Beating Diabetes: Predicting Early Diabetic Patient Hospital Readmittance to Help Optimize Patient Care

A PROJECT REPORT

Submitted by

Dilip kumar (17114026, 8769583553),
Balwant singh (17114018, 9116590143),
Akhilesh kumar (17114007, 8083387074)

for the fulfillment

of

CSN-300: Lab-Based Project

Under the guidance of

R. Balasubramanian, Himanshu buckchash

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE-247667

JUNE 2020

Abstract

Table of Content

Chapter-1: Introduction

As the healthcare system moves toward value-based care, CMS has created many programs to improve the quality of care of patients. One of these programs is called the Hospital Readmission Reduction Program (HRRP), which reduces reimbursement to hospitals with above average readmissions. For those hospitals which are currently penalized under this program, one solution is to create interventions to provide additional assistance to patients with increased risk of readmission. But how do we identify these patients? We can use predictive modeling from data science to help prioritize patients.

One patient population that is at increased risk of hospitalization and readmission is that of diabetes. Diabetes is a medical condition that affects approximately 1 in 10 patients in the United States. According to Ostling et al, patients with diabetes have almost double the chance of being hospitalized than the general population (Ostling et al 2017). Therefore, in this article, I will focus on predicting hospital readmission for patients with diabetes.

Motivation

The US spends $332 billion on diabetic and prediabetic care every year

Diabetes affects 350 million people with 3 million dying each year due to complications

Through analyzing diabetic patient data, we hope to help reduce the mortality rate of diabetic people through improving patient care and decrease its cost

We will analyze data from 130 hospitals in the United States from 1999 to 2008 to create 2 models that will:

1. Predict whether a diabetic patient will be readmitted to the hospital in less than 30 days (i.e., a binary model)

2. Predict the probability of readmittance within 30 days for a diabetic patient

Doctors can use these models during patient visits to guide patient care decisions

The binary model can be used to infer general patterns in the data and for holistic research on early hospital readmittance

Chapter-2: Literature Survey

According to the American Society of Diabetes, the cost of care for diabetic and prediabetic patients in the United States is 332 billion USD. This global epidemic affects over 350 million people, with 3 million people dying each year due to diabetes related complications, predominantly cardiovascular or nephropathic ones. By analyzing an extensive diabetic patient dataset accumulated from 130 hospitals in the United States from 1999 to 2008, we wish to reduce the mortality rate of diabetic people by improving patient care, while also reducing the astronomical yearly cost of diabetic patient care. In order to accomplish this, we will create a model to predict whether a diabetic patient will be readmitted to the hospital in less than 30 days from the last visit, as well as a model that expresses the probability of a diabetic patient being readmitted in less than 30 days from the last visit. Thus, the input to our models will be patient data, and we then use a variety of models (e.g. logistic regression, SVMs, random forests) to output a prediction as to whether the patient will be readmitted early. Each model is valuable in different ways. The probabilistic model can be used by doctors during a patient visit to help them with decisions such as whether to administer an HbA1 test, a costly procedure, or whether to change the dosage or administer new medications. The doctor could theoretically run the model with different hypothetical features, different medications, different tests and inform their decision using the combination of features that reduces the patient's chance of readmission into the hospital. The binary model can be used to infer general patterns in the data and to determine which patient characteristics may lead to early hospital readmission. This may expose weaknesses in the current approach doctors are taking in diabetic patient care, and hopefully pave the path to better health outcomes for diabetic people.

Related Work

Prediction of early readmittance is a topic of major interest, with various studies on this topic focusing on major diseases and procedures using electronic, such as for heart failure (Philbin & DiSalvo) and intestinal surgery (Kiran et al.).

However, these studies primarily focus upon manual analysis like Student's t-tests or chi-squared tables (Philbin & DiSalvo; Kiran et al.). However, there have been a few studies using machine learning on hospital readmission, like heart failure (Shameer et al.). In fact, that model used just a Naive Bayes algorithm to obtain an accuracy of 83%.

There has been some research conducted regarding early readmittance in the case of diabetes, where multivariable logistic regression models were used to assess the impact of HbA1c values on early readmittance. While this model did delve thoroughly into the relationship between HbA1c values and early readmittance from a statistical standpoint, particularly with the use of p-values and significance tests, there was not much focus on utilizing the logistic regression model or other machine learning algorithms for predictions. Thus, we think our application of machine learning models to this problem will yield exciting and interesting insights regarding early readmission for diabetes patients. The dataset we are leveraging for our algorithms has over 55 features, thus techniques on feature selection to diminish noise and pull out signal appeal greatly to our causes.

Chapter-3: Methodology/Experimental Setup

Setup

1. Linux

2. Jupyter notebook

Great tool to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

3. Python=3.7.3

A high-level language easy to write and provides great Machine Learning tools.

4. Scikit-Learn

Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

5. Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

6. NumPy

7. Matplotlib
8. Seaborn

Matplotlib and Seaborn are comprehensive libraries for creating static, animated, and interactive visualizations in Python.

