Data Report

Predicting Diabetic Patients Readmission Rates

Contributors

1. Aaron Onserio,
2. Daniel Ekale,
3. Emily Njue,
4. Robert Mbau,
5. Yussuf Hersi,
6. Jimcollins Wamae,
7. Edna Wanjiku

14th July, 2023

Table of Contents

Table of Contents ... 1
1. Business Understanding ... 3
    1.1 Problem Statement ... 3
    1.2 Business Questions ... 3
    1.3 Business Requirements ... 3
    1.4 Expected Benefits ... 3
2. Data Understanding ... 4
    2.1 Data Collection ... 4
    2.2 Data Description ... 4
3. Data Preparation ... 5
    3.1 Tasks ... 5
        3.1.1 Cleaning ... 5
        3.1.2 Deriving new features ... 5
        3.1.3 Exploratory Data Analysis ... 5
        3.1.4 Preprocessing ... 5
4. Modeling ... 7
    4.1 Baseline Model ... 7
        4.1.1 Test Design ... 7
        4.1.2 Model Building ... 7
            4.1.2.1 Parametre Setting ... 7
            4.1.2.2 Model ... 7
            4.1.2.3 Description ... 7
            4.1.2.4 Assessment ... 8
    4.2 Logistic Regression ... 8
        4.2.1 Test Design ... 8
        4.2.2 Model Building ... 8
            4.2.2.1 Parametre Setting ... 8
            4.2.2.2 Model ... 8
            4.2.2.3 Description ... 8
            4.2.2.4 Assessment ... 9
    4.3 Random Forest ... 9
        4.3.1 Test Design ... 9

4.3.2 Model Building 9
4.3.2.1 Parametre Setting 9
4.3.2.2 Model 10
4.3.2.3 Description 10
4.3.2.4 Assessment 10
4.4 Decision Tree 10
4.4.1 Test Design 10
4.4.2 Model Building 10
4.4.2.1 Parametre Setting 10
4.4.2.2 Model 11
4.4.2.3 Description 11
4.4.2.4 Assessment 11
4.5 XGBoost 11
4.5.1 Test Design 11
4.5.2 Model Building 11
4.5.2.1 Parametre Setting 11
4.5.2.2 Model 11
4.5.2.3 Description 11
4.5.2.4 Assessment 12
5. Evaluation 13
6. Deployment 14
6.1 Deployment Plan 14

1. Business Understanding

1.1 Problem Statement

This project aims to develop a predictive model to predict the likelihood of readmission of a diabetic patient. By identifying high-risk patients, healthcare providers can intervene before the readmission. Through advanced machine learning, the project seeks to enhance patients' outcomes, optimize resource allocation and improve the overall quality of care for diabetic patients.

1.2 Business Questions

I. How can we accurately predict the likelihood of readmission for diabetic patients?
II. What significant factors contribute to readmission?
III. Can we identify ways to reduce readmission rates?
IV. Which features have the strongest association with readmission rates?

1.3 Business Requirements

I. High accuracy in predicting readmission.
II. Identify key factors influencing readmission.
III. Develop a user friendly interface for healthcare providers to access.

1.4 Expected Benefits

I. Enhanced patient care: Healthcare providers can proactively identify high-risk patients thus providing targeted interventions to prevent readmissions.
II. Resource optimization: Resource allocation will be optimized by focusing on areas that have the most impact on reducing readmission rates for individual patients.
III. Cost reduction: By reducing readmission rates, healthcare organizations can minimize healthcare costs associated with repeated hospitalizations.
IV. Improved patient outcomes: By addressing the factors contributing to readmissions, healthcare providers can improve patient outcomes and overall quality of care.

The project’s success metric is Recall. This is because for the healthcare industry, a false positive which in this case would be predicting Readmission when it is not, is preferable than predicting No Readmission when it should be Readmission.

2. Data Understanding

2.1 Data Collection

The data used in this project was acquired from the University of California, Irvine machine learning datasets.

The data has over 40 features and about 100,000 instances. Some of the features in the data are not relevant to predicting readmission rates.

