# Machine Learning Report

## Ready to be discharged: Examining Hospital Readmissions

- André Lopes (20230570@novaims.unl.pt)

- Adriana Costinha (20230567@novaims.unl.pt)

- Helena Mashayekhi(20230561@novaims.unl.pt)

- Lía García (20230586@novaims.unl.pt)

- Luís Queiroz (20230584@novaims.unl.pt)

- Diana Silva (20230586@novaims.unl.pt)

**NOVA IMS**

## Contents

## Abstract

This study was done to develop a predictive model for the Hospital readmissions of patients, in order to answer our 2 questions, if (1) it is possible to accurately classify new patients as readmitted, or not readmitted to the hospital and (2) if it is possible to see in what timeframe they are readmitted. The main goal of this end-to-end machine learning project was to reach the best possible predictions on a test dataset. To accomplish that, we explored the data and performed data pre-processing. At this stage, we did cardinality checks, missing value imputations, feature engineering, and finally feature selection. We obtained satisfactory results on our binary classification baseline model, using several classifiers: Random Forest, MLP, AdaBoost, Gradient Boost and Categorical Naive Bayes classifiers. And also obtained satisfactory results on our multiclass classification, using AdaBoost, Ridge, Random Forest and MLP classifiers. Finally, the best result for the binary classification was obtained with an Adaboost with a F1 score of 0.67 on the Training dataset, and 0.3 on the Validation dataset. For our multiclass classification we obtained with a Random Forest classifier a weighted F1 score of 0.57 on the Validation dataset, and 0.71 on the Training dataset.

Keywords: Machine Learning, Predictive Modelling, Binary Classification, Multiclass Classification, Supervised Learning.

## 1 Introduction

Hospital readmissions are a crucial concern in the healthcare sector, serving as a key indicator of the quality of care and a substantial driver of escalating costs. The rehospitalization of patients shortly after discharge not only suggests potential gaps in care but also imposes a significant financial burden on the healthcare system. Of particular note is the impact of readmissions among diabetic patients, which substantially contributes to the overall healthcare expenditure. The overall goals of the project include two main objectives: a binary classification and a multiclass classification.

The aim of the study is to develop prediction models based on a training data set of 71,237 observations. For the binary classification, the target variable is ”readmitted_binary”, indicating ”yes” for readmission and ”no” for non-readmission. In the multiclass classification, the outcome variable is “readmitted_multiclass”, with patients categorized as ”no”, ”<30 days” or ”>30 days” for readmission.

The dataset is composed of socio-demographic features (*i.e: country, patient id, race, gender, age, weight, payer code*) and medical features (*i.e: outpatient visits, emergency visits, inpatient visits*).

This study’s objective is to obtain the best possible performance on a test dataset, by implementing the binary classification and the multiclass classification on 30.531 unseen records.

## 2 Data Exploration and Preprocessing

The first step undertaken, right before splitting the training data, was importing the train.csv file into a validation dataset and a training dataset with a proportion of 30% for validation and 70% for training. All subsequent exploration was conducted on the training dataset.

### 2.1 Data Exploration

#### 2.1.1 Analyzing Feature Categories

In our initial exploration, our focus was on examining the characteristics of the features at our disposal. We sought to categorize these features into two main types: categorical and numerical. This foundational step laid the groundwork for a more in-depth analysis of the dataset.

From the analysis presented in Table A.1, it appeared, at first glance, that the features were encoded with the correct data types.

#### 2.1.2 Initial Examination of the Training Dataset

Here, our objective was to take a quick look at the dataset, check for any unusual behaviors in certain columns, and make interesting observations for further investigation. All the notes we found are summarized in Table 1.

| **Column name** | **Observations** |
|---|---|
| patient_id | Repeated patient IDs |
| country | Only has one value in this column: ”USA” |

| Column name | Observations |
|---|---|
| race | A weird value ”?” was found |
| gender | A weird value ”Unknown/Invalid” was found |
| weight | Many weird values ”?” were found |
| payer_code | Many weird values ”?” were found |
| admission_type | Categorical feature with high cardinality |
| medical_specialty | Categorical feature with high cardinality and many weird values ”?” were found |
| discharge_disposition | Categorical feature with high cardinality |
| primary_diagnosis | Categorical feature with high cardinality |
| secondary_diagnosis | Categorical feature with high cardinality |
| additional_diagnosis | Categorical feature with high cardinality |
| medication | Values exhibit unusual formatting |

Table 1: Observations about the data set.

Upon examining Table 1, it became evident that the categorical features pose more challenges compared to the numerical features.

### 2.1.3 Features with Missing Data

To conclude our preliminary examination of the dataset, we compiled a table containing all the columns with missing values, Table A.2.

The prevalence of missing values in the columns for glucose test results and A1C test results is noteworthy. Importantly, these missing values do not signify data collection errors or oversights. Rather, they explicitly indicate instances where the associated tests were not conducted or measured during the patient’s healthcare interactions. It was crucial to acknowledge and address this aspect, in both the exploration and preprocessing phases, to ensure a nuanced and accurate understanding of the dataset.

### 2.1.4 Analysis of the ”Patient_id” Feature

During the data exploration, we discovered that the ”patient_id” column contains repeated values. Utilizing this information, we established a new column named ”patient_id_count”, more details on the Section 2.2. Upon further investigation, it became apparent, as shown in Figure A.1, that, on average, patients who have not been readmitted appear fewer times in the data set compared to those who have been readmitted.

### 2.1.5 Exploration of Numerical Features

The next step in our data exploration involved examining the numerical columns. Our main focus here was:

- Checking the numerical data distributions.
- Identifying outliers.

- Investigating correlations between numerical features.
- Visualizing the impact of each feature on the target variable.

Examining Figure A.3 revealed several interesting observations. Firstly, the majority of patients in our dataset made zero visits in the previous year, while a few patients made more than one visit. Another intriguing aspect was the distribution of the number of lab tests, which showed a bin around zero that appeared to be outside the typical distribution pattern. Additionally, the distribution of the number of medications appeared to be normal, but slightly right-skewed, potentially influenced by outliers.

Now looking at the box plots, Figure A.4, our initial observation was that the visits feature marked all patients with more than zero visits as outliers. However, in this context, these outliers are not anomalies, but rather a reflection of the inherent nature of the data. In contrast, the remaining columns exhibited some points identified as outliers, but these instances are more reasonable. Furthermore, the percentage of data points classified as outliers in these columns was significantly lower.

Upon reviewing the heatmap, Figure A.9 in appendix, it is evident that there was a lack of strong correlation among the numerical features.

Now, examining the relationship between the numerical features and the ”readmitted_binary” column, we plotted the mean values for each readmission group, as shown in Figure A.12. The initial observation revealed that inpatient, outpatient, and emergency visits are correlated with the readmitted_binary column. On average, readmitted patients tended to make more visits, particularly in terms of inpatient visits. This reinforced the notion that the outliers identified in the box plots were not anomalies, but rather individuals who made more visits, emphasizing the importance of retaining this information. The remaining features, at least from this perspective, did not exhibit a significant impact on the target variable.

### 2.1.6 Exploration of Categorical Features

Categorical features could be divided into two distinct groups based on their cardinality (Table A.3). This classification was crucial as it influenced the preprocessing and exploration approaches applied to these features.

