_encoder=False,
eval_metric='logloss', random_state=42)),
    ('lgbm', LGBMClassifier(random_state=42))
]

# Create stacking model
stacking_model = StackingClassifier(estimators=base_models,
final_estimator=LogisticRegression())
stacking_model.fit(X_resampled, y_resampled)

# Evaluate stacking model
y_pred_stack = stacking_model.predict(X_test)
y_prob_stack = stacking_model.predict_proba(X_test)[:, 1]

print("Stacking Model Classification Report:\n",
classification_report(y_test, y_pred_stack))
```

```
print("Stacking Model ROC AUC Score:", roc_auc_score(y_test,
y_prob_stack))
```

```
[LightGBM] [Info] Number of positive: 10859, number of negative: 10859
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead
of testing was 0.009514 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 1305
[LightGBM] [Info] Number of data points in the train set: 21718,
number of used features: 56
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.500000 ->
initscore=0.000000
[LightGBM] [Info] Number of positive: 8687, number of negative: 8687
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead
of testing was 0.007916 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 1308
[LightGBM] [Info] Number of data points in the train set: 17374,
number of used features: 56
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.500000 ->
initscore=0.000000
[LightGBM] [Info] Number of positive: 8687, number of negative: 8687
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead
of testing was 0.007789 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 1311
[LightGBM] [Info] Number of data points in the train set: 17374,
number of used features: 56
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.500000 ->
initscore=0.000000
[LightGBM] [Info] Number of positive: 8687, number of negative: 8687
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead
of testing was 0.007584 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 1298
[LightGBM] [Info] Number of data points in the train set: 17374,
number of used features: 56
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.500000 ->
initscore=0.000000
[LightGBM] [Info] Number of positive: 8688, number of negative: 8687
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead
of testing was 0.007670 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 1311
[LightGBM] [Info] Number of data points in the train set: 17375,
```

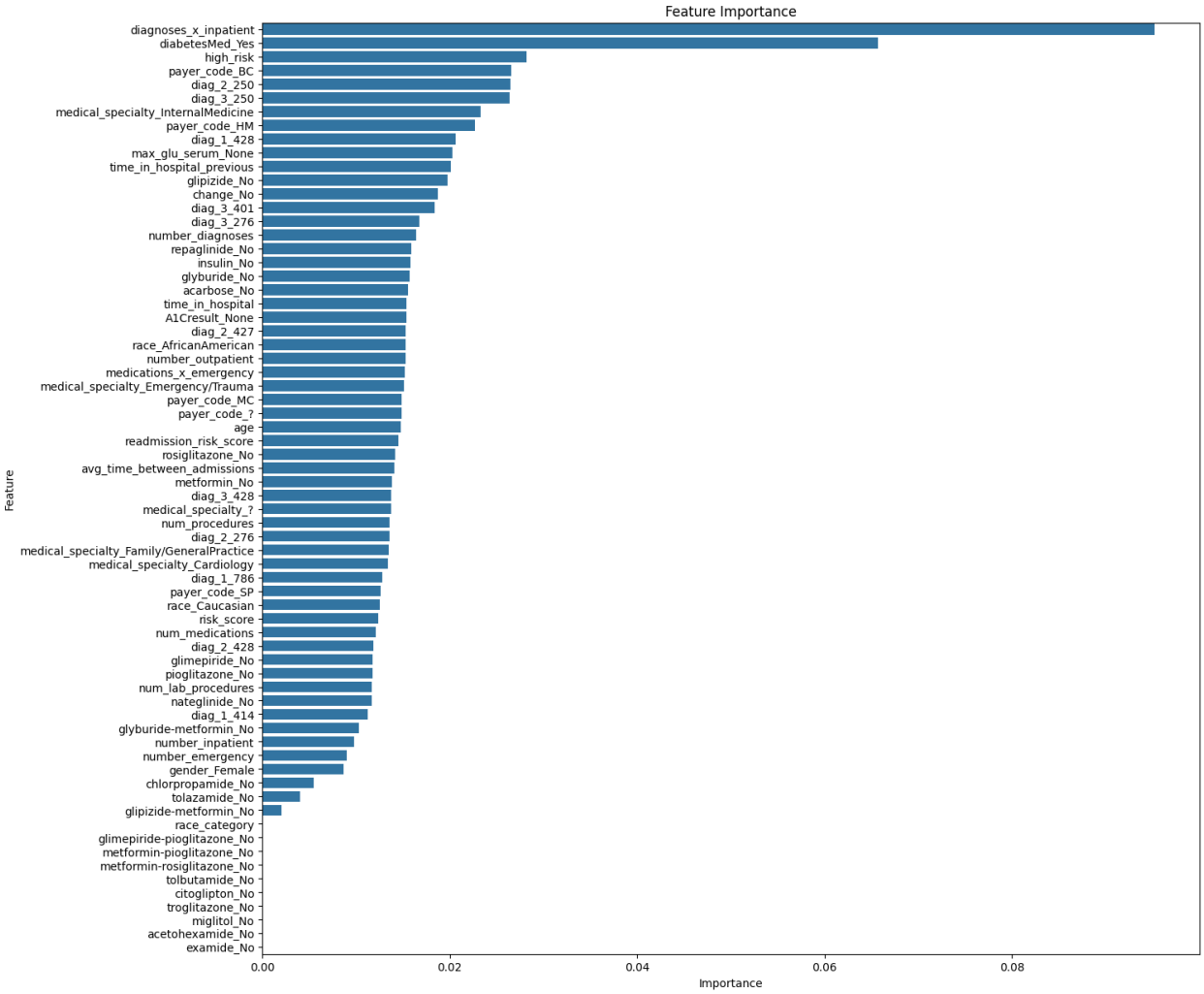
```
number of used features: 56
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.500029 ->
initscore=0.000115
[LightGBM] [Info] Start training from score 0.000115
[LightGBM] [Info] Number of positive: 8687, number of negative: 8688
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead
of testing was 0.007401 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 822
[LightGBM] [Info] Number of data points in the train set: 17375,
number of used features: 56
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.499971 ->
initscore=-0.000115
[LightGBM] [Info] Start training from score -0.000115
Stacking Model Classification Report:
              precision    recall  f1-score   support