2.2 Data Description

The data had various issues that we had to deal with. This include:

I. Missing Values: Some columns had large percentages of missing values.
II. Placeholders: Some of the columns had placeholders which also had a high occurence.

There were 13 numerical columns and 37 categorical columns in the data.The dataset was large enough to accommodate all the necessary cleaning and wrangling steps that had to be undertaken.

3. Data Preparation

3.1 Tasks

Due to the issues our data had, we used several methods to prepare it for models. This steps are outlined as follows:

3.1.1 Cleaning

In this task, we had columns that were dropped due to the high number of missing values and placeholders in them. The most influential column that was dropped was weight which had 97% missing values.There were also some columns that had entries that could be combined into one entry. This was achieved by understanding what each entry signified.

3.1.2 Deriving new features

Some features had similar data that could be combined into one thereby reducing the features used in modeling. For others, we created new features to add another dimension that we felt would be complementary to the models.

3.1.3 Exploratory Data Analysis

From the various features we had at this point, we applied appropriate visualizations to analyze the various patterns within the data. There were some interesting discoveries which led to some features being excluded from the modeling stage. Bivariate and Univariate analysis were the most prominent.

3.1.4 Preprocessing

This was a critical step for our data due to the various features available. We applied the following steps:

I. Label Encoding: We preferred using this compared to One Hot Encoding because it does not create additional columns.

II. Train - Test Split: We used a 75:25 split for our data so as to have enough data for both training and testing.

III. SMOTE: Our data had class imbalance and we used SMOTE to increase representation for the Readmission class.

IV. Standard Scaler: We used to make our features more comparable and to avoid the influence of different scales to the models.

4. Modeling

4.1 Baseline Model

For the baseline, we used the DummyClassifier from sklearn. The reason for this was to have a basic idea about how our data is fairing in the model and to have a baseline on which to iterate.

4.1.1 Test Design

Our dataset had already been split into training and testing using the 75:25 split. The model was trained on the train split and tested on the test split.

Recall was the set measure of success throughout the project.

4.1.2 Model Building

4.1.2.1 Parametre Setting

We used two parameters for this model:

Strategy: ‘most_frequent’ - predicts the most frequent class

Random state: 42 - Help with reproducibility any time the code is run

4.1.2.2 Model

This produced one model that predicts the most frequent class in the training data.

4.1.2.3 Description

This is a simple baseline model that will always predict the most frequent class from the training data

It was developed to serve as a benchmark for comparing with more advanced models.

The model doesn’t learn any patterns in the data and provides no insights into the underlying factors related to the target variable.

This model is not suitable for prediction but serves as a reference point to assess the performance of later models.

4.1.2.4 Assessment

The model’s recall for the minority class(readmission) is 0%, indicating that it does not correctly identify any instances of the class. Recall for the majority class is 100% indicating that ii is biased towards predicting the majority class.

Given the target variable, it is crucial to prioritize the recall of the minority class to accurately identify and intervene for those at higher risk. The current model’s recall highlights the limitations of using the data as it is and using the ‘most_frequent’ strategy.

For the next models, the data was oversampled to balance the two classes.

4.2 Logistic Regression

The model utilized balanced data to predict the target variable.

4.2.1 Test Design

Our dataset had already been split into training and testing using the 75:25 split. Oversampling using SMOTE was applied to the training data to remove the effects of class imbalance.

4.2.2 Model Building

4.2.2.1 Parametre Setting

We used two parameters for this model:

Fit intercept: True - Gives more flexibility to the model to adapt to the data and also potentially capture more complex relationships.

penalty: ‘l2’ - Helps to prevent overfitting and encourage the model to generalize to unseen data.

4.2.2.2 Model

This produced one model with the specified parameters.

4.2.2.3 Description

This was a basic regression model for binary classification tasks.

It learns the relationship between the input features and the target variable by estimating the coefficients for each feature.

The "fit_intercept=True" setting includes an intercept term in the model, allowing for a shift in the decision boundary.

