Case Study 2: Predicting Hospital Readmissions for Diabetic Patients

Matt Farrow

May 22, 2022

1 Introduction

The data for this analysis comes from UCI’s Machine Learning Repository and is intended to predict hospital readmittance among patients with diabetes.

The objective of this case study is take a data set that contains missing information and make decisions about what predictors to keep and which to discard. In addition, data that is kept will likely need to be imputed to take into account missing data.

2 Methods

## 2.1 Data Examination

The initial data set is comprised of two separate files: diabetic\_data.csv and IDs\_mapping.csv; the latter file contains feature mappings for several variables that will be used in the data manipulation. The response variable readmitted notes whether the patient was not readmitted to the hospital, was readmitted after more than 30 days, or was readmitted within 30 days. The new data set contains 101,766 observations and 50 variables including our response.

In understanding the data, we first reviewed the number of unique obervations in each variable along with the amount of missing data. The decision was made to remove the weight variable since almost 97% of the entries were missing. Although payer\_code and medical\_specialty may have been able to be imputed, with so much data missing the decision was made to remove those features.

Table

Description automatically generated

Table 1: Missing Data

Additional examination revealed that examide and citoglipton each only contained a single value, so those were also removed from the analysis. ID columns encounter\_id and patient\_nbr could be useful in more detailed examinations of diabetic patient readmission, but in this case study those fields could also be removed.

Based on previous research[[1]](#footnote-2), the icd9 codes were able to be mapped to the appropriate group name and then to the three diagnosis variables. The mapping data set broke down the different types of admission\_type\_id, admission\_source\_id, and discharge\_disposition. In the case of the two ID columns, there were several values that all mapped to various versions of NULL data; therefore those were all collected into a single value. In the case of dischanges, patients who passed away or who went from the hospital into hospice were removed from the analysis since they would not be part of the readmission population.

## 2.2 Model Preparation & Execution

The response variable readmitted, as noted earlier, contains three values – ‘NO’, ‘>30’ and ‘<30’. Rather than try and run a multinomial logistic regression, we combined the ‘NO’ and ‘>30’ values and set the response to be a binary value of 0 or 1 with 1 being a patient who was readmitted to the hospital within 30 days. The response was separated from the rest of the data and then both the X and y data sets were split into test and train data sets using a 75%/25% split.

Sklearn’s pipeline feature was used set up imputation and preprocessing steps prior to modeling. For numeric variables, missing data was imputed using the median of the variable and the data was scaled used the RobustScaler. For categorical variables, the missing data was imputed using the most frequently found string in the data. These variables were also one-hot encoded. Finally a preprocessor was defined to iterate over the numeric and categorical columns to apply the transformations.

This preprocessing was sent through Pipeline’s imbpipeline function along with a SMOTE transformation along with the logistic regression. SMOTE, which stands for Synthetic Minority Oversampling Technique, is a machine learning technique that can be used to address imbalanced data sets.

Finally, a stratified K-fold cross validation was set up along with a parameter grid search that used ROC curves as the scoring metric.

|  |  |  |
| --- | --- | --- |
| Grid Search | Hyperparameter | Search Values |
| 1 | C | 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000 |
|  | Penalty | L1 |
|  | Solver | saga |
| 2 | C | 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000 |
|  | Penalty | L2 |
|  | Solver | lbfgs |

Table 2: Hyperparameter Grid Search

3 Results

## 3.1 Model Results

After running the model through the grid search, the resulting cv\_score and test\_score were examined.

|  |  |
| --- | --- |
| cv\_score | 0.645 |
| test\_score | 0.649 |

Table 3: Model Results

## 3.2 Feature Importance

After completing the logistic regression, the most positive and negative features were examined.

|  |  |
| --- | --- |
| Feature | Weight |
| number\_inpatient | + 0.235 |
| number\_diagnoses | + 0.127 |
| A1Cresult\_None | + 0.093 |
| Diag1\_Circulatory | + 0.080 |
| race\_Caucasion | + 0.077 |
| discharge\_disposition\_id | + 0.060 |
| num\_procedures | - 0.056 |
| Age\_[50-60) | - 0.059 |
| insulin\_No | - 0.079 |
| <BIAS> | - 1.195 |

Table 5: Most Importance Features

Chart, line chart

Description automatically generated

4 Conclusion

In examining the top five coefficients of each model, additional work may have been needed to avoid potential multicollinearity as both models contained high performing coefficients that appear as though they may be correlated. It is also interesting to note that the coefficients of the L1 (LASSO) model have a much narrower distribution; within the L2 (Ridge) model, the top two performing coefficients were significantly separated from the next values.

# Appendix

## Sources

* [Lasso Regression with Python](https://www.kirenz.com/post/2019-08-12-python-lasso-regression-auto/#model-evaluation)
* [How to Use Sklearn Pipelines For Ridiculously Neat Code](https://towardsdatascience.com/how-to-use-sklearn-pipelines-for-ridiculously-neat-code-a61ab66ca90d)
* [Pre-Process Data with Pipeline to Prevent Data Leakage during Cross-Validation](https://towardsdatascience.com/pre-process-data-with-pipeline-to-prevent-data-leakage-during-cross-validation-e3442cca7fdc)
* [sklearn.linear\_model.LinearRegression](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html)
* [How to Develop LASSO Regression Models in Python](https://machinelearningmastery.com/lasso-regression-with-python/)
* [How to create a linear regression model using Scikit-Learn](https://practicaldatascience.co.uk/machine-learning/how-to-create-a-linear-regression-model-using-scikit-learn)
* [Linear Regression in Python](https://realpython.com/linear-regression-in-python/)

## Code

Code begins on the following page.

1. Beata Strack, Jonathan P. DeShazo, Chris Gennings, Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios, 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. https://doi.org/10.1155/2014/781670 [↑](#footnote-ref-2)