This project will demonstrate how to build a model predicting readmission in Python using the following steps

- data exploration
- feature engineering
- building training/validation/test samples
- model selection
- model evaluation (Results)

Data Exploration

The data that is used in this project originally comes from the UCI machine learning repository (link). The data consists of over 100000 hospital admissions from patients with diabetes from 130 US hospitals between 1999–2008.

In this project, we will utilize Python to build the predictive model. Let's begin by loading the data and exploring some of the columns.

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import time

In [2]: # load the csv file
        df = pd.read_csv('diabetic_data.csv')

In [3]: print('Number of samples:',len(df))

        Number of samples: 101766
```

From briefly, looking through the data columns, we can see there are some identification columns, some numerical columns, and some categorical (free-text) columns.

There is some missing data that are represented with a question mark (?). We will deal with this in the feature engineering section.

The most important column here is readmitted, which tells us if a patient was hospitalized within 30 days, greater than 30 days or not readmitted.

```
In [5]: # count the number of rows for each type
        df.groupby('readmitted').size()

Out[5]: readmitted
        <30    11357
        >30    35545
        NO     54864
        dtype: int64
```

Another column that is important is discharge_disposition_id, which tells us where the patient went after the hospitalization. If we look at the IDs_mapping.csv provided by UCI we can see that 11,13,14,19,20 and 21 are related to death or hospice. We should remove these samples from the predictive model since they cannot be readmitted.

```
In [6]: df = df.loc[~df.discharge_disposition_id.isin([11,13,14,19,20,21])]
```

Now let's define an output variable for our binary classification. Here we will try to predict if a patient is likely to be re-admitted within 30 days of discharge.

```
In [7]: df['OUTPUT_LABEL'] = (df.readmitted == '<30').astype('int')
```

Let's define a function to calculate the prevalence of population that is readmitted with 30 days.

```
In [8]: def calc_prevalence(y_actual):
            return (sum(y_actual)/len(y_actual))

In [9]: print('Prevalence:%.3f'%calc_prevalence(df['OUTPUT_LABEL'].values))