The L2 penalty (ridge regularization) helps control overfitting by adding a penalty term based on the squared magnitude of the coefficients.

4.2.2.4 Assessment

The model's recall improved to 57%. This was a significant jump from 0% in the baseline model. This meant that the model was able to accurately predict more than half of the positive class.

The model was able to generalize on unseen data evident from the results from the test data.

However, this is still not meeting the target threshold set for the project.

4.3 Random Forest

The model utilized balanced data to predict the target variable. It was implemented using a pipeline.

4.3.1 Test Design

Our dataset had already been split into training and testing using the 75:25 split. It used the oversampled data.

4.3.2 Model Building

4.3.2.1 Parametre Setting

We used four parameters for this model:

criterion='entropy': Uses the entropy criterion to measure the quality of a split during tree construction.

max_depth=31: It sets the maximum depth of the decision trees in the random forest to 31.

max_features='auto': Automatically determines the number of features to consider when looking for the best split at each tree node.

random_state=42: It sets the random seed for reproducibility

.

4.3.2.2 Model

The pipeline combined StandardScaler and RandomForest to produce one model.

4.3.2.3 Description

The model built using the pipeline is a random forest classifier with specified parameters.

The model demonstrated high metrics across all data splits.

4.3.2.4 Assessment

The model’s recall was the best of all models. It was however overfitting slightly on the training data.

It has a high sensitivity in correctly identifying positive cases.

Due to the overfitting, we proceeded to another model.

4.4 Decision Tree

It selects the best attribute to split the data based on a metric such as entropy or Gini impurity, which measures the level of impurity or randomness in the subsets. The goal is to find the attribute that maximizes the information gain or the reduction in impurity after the split.

4.4.1 Test Design

Our dataset had already been split into training and testing using the 75:25 split. It used the oversampled data.

4.4.2 Model Building

4.4.2.1 Parametre Setting

We used four parameters for this model:

> criterion='entropy': Uses the entropy criterion to measure the quality of a split during tree construction.
>
> max_depth=31: It sets the maximum depth of the decision trees in the random forest to 31.
>
> max_features='auto': Automatically determines the number of features to consider when looking for the best split at each tree node.
>
> random_state=42: It sets the random seed for reproducibility.

4.4.2.2 Model

The pipeline combined StandardScaler and Decision Tree to produce one model.

4.4.2.3 Description

The model built using the pipeline is a decision tree classifier with specified parameters.

The model also demonstrated high metrics across all data splits.

4.4.2.4 Assessment

The model's recall dropped to 86% which met the target threshold of the project. It still had slight overfitting which led to creating another model.

4.5 XGBoost

It uses a pipeline to implement the xgbclassifier.

4.5.1 Test Design

Our dataset had already been split into training and testing using the 75:25 split. It used the oversampled data.

4.5.2 Model Building

4.5.2.1 Parametre Setting

We used one parameters for this model:

random_state=42: It sets the random seed for reproducibility.

4.5.2.2 Model

The pipeline combined StandardScaler and XGBClassifier to produce one model.

4.5.2.3 Description

The model built using the pipeline is an xgboost with specified parameters.

The model also demonstrated high metrics across all data splits.

4.5.2.4 Assessment

It posted a high recall of 93% meaning it was able to accurately identify approximately 93% of the positive instances out of all the actual positive instances. This suggests a high sensitivity in correctly identifying positive cases.

It was able to generalize on unseen data thereby being preferred over random forest.

5. Evaluation

The results from the model were as follows:

Recall 93%: This was in two models. Xgboost and random forest.This mean that the two models could accurately predict 93% of the positive cases correctly.

Training Score 94%: This suggests that the model performs well on the data it was trained on.

Test Score 93.39%: The test score is similar to the training score, indicating that the model is not overfitting or underfitting the training data and generalizes well to unseen data.

Overall, the performance was satisfactory to the set objectives.

6. Deployment

6.1 Deployment Plan

XGBoost was the model taken to deployment.

Streamlit was used in deployment due to the easier creation of a user-friendly interface.

The features used by the model were all added to the user interface.