Our analysis began with a scrutiny of bar plots in the appendix (Figure A.5) to assess category coherence in low cardinality features. Identified issues included unconventional categories in certain features and the presence of missing values. To address these concerns, we implemented modifications outlined in the preprocessing section (Section 2.2). These adjustments aimed to enhance data quality for subsequent analyses.

Having successfully addressed the identified issues, we proceeded to regenerate the bar plots. The updated visualization is presented in Figure A.11.

Our next step involved analysing the relationship between the features and the target variable. To achieve this, we aimed to plot the proportions of each class for a more comprehensive understanding, these plots can be seen in the Figure A.6.

From this perspective, it seemed that none of the low cardinality categorical features showed a strong correlation with the target variable. The instances where bars deviated from established patterns could be attributed to their exceptionally low frequency, rendering them unrepresentative in the overall distribution.

Navigating high cardinality features posed challenges akin to those encountered with their low cardinality counterparts. To tackle the intricacies associated with numerous categories, as outlined in the cardinality table (Table A.3), our objective was to improve the visualization of these features, while simultaneously reducing their cardinality. The methodology to address these challenges was elucidated in section 2.2.

Following the completion of the preprocessing steps, these issues were successfully resolved. We could now proceed with plotting the bar plots for each feature (Figure A.10).

Upon inspecting the plots, it became evident that our dataset was heavily skewed towards specific categories in each feature. Another observation revealed redundancy among the primary, secondary, and additional diagnosis, as they conveyed nearly identical information. Therefore, considering the consistency of information across these features, it appeared practical to retain only one of them, minimizing redundancy without sacrificing informative content.

The next step was to try to visualize the influence this features had in the target variable. This plot can be found in appendix (Figure A.7)

Upon scrutinizing these plots, notable observations emerge. As anticipated, patients who expired did not exhibit readmission. Additionally, the ”discharge_disposition” feature appeared to exert some influence on the readmission proportions. For the remaining features, while there was variation in readmission proportions across categories, no extraordinary or abnormal patterns were evident.

It is crucial to highlight that bivariate analyses were conducted for the multiclass variable as well, as illustrated in Figures A.8, A.13, and A.2. However, it is noteworthy that the conclusions drawn and the observed behaviors of features closely mirror those identified in the binary predictions.

## 2.2 Preprocessing

In the process of data exploration and feature engineering for our project, we encountered several challenges with columns that were important for our analysis. In the following sections, we explain more of the challenges that we encountered and how we dealt with each one.

To facilitate comprehension, we split the explanation into three parts, aligning with the facets explored during the analysis: Patient ID Column, Numerical Features and Categorical Features (Low Cardinality and High Cardinality).

### 2.2.1 Patient_id column

Building upon insights from the exploration phase, where ”patient_id” repetition was observed, our preprocessing strategy involved introducing a new column termed ”patient_id_count”. This column was designed to capture the frequency encoding of the original ”patient_id” column, providing a nuanced representation of the patient ID occurrences in the dataset.

### 2.2.2 Numerical Features

The preprocessing of numerical features experienced only minimal adjustments, primarily focusing on the application of the Interquartile Range (IQR) method to cap outliers in specific columns (”length_of_stay_in_hospital”, ”number_lab_tests”, ”non_lab_procedures”, ”number_of_medications”, ”number_diagnoses”). This measure was chosen judiciously, recognizing that certain data points initially identified as outliers during exploration were, in fact, intrinsic to the nature of the data. By implementing the IQR method, we aimed to ensure the robustness of these numerical features while preserving the inherent characteristics of the dataset.

### 2.2.3 Categorical Features

In handling low cardinality features, our initial focus involved addressing missing values through a structured imputation process. Notably, for certain columns such as ”glucose_test_result” and ”a1c_test_result”, missing values were indicative of ”Not measured” and were therefore more intentionally mapped.

Additionally, in instances where distinct categories were observed, such as ”Unknown/invalid” in the gender column, we decided to replace them with the mode value. Similarly, for the race column, instances of ”?” were substituted with the mode, contributing to a more coherent, and standardized representation of the categorical data.

In our approach to handling high cardinality categorical features, we initially addressed missing values. Some features with missing values were filled using the mode value, while others with rows containing ”?” were first converted to missing values and subsequently imputed with the mode.

The second challenge involved managing the high cardinality itself. To tackle this, we introduced a frequency threshold, systematically mapping categories with frequencies below the specified threshold to an ”Other” category. This strategic intervention not only refined the representation of high cardinality features but also yielded a dataset that was more manageable and interpretable.

In certain instances, specific columns required preliminary attention before the preceding two steps could be executed. For example, in the case of the diagnosis columns, an initial mapping of the ICD-9 codes to diagnosis groups was crucial. To perform this mapping, we referenced the guidelines provided by the World Health Organization (1977), ensuring a comprehensive and standardized representation of the diagnostic information.

After exploring various approaches for managing the ”medication” column, we opted for an efficient solution. Our decision involved creating a new column named ”medication_length,” which quantified the quantity of the medication. This approach not only simplified the representation of medication information, but also offered a meaningful metric that could be readily utilized for further analysis.

### 2.2.4 Preprocessing pipelines

Pipeline 1 focused on addressing missing and inconsistent data. ”FillNAWithNotMeasured” filled missing values with ”Not Measured”, ensuring a standardized representation. The ”ReplaceQM” step replaced question marks with NaNs, contributing to data uniformity. The ”fill_payer_code_na”, as the name suggest, input the missing values in the payer code feature with ”None”. The ”Replace_unknown” step specifically handled the ”Unknown/invalid” category in the gender feature, substituting it with the mode value. Lastly, ”RemoveHighMissingColumns” systematically eliminated features with over 30% missing values, streamlining the dataset.

In Pipeline 2, two distinct pipelines, namely ”pipeline_2” and ”pipeline_2_m”, addressed the task of handling missing values. Pipeline_2 utilized the strategic approach of filling missing values with the mode, while pipeline_2_m took a different route by incorporating the distribution of values for filling missing entries. This nuanced separation was a result of empirical findings: pipeline_2 demonstrated superior performance in binary prediction scenarios, while pipeline_2_m yielded better results in multiclass prediction scenarios. This tailored approach underscored the importance of adapting the handling of missing data based on the specific nature of the prediction task at hand.

Pipeline 3 encompassed a comprehensive series of transformations aimed at enhancing feature representation and relevance. The initial step, ”MapCategories”, strategically mapped high-cardinality features to a condensed set. In this pipeline, we additionally converted ICD-9 codes to their respective diagnostic groups while concurrently reducing the number of diagnoses. The introduction of the ”ValueCountTransformer” generated a new informative column, ”patient_id_count,” as previously explained. Additionally, the age column underwent a transformation, transitioning from an categorical feature to a numerical representation by assigning a random age within the given interval. The pipeline further addressed outliers, as exemplified earlier, and concluded by excluding columns marked as irrelevant: ”encounter_id”, ”country”, ”secondary_diagnosis”, ”additional_diagnosis”, and ”medication.” This intricate sequence of transformations aimed to optimize feature quality and relevance, fostering improved model performance.

## 3 Binary Classification

In this section, we implemented and evaluated various classification models, focusing on the F1 score for the ”yes” class. Our objective was to enhance the model's ability to predict whether a patient would be readmitted within 30 days. We employed techniques such as random search, cross-validation, and the holdout method to thoroughly assess and compare the performance of these models.