        Prevalence:0.114
```

Around 11% of the population is hospitalized. This represented an imbalanced classification problem so we will address that below.

From further analysis of the columns, we can see there are a mix of categorical (non-numeric) and numerical data. A few things to point out,

encounter_id and patient_nbr: these are just identifiers and not useful variables.

age and weight: are categorical in this data set

admission_type_id, discharge_disposition_id, admission_source_id: are numerical here, but are IDs (see IDs_mapping). They should be considered categorical.

examide and citoglipton only have 1 value, so we will not use these variables

diag1, diag2, diag3: are categorical and have a lot of values. We will not use these as part of this project, but you could group these ICD codes to reduce the dimension. We will use number_diagnoses to capture some of this information.

feature engineering

In this section, we will create features for our predictive model. For each section, we will add new variables to the dataframe and then keep track of which columns of the dataframe we want to use as part of the predictive model features. We will break down this section into numerical features, categorical features and extra features.

(?) will be replaced by nan values.

Numerical features will not be modified.

The next type of features we want to create are categorical variables. Categorical variables are non-numeric data such as race and gender. To turn these non-numerical data into variables, the simplest thing is to use a technique called one-hot encoding.

Of our categorical features, race, payer_code, and medical_specialty have missing data. Since these are categorical data, the best thing to do is to just add another categorical type for unknown using the “fillna” function.

The last two columns we want to make features are age and weight. Typically, you would think of these as numerical data, but they are categorical.

Through this process we created 143 features for the machine learning model. The break-down of the features is

- 8 numerical features
- 133 categorical features
- 2 extra features

Building Training/Validation/Test Samples

So far we have explored our data and created features from the categorical data. It is now time for us to split our data. The idea behind splitting the data is so that you can measure how well your model would do on unseen data. We split into three parts:

1. Training samples: these samples are used to train the model
2. Validation samples: these samples are held out from the training data and are used to make decisions on how to improve the model
3. Test samples: these samples are held out from all decisions and are used to measure the generalized performance of the model

In this project, we will split into 70% train, 15% validation, and 15% test.

The first thing I like to do is to shuffle the samples using sample in case there was some order (e.g. all positive samples on top). Here n is the number of samples. random_state is just specified so the project is

reproducable. You wouldn't need random_state necessarily in your own projects.

We can use sample again to extract 30% (using frac) of the data to be used for validation / test splits. It is important that validation and test come from similar distributions and this technique is one way to do it.

Most of the things are explained in Jupyter Notebook so I am not going to repeat. (Source)

Model Selection

Wow! so much work to get ready for a model. This is always true in data science. You spend 80–90% cleaning and preparing data.

In this section, we train a few machine learning models and use a few techniques for optimizing them. We will then select the best model based on performance on the validation set.

We will first compare the performance of the following 7 machine learning models using default hyperparameters:

- K-nearest neighbors
- Logistic regression
- Stochastic gradient descent
- Naive Bayes
- Decision tree
- Random forest
- Gradient boosting classifier

There are many ways to select model

1. Analysis of baseline models

Make a dataframe with the results of all the baseline models and plot the outcomes using a package called seaborn. In this project, we will utilize the Area under the ROC curve (AUC) to evaluate the best model. This is a good data science performance metric for picking the best model since it captures the trade off between the true positive and false positive and does not require selecting a threshold.

2. Learning Curve

We can diagnose how our models are doing by plotting a learning curve.

3. Feature Importance

One path for improving your models to understand what features are important to your models. This can usually only be investigated for simpler models such as Logistic Regression or Random Forests.

4. Hyperparameter Tuning

The next thing that we should investigate is hyperparameter tuning. Hyperparameter tuning are essentially the design decisions that you made when you set up the machine learning model. For example, what is the maximum depth for your random forest? Each of these hyperparameters can be optimized to improve the model.

5. Best Classifier

Here we will choose the gradient boosting classifier since it has the best AUC on the validation set. You won't want to train your best classifier every time you want to run new predictions. Therefore, we need to save the classifier. We will use the package pickle.

Chapter-4: Results

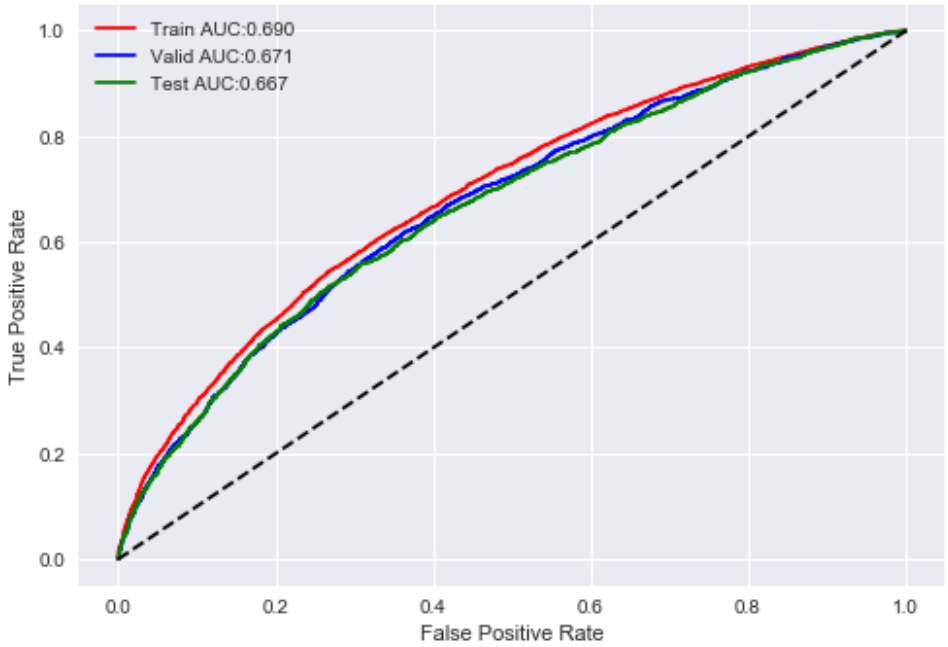
Model Evaluation

Now that we have selected our best model (optimized gradient boosting classifier). Let's evaluate the performance of the test set.

The final evaluation is shown in the table

	Training	Validation	Test
Prevalence	0.5	0.113	0.117
AUC	0.69	0.671	0.667
Accuracy	0.637	0.66	0.65
Recall	0.583	0.583	0.578
Precision	0.654	0.184	0.184
Specificity	0.692	0.67	0.66

We can also create the ROC curve for the 3 datasets as shown below



Chapter-5: Conclusion and Future Work

- Through this project, we created a machine learning model that is able to predict the patients with diabetes with highest risk of being readmitted within 30 days. The best model was a gradient boosting classifier with optimized hyperparameters. The model was able to catch 58% of the readmissions and is about 1.5 times better than just randomly picking patients.
- In the future, we would be interested in running predictions on other aspects of the dataset (e.g. predicting A1C values).

References

Ostling, Wyckoff, Ciarkowski, Pai, Choe, Bahl, Gianchandani (2017). “The relationship between diabetes mellitus and 30-day readmission rates” in *Clinical Diabetes and Endocrinology*.

Shameer, K., Johnson, K. W., Yahi, A., Miotto, R., Li, L. I., Ricks, D., ... & Moskovitz, A. (2017). Predictive modeling of hospital readmission rates using electronic medical record-wide machine learning: a case-study using Mount Sinai Heart Failure Cohort. In PACIFIC SYMPOSIUM ON BIOCOMPUTING 2017 (pp. 276-287).

Hospital Readmissions Reduction Program. (n.d.). Retrieved December 13, 2017, from https://www.medicare.gov/hospitalcompare/readmission-reduction-program.html

- References for Dataset

Beata Strack, Jonathan P. DeShazo, Chris Gennings, Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios, and John N. Clore, “Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records,” BioMed Research International, vol. 2014, Article ID 781670, 11 pages, 2014.

“ Diabetes 130-US Hospitals for Years 1999-2008 Data Set.” UCI Machine Learning Repository: Diabetes Data Set, UCI Center for Machine Learning and Intelligent Systems, 3 May 2014, https://archive.ics.uci.edu/ml/datasets/diabetes+130-us+hospitals+for+years+1999-2008