           0       0.65      0.64      0.64      2731
           1       0.57      0.58      0.57      2269

    accuracy                           0.61      5000
   macro avg       0.61      0.61      0.61      5000
weighted avg       0.61      0.61      0.61      5000

Stacking Model ROC AUC Score: 0.6543414906048264

import shap

explainer = shap.TreeExplainer(xgb_model)
shap_values = explainer.shap_values(X_test)

# Summary plot
shap.summary_plot(shap_values, X_test)

# Force plot for a single prediction
shap.force_plot(explainer.expected_value, shap_values[0],
X_test.iloc[0])
```




```
<shap.plots._force.AdditiveForceVisualizer at 0x7960e53a0ca0>
```

```
import seaborn as sns

# Feature importance from XGBoost
importances = xgb_model.feature_importances_
```

```
feature_names = X_train.columns
importance_df = pd.DataFrame({'Feature': feature_names, 'Importance':
importances}).sort_values(by='Importance', ascending=False)

# Plot feature importance
plt.figure(figsize=(15, 15))
sns.barplot(data=importance_df, x='Importance', y='Feature')
plt.title('Feature Importance')
plt.show()
```




```
from sklearn.model_selection import cross_val_score

# Perform 5-fold cross-validation
scores = cross_val_score(xgb_model, X_resampled, y_resampled, cv=5,
scoring='roc_auc')
print("Cross-Validated ROC AUC:", scores.mean())

Cross-Validated ROC AUC: 0.6487638686502117
```