### 3.1 Scaling, Encoding, and Oversampling

In this dataset, we observed a significant class imbalance in the target variable ”readmitted_binary”. The proportion of individuals classified as ”no” (not readmitted) substantially outweighs those classified as ”Yes” (readmitted).

To address this issue, we utilized the oversample function, designed to handle class imbalance by randomly oversampling the minority class. It's important to note that this process was exclusively applied to the training data, ensuring the preservation of the originality of the validation data.

Also, the numeric features exhibit different scales. To address this, we employed the ”scale_data” function to mitigate the impact of varying scales among numeric features, thereby preventing specific features from overpowering others. The `scale_data` function normalizes numeric features using Min-Max scaling ('scaler_type' = 0). It took training, validation, and test dataframes as input, along with a list of columns to scale. This method aided in enhancing the performance of models sensitive to the magnitude of input features.

Since many machine learning models can only work with numerical values, another procedure was taken to convert some categorical variables into numerical ones. The `label_encode_dataframe` function performs label encoding on specified categorical columns for training, validation, and test dataframes.

It's important to note that the chosen scaler and encoder were determined through an iterative process, and we selected the ones that yielded the best results.

## 3.2 Feature selection

To obtain insights into the significance of different features, we calculated the importance of each feature by employing a Random Forest model.

Based on the results obtained from the Random Forest model, we select the top 10 most important features. The selected features are as follows: ”patient_id_count”, ”average_pulse_bpm”, ”number_lab_tests”, ”age”, ”number_of_medications”, ”length_of_stay_in_hospital”, ”primary_diagnosis”, ”inpatient_visits_in_previous_year”, ”discharge_disposition”, ”payer_code.

## 3.3 Modelling

During the modeling process, a decision had to be made: compare the models in their default state or optimize their performance through hyperparameter tuning before making comparisons.

Hyperparameter tuning involves systematically adjusting the configuration of each model to find the set of parameters that maximizes performance metrics. After optimization, the models are compared, enabling us to make well-informed decisions about their overall performance. Therefore, we decided that it would be more prudent to tune the hyperparameters first and then compare the models.

When tuning model parameters, we considered several approaches. One option was the hold-out method, where we manually tweak each hyperparameter and observe its impact. However, this method can be labor-intensive. Another approach was using cross-validation, where we systematically tried a range of parameter values to understand their influence on the model. While this provides comprehensive insights, it can be computationally expensive, especially when dealing with multiple models.

To address the computational challenge, we explored grid search, a method that exhaustively tests all possible combinations of specified parameters to identify the best-performing set. Although effective, grid search becomes impractical when dealing with numerous parameter combinations.

To balance computational efficiency and effectiveness, we opted for random search. This method randomly samples a defined number of parameter combinations, offering greater control over computational expenses. While random search doesn’t guarantee finding the optimal solution, it generally yields good parameter sets for model comparison.

In summary, we chose to implement the random search for parameter tuning across all the models we wanted to compare, aiming for a balance between computational efficiency and obtaining satisfactory model performance for meaningful comparisons.

The models under consideration were **Random Forest (RF)**, **MLP Classifier (MLP)**, **AdaBoost**, **Gradient Boost (GB)**, and **Categorical Naive Bayes (NB)**.

## 3.4 Model Comparison

To compare the various models, we used a cross-validation with five folds for each model. After that, we generated a table (table 2) with the average values for selected metrics (such as F1 score, precision, and recall) to compare the models.

| Model | Train F1 | Test F1 | Train Precision | Test Precision | Train Recall | Test Recall |
|---|---|---|---|---|---|---|
| RF | 0.71 | 0.313 | 0.686 | 0.216 | 0.736 | 0.568 |
| AdaBoost | 0.668 | 0.305 | 0.683 | 0.229 | 0.654 | 0.457 |
| MLP | 0.694 | 0.305 | 0.677 | 0.211 | 0.713 | 0.548 |
| NB | 0.534 | 0.217 | 0.58 | 0.141 | 0.495 | 0.475 |
| GB | 0.879 | 0.256 | 0.866 | 0.219 | 0.892 | 0.31 |

Table 2: Performance Metrics for Different Models.

After analyzing and comparing the models, we identified that AdaBoost, Random Forest and MLP Classifier exhibited similar and good metrics. For these models, we employed the hold-out method and generated a more comprehensive report for each one. This detailed report allowed us to thoroughly assess the performance of each model.

Based on those results, we identified that the MLP Classifier and Random Forest models exhibited higher overall F1 scores. However, the AdaBoost model showcased a more balanced performance with closely aligned recall and precision metrics. This equilibrium in recall and precision suggested that AdaBoost may provide a more robust and reliable classification, making it a compelling choice, even in situations where the F1 score alone might favor other models. The decision to select the most suitable model should consider the specific requirements and trade-offs between precision and recall, with AdaBoost standing out for its balanced performance.

### 3.5 Hyperparameter Optimization for the Selected Model

Once we had selected AdaBoost as our final model, we performed hyperparameter tuning to optimize its performance even more. We observed that as the number of estimators increased, the performance of the model improved. However, beyond 200 estimators, the performance gains became marginal. Consequently, the optimal configuration for the AdaBoost model was determined through hyperparameter tuning, culminating in the selection of a final model that utilized 200 estimators. With this model, we obtained an F1 score of 0.67 for the 'Yes' class on the training data and 0.30 on the validation data.

## 4 Multiclass Classification

### 4.1 Scaling, Encoding and Oversampling

For the multiclass classification, we decided to apply the same pre-processing already talked about in the binary section, with one caveat. Instead of filling the missing values with the mode, we tried filling the missing values with respects to the distribution of the feature we would impute values into. This ended up giving us a slightly better result for the models, one could call this difference in results negligible, but the reason we kept it was mainly to give a glimpse of another possible way to impute values.

The concept here was to obtain the probabilities of each possible non-missing value occurring in a feature and select a value for imputation from the set of values based on their respective probabilities.

For the oversampling part, a different approach was taken, in regards to the binary section. In this case, we believed the RandomOverSampler to work better due to the nature of the data we were working with. SMOTE is likely to introduce additional complexity that could lead to overfitting. In contrast, by duplicating random samples from the minority class, we could maintain the original data distribution. Even though we were repeating samples, it might have been providing a more realistic training scenario for our model.

## 4.2 Feature Selection

To feed our model with more specific and important data, we employed the same feature selection approach as in the binary classification model. Knowing that, identifying key factors that contribute to hospital readmissions was a crucial step, we implemented a Random Forest model to evaluate the importance of various features. In this multiclass section we opted for selecting the 15 most important variables. This helped keeping the model simple and more objective.

Based on the results we obtained we selected the following 15 features: ”average_pulse_bpm”, ”number_lab_tests”, ”age”, ”patient_id_count”, ”number_of_medications”, ”length_of_stay_in_hospital”, ”primary_diagnosis”, ”payer_code”, ”discharge_disposition”, ”number_diagnoses”, ”non_lab_procedures”, ”inpatient_visits_in_previous_yes”, ”medication_length”, ”admission_type”, ”admission_source”.

