**Data Mining Project**

**Douglas Mensah**

**Jake Okinow**

**Statistics of model**

Model: From imblearn – balancedrandomforestclassifier

Hyperparameters:

n\_estimators: 300

max\_features: auto

max\_depth: 20

criterion: entropy

random\_state: 42

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Precision | Recall | F1-score | Support |
| Not readmitted | 78% | 58% | 66% | 12376 |
| Readmitted | 40% | 64% | 49% | 5504 |
|  |  |  |  |  |
| accuracy |  |  | 60% | 17880 |

|  |  |  |
| --- | --- | --- |
| Confusion matrix | Not readmitted | Readmitted |
| Not readmitted | 7150 | 5226 |
| readmitted | 2003 | 3501 |

**Discussion of data mining algorithms and techniques employed**

Data Exploration

The dataset had over 100000 records/entries from about 130 hospitals between 1998 - 2008. Upon observation of the features and their values, it was identified that there were 3 types of datatypes thus, categorical and numerical. This further gave us an idea of what kind of processing will be involved in our data wrangling and cleaning during the preprocessing phase of the project. The readmitted column was indicated to be the target which was also categorical data.

Data Cleaning

Removing multiple encounters for the same patient

It is observed that there were multiple entries or records based on the patient\_nbr. We got rid of all multiple entries and kept only the first entries which further reduced the length of the dataset to about 71,000(tackling data redundancy)

Dealing with missing data

It’s observed that missing data was represented by ?. To better handle missing data we replaced ? with np.nan. This will enable us to get proper statistics and easy manipulation of each feature.

Also for missing categorical features, we used the mode value(most frequent) of each feature to impute the missing values.

Medical specialty: this feature has about 73 variables. It’s preferred to always have about 10-15 variables per feature when performing feature engineering. We processed this by replacing all missing values with the mode. The top 14 variables identified and the remaining variables were labeled as other. This concluded with having 15 variables per this feature.

Feature Engineering

Dropping features

At this early stage we dropped the following features from the dataset:

'encounter\_id','patient\_nbr','examide','citoglipton','weight','payer\_code’

encounter\_id and patient\_nbr: is of no benefit because they only serve as id.

payer\_code: has no importance with dataset

weight: had about 70% missing data so it was dropped

examide & citoglipton: these features had one value running through all the records. hence there is no need to keep an additional column that fails to show any variation in the dataset because it’s not going to help our model learn different patterns.

Encoding Features

One Hot Encoding

To convert our categorical features to numbers. One hot encoding was performed on the race feature and to further prevent the columns from being correlated, the first categorical value was dropped.

Diag\_1, diag\_2 & diag\_3 were categorical data that had their own mapping per value. The values were mapped to the original disease per the article and one hot encoding performed on it. We dropped diag\_2 and diag\_3 and focused on the primary diag thus diag\_1

Label Encoding

The remaining dataset that had categorical features were converted to numerical values by performing label encoding.

In the end we have 63 features

We used a cross validated grid search to locate the best hyperparameters for my model. Cross validation is the process of splitting data into k groups and then training the model on k-1 groups and testing on the left out group. This allows the model to train and test on all the data. This helps reduce bias. A grid search is used to exhaustively search through hyperparameters for a given model. For example if you have three hyperparameters that can have three values each then a grid search can be used to run the model on each combination of hyperparameters and select the combination that gives the best score. We used undersampling techniques to try to deal with imbalanced data. Undersampling is the technique of using the number of samples from the smallest class as the number of samples from each class. For example, there are 50 total samples and we have 40 from class 1 and 10 from class 2. In undersampling we would only use 10 from class 1 and 10 from class 2 and throw out the rest of the samples from class 1

**Report analyzing your models performance and discoveries on data set**

The objective of this project was “to predict, as accurately as possible, whether a person will be readmitted within 30 days after being discharged from the hospital.” Initially we ran the data though nearest neighbor, gradient boosting, and random forest models. The accuracies were around 64%, 68%, and 69% respectively. We thought that was alright results but when I looked at the confusion matrix we realized that the accuracy was not the right metric to analyze whether our model is completing the objective. Accuracy is the measure of all properly identified classes over all samples. What these models were doing was basically identifying all the labels as not readmitted. The recall, which is the total positive/ false negative + total positive, is a better metric for how well we predict whether a person will be readmitted. It is essentially what percentage did we guess right for that class. The recall for these models were around 10%. So these models did not actually help much at all. We tried to undersample to combat the imbalanced data and use the built in class weight feature for randomforests. This