## 4.3 Model Tuning

Just as we did for the binary classification, we also chose to implement the random search to find the most effective combination of parameters for our model based on the F1 weighted score. The models we chose to compare were **Adaboost**, **Ridge**, **Random Forest (RF)**, and **MLP Classifier (MLP)**.

## 4.4 Model Comparison

To assess and contrast various models, we employed a cross-validation methodology incorporating five folds for each model, mirroring our approach in binary classification. In table 3, it is possible to observe the average values for chosen metrics, such as F1 score, precision and recall facilitating the comparison between models.

| Model | Train F1 | Test F1 | Train Precision | Test Precision | Train Recall | Test Recall |
|---|---|---|---|---|---|---|
| RF | 0.86 | 0.599 | 0.87 | 0.594 | 0.858 | 0.615 |
| AdaBoost | 0.484 | 0.556 | 0.503 | 0.578 | 0.502 | 0.564 |
| MLP | 0.5 | 0.558 | 0.508 | 0.587 | 0.514 | 0.556 |
| Ridge | 0.446 | 0.504 | 0.458 | 0.54 | 0.464 | 0.506 |

Table 3: Performance Metrics for Different Models.

Upon further analysis, it became evident that Random Forest and MLP classifier exhibited comparable and commendable metrics. However, these models are prone to overfitting, as it was apparent that the average difference in the F1 score between training and test data was approximately 0.41.

One possible and probable reason for the over fitting problem could be the complexity of the models used, particularly those with numerous hyper-parameters like MLP classifier or Gradient Boost. When these models are optimally tuned, they may become too complex, fitting too closely to the training data and struggling to generalize effectively to new data. For example, MLP classifier with an excessive number of layers or neurons could easily result in overfitting, where the model performs exceptionally well on the training data, it struggles to generalize unseen data due to its overly complex nature.

Hence, we opted to tackle this issue through manual adjustment of each hyperparameter, leveraging the values acquired from the random search as a baseline. Consequently, our ultimate parameter configuration entailed a maximum tree depth of 12 and 200 estimators.

With that, we achieved a F1 weighted score of 0.57 for the validation which indicates moderate effectiveness of the model in predicting hospital readmissions and managed to reduce over fitting, as the difference between training set and test set is lower due to the f1 score for the training set being now 0.71. It suggested that the model was better than random guessing, but still had significant room for improvement, particularly in predicting readmissions within the ”<30 days” category.

## 5 Conclusion

In this machine learning project, we aimed to tackle two classification problems: the binary problem which is to predict the readmission of patients, and the multiclass problem which is to see within what time-frame they were readmitted in. Beginning with a detailed examination of the dataset and its features we identified important insights, patterns and relations.

The preprocessing phase involved addressing missing data, handling outliers, handling low cardinality and transforming categorical features for optimal representation. Additionally, two new features, namely ”patient_id_count” and ”medication_length” were created.

In feature selection we calculated the feature importance based on a random forest and ended up selecting the top ten most important features for the binary section and the top fifteen features for the multiclass part of the project.

Moving to the model training and evaluation phase, we systematically compared the performances of diverse algorithms through cross validation, encompassing Random Forest, MLP Classifier, AdaBoost, Gradient Boost, and Categorical Naive Bayes. Rigorous evaluation metrics, including F1 score, precision, and recall, facilitated a nuanced understanding of each model’s strengths and limitations.

In summation, this project not only achieved its primary objectives but also laid the groundwork for continued exploration and improvement in the realm of healthcare predictive analytics.

## 6 References

World Health Organization, Manual of the International Classification of Diseases, Injuries, and Causes of Death, Ninth Revision. Geneva: World Health Organization, 1977.

## A Appendix

| Feature type | Columns |
|---|---|
| Categorical | ’country’, ’race’, ’gender’, ’age’, ’weight’, ’payer_code’, ’admission_type’, ’medical_specialty’, ’discharge_disposition’, ’admission_source’, ’primary_diagnosis’, ’secondary_diagnosis’, ’additional_diagnosis’, ’number_diagnoses’, ’glucose_test_result’, ’a1c_test_result’, ’change_in_meds_during_hospitalization’, ’prescribed_diabetes_meds’, ’medication’ |
| Numerical | ’outpatient_visits_in_previous_year’, ’emergency_visits_in_previous_year’, ’inpatient_visits_in_previous_year’, ’average_pulse_bpm’, ’length_of_stay_in_hospital’, ’number_lab_tests’, ’non_lab_procedures’, ’number_of_medications’ |
| Identifier ID’s | ’encounter_id’, ’patient _id’ |

Table A.1: Feature types.

| Column name | Missing values |
|---|---|
| race | 2460 |
| age | 2511 |
| admission_type | 2595 |
| admission_source | 3343 |
| glucose_test_result | 47281 |
| a1c_test_result | 41570 |

Table A.2: Columns with missing values.

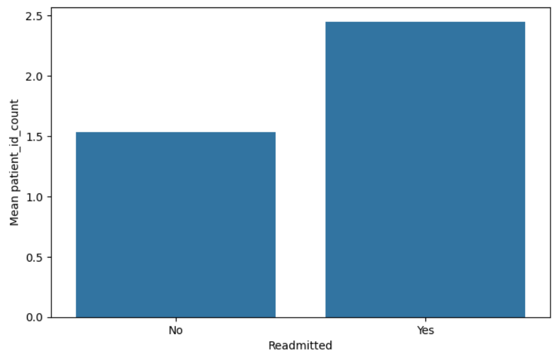

Figure A.1: Patient_id_count vs readmitted_binary

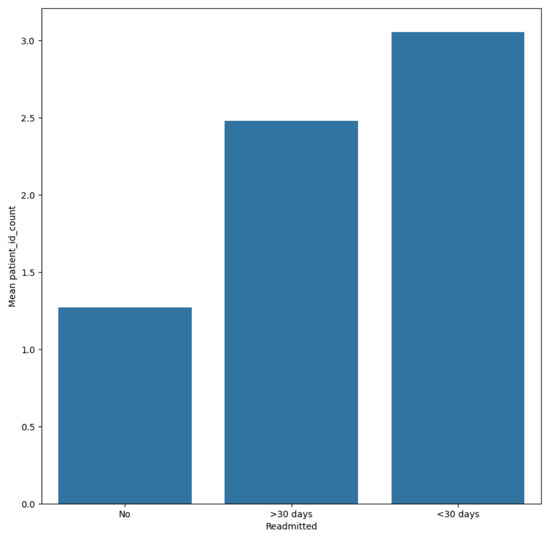

Figure A.2: Patient_id_count vs readmitted_multiclass.

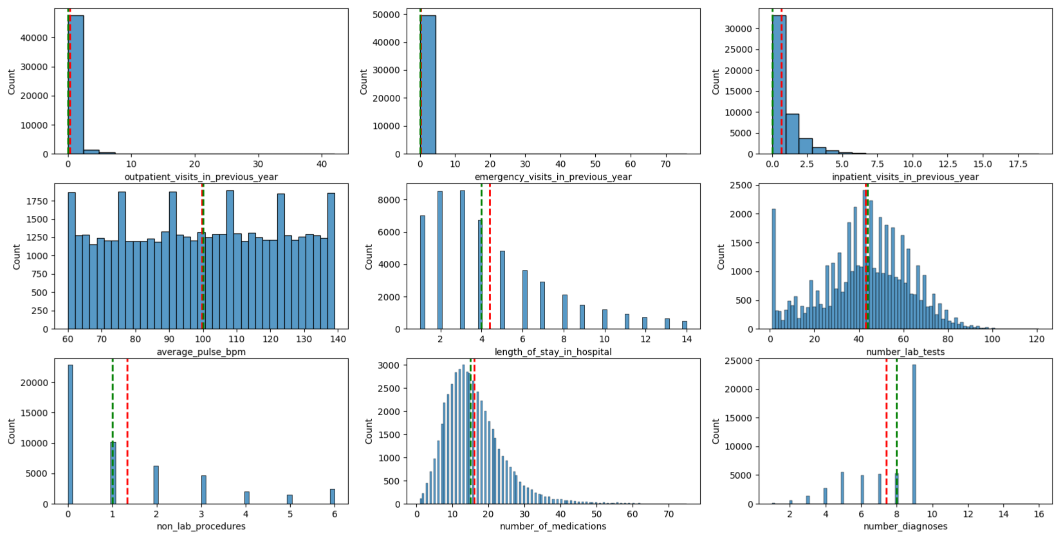

Figure A.3: Numerical features distribution.

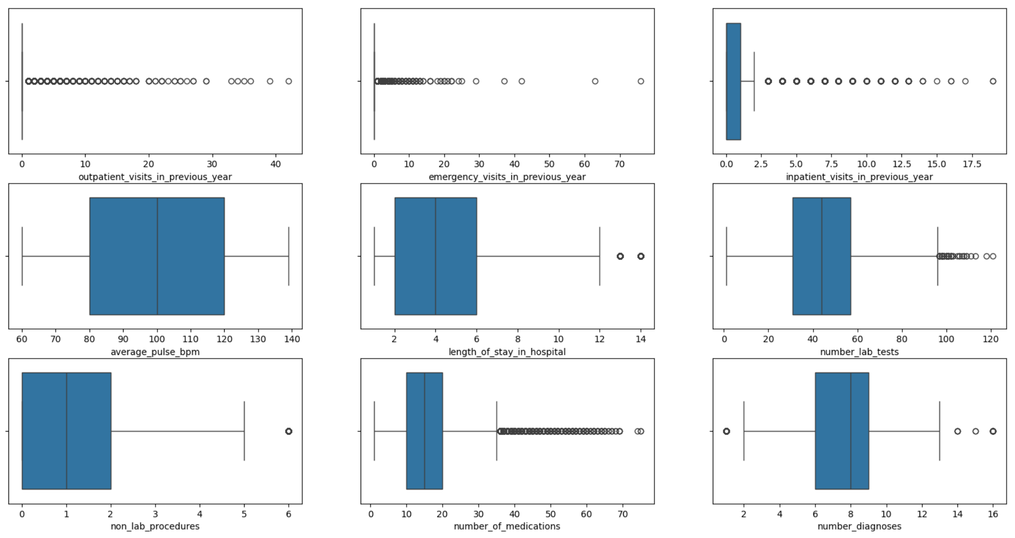

Figure A.4: Box plots of numerical features.

| **Column Name** | **Cardinality** | **Cardinality Group** |
|---|---|---|
| country | 1 | Low |
| race | 6 | Low |
| gender | 3 | Low |
| age | 10 | Low |
| weight | 10 | Low |
| admission_type | 7 | Low |
| glucose_test_result | 3 | Low |
| a1c_test_result | 3 | Low |
| change_in_meds_during_hospitalization | 2 | Low |
| prescribed_diabetes_meds | 2 | Low |
| payer_code | 18 | High |
| medical_specialty | 68 | High |
| discharge_disposition | 24 | High |
| admission_source | 16 | High |
| primary_diagnosis | 648 | High |
| secondary_diagnosis | 657 | High |
| additional_diagnosis | 698 | High |
| medication | 275 | High |

Table A.3: Cardinality group distribution per feature.

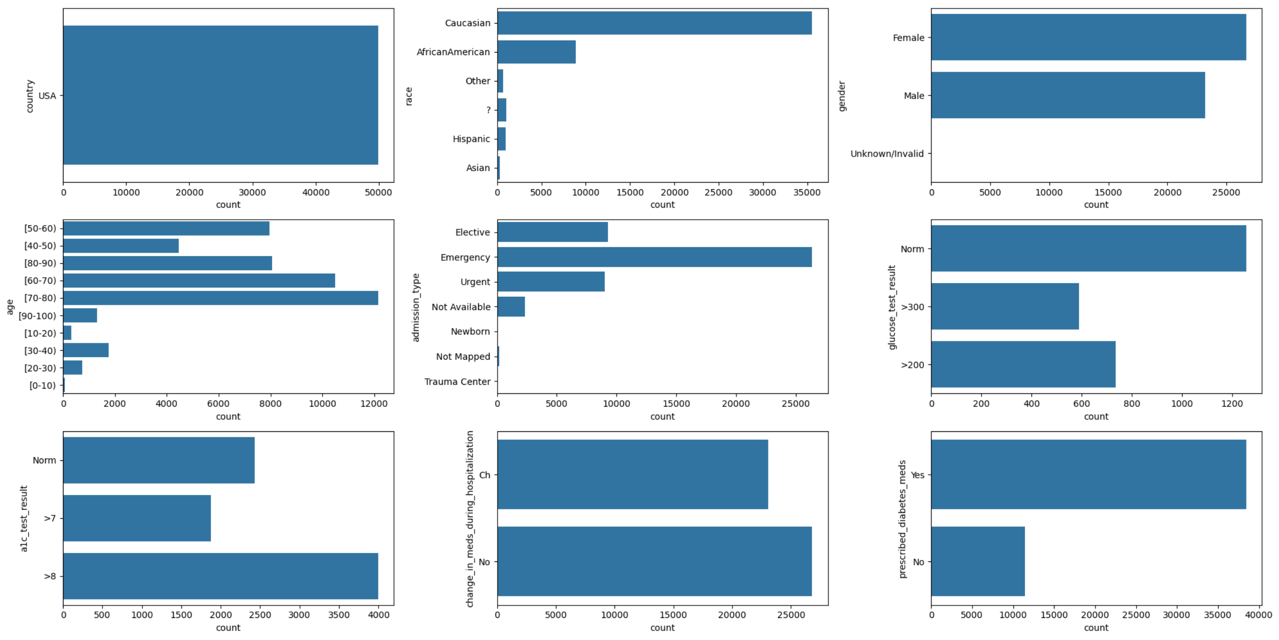

Figure A.5: Bar plots illustrating low cardinality categorical features.

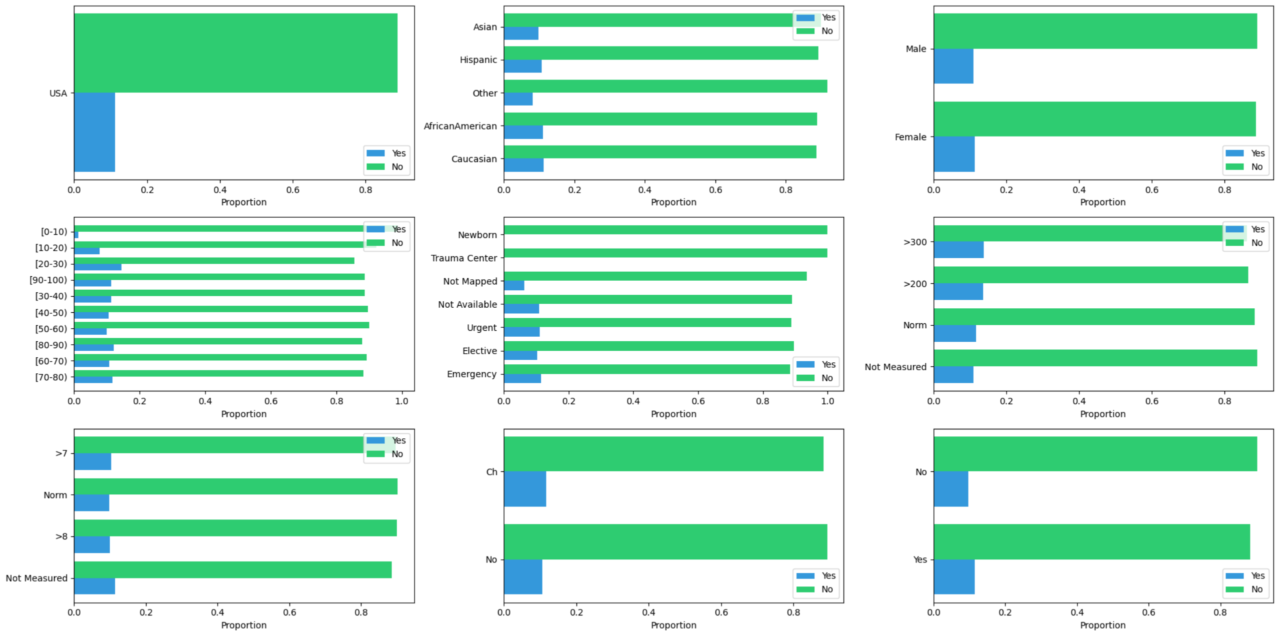

Figure A.6: Proportions plots illustrating low cardinality categorical features with readmitted_binary.

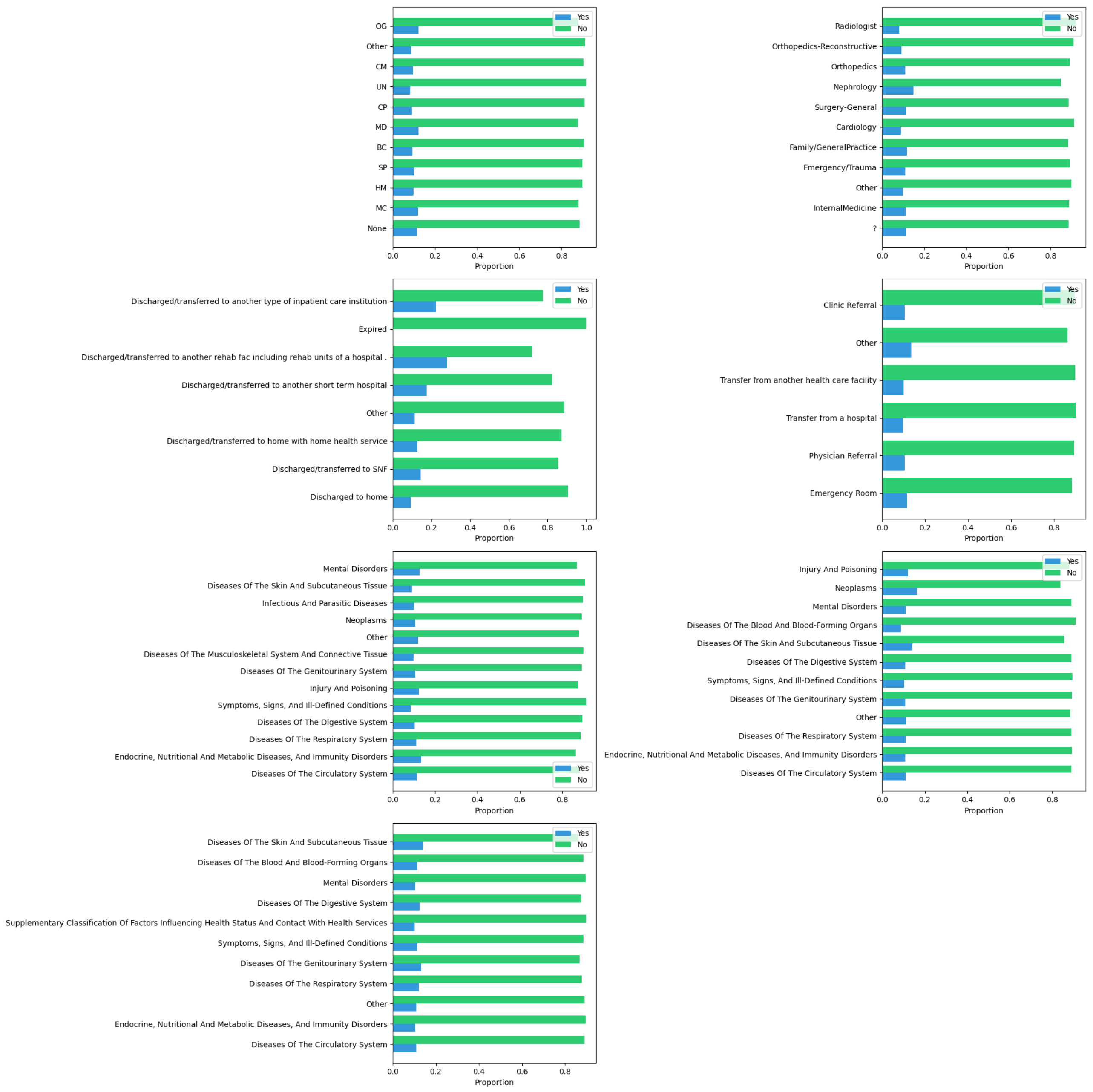

Figure A.7: Proportion plots of high cardinality categorical features and readmitted_binary.

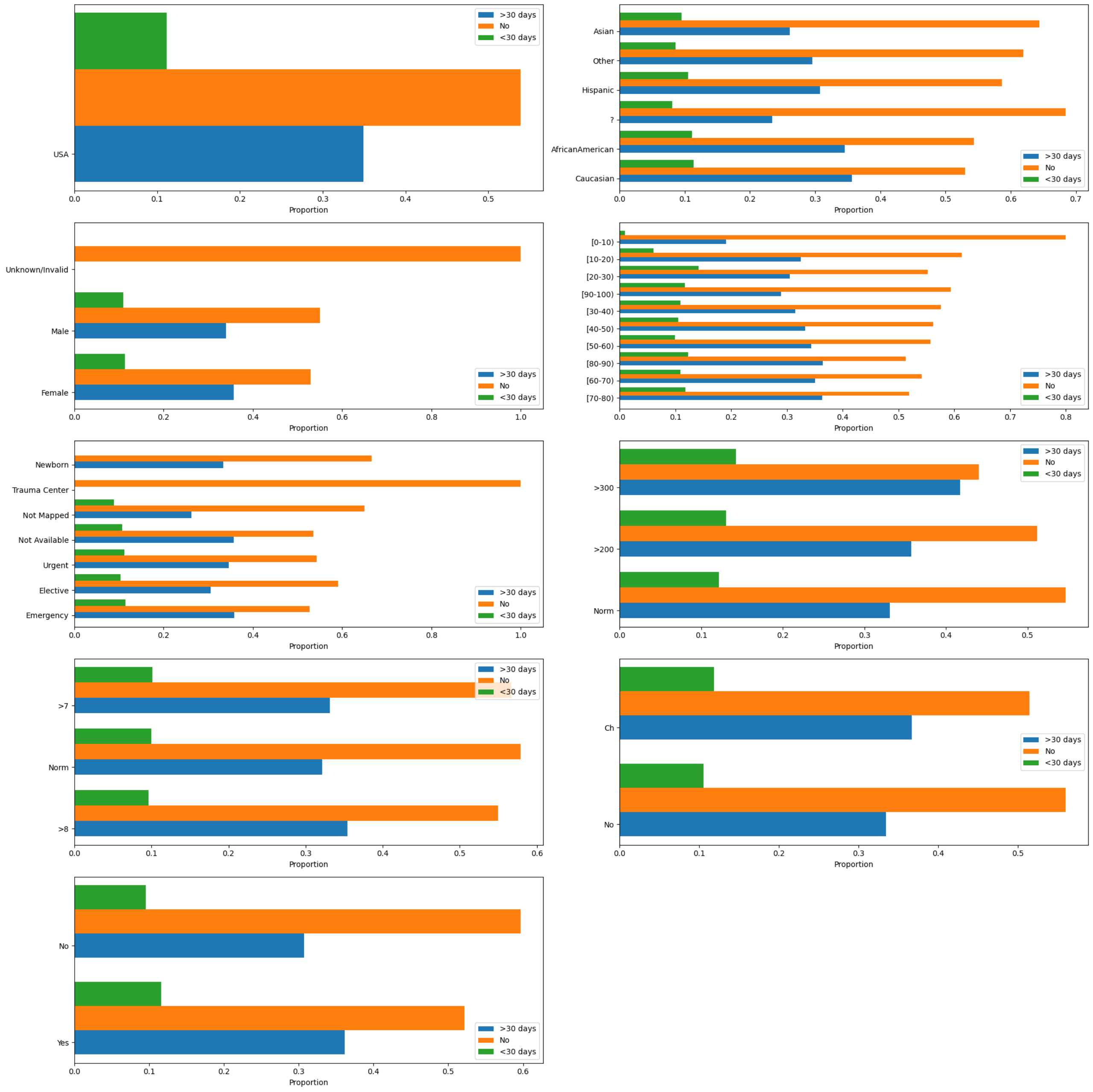

Figure A.8: Proportion plots of low cardinality categorical features and readmitted_multiclass.

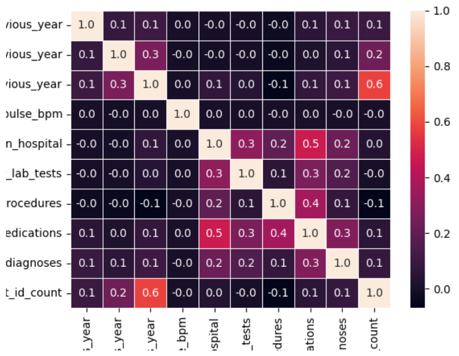
Figure A.9: Heatmap of numerical features.

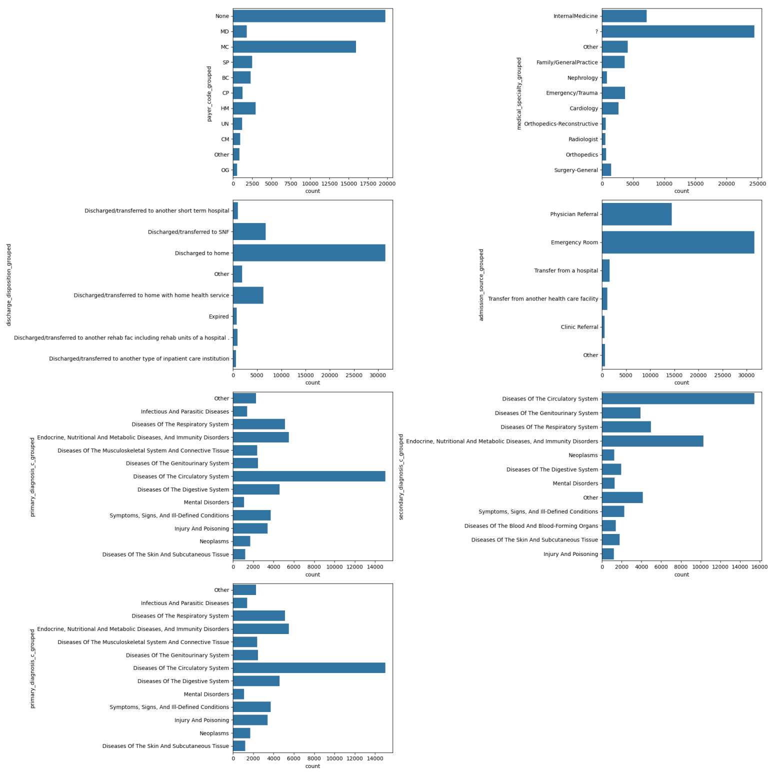
Figure A.10: Bar plots illustrating high cardinality categorical features after corrections.

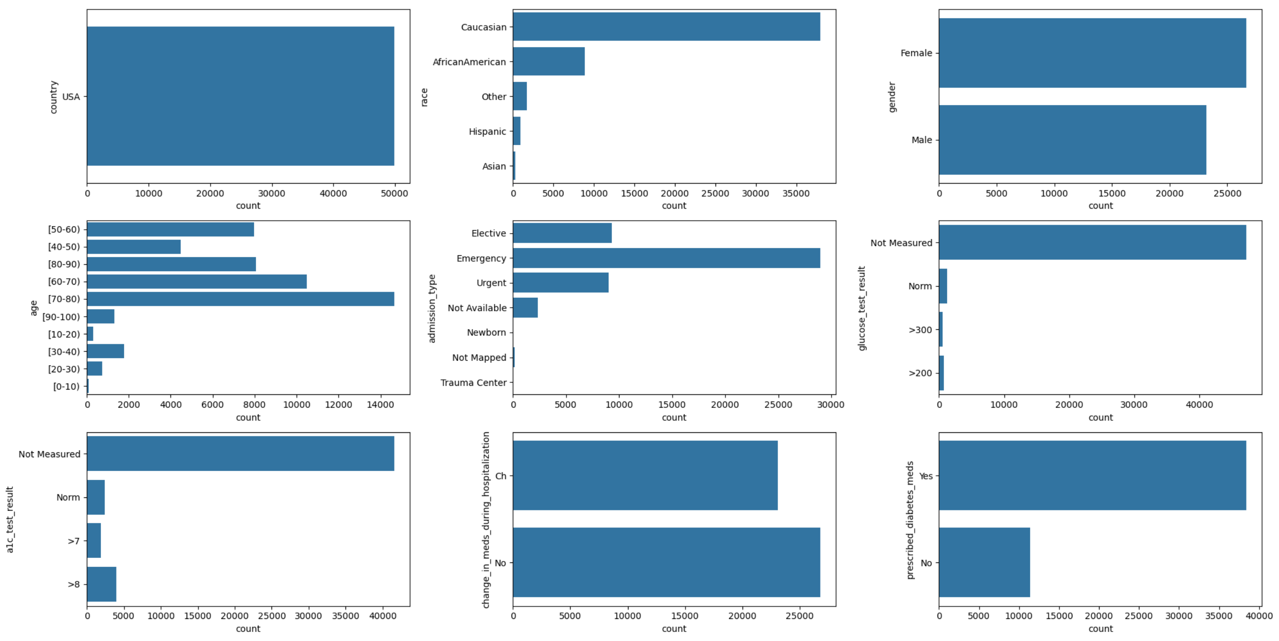

Figure A.11: Bar plots illustrating low cardinality categorical features after corrections.

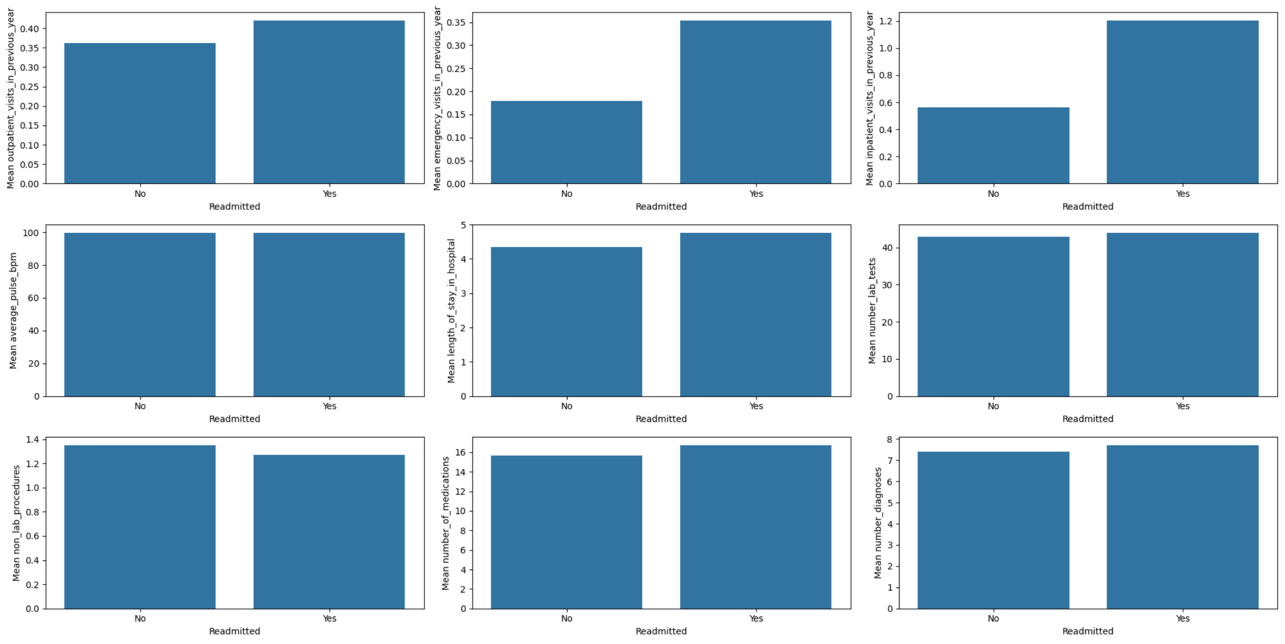

Figure A.12: Numerical features and readmitted_binary.

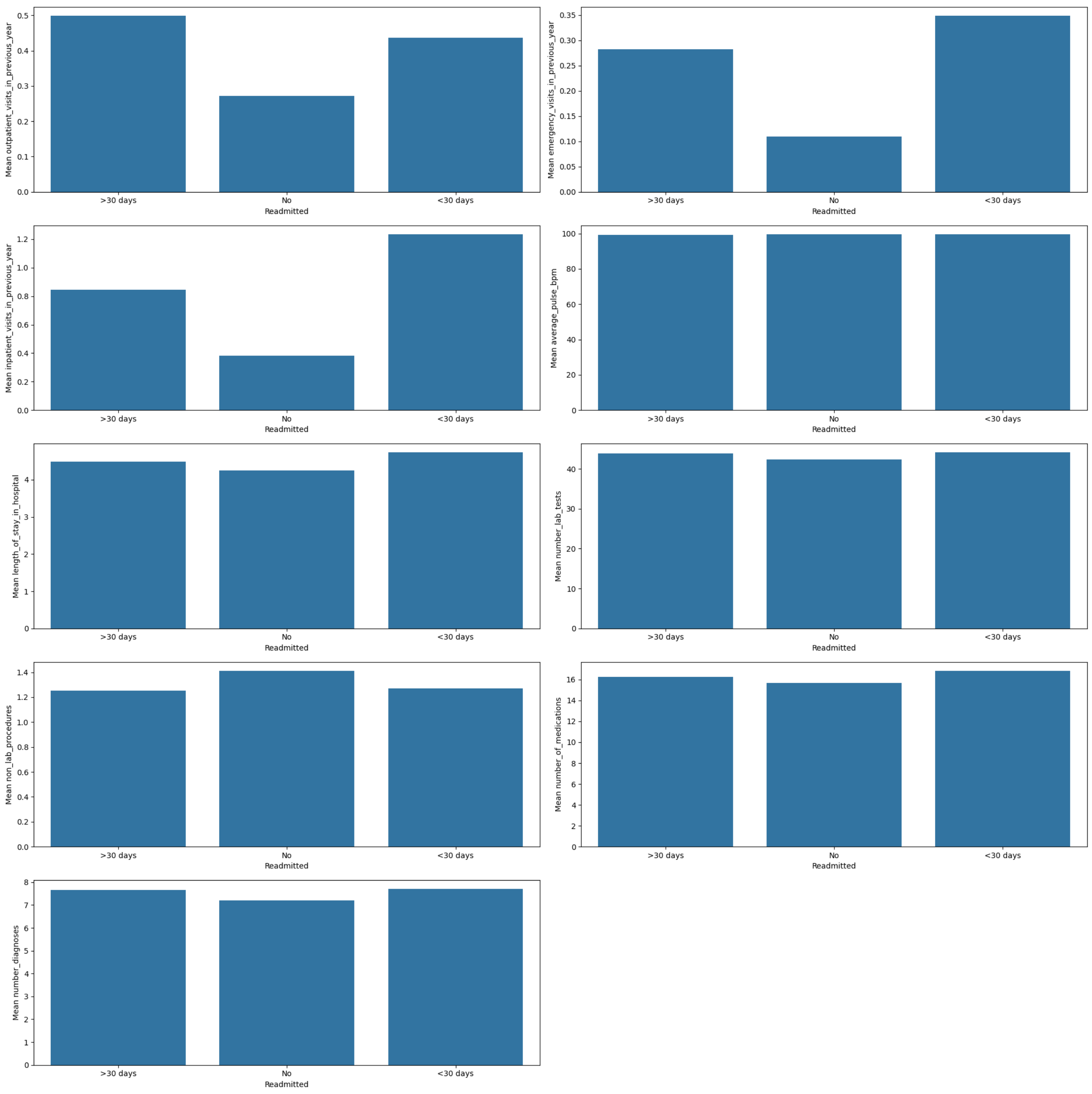

Figure A.13: Numerical features and readmitted_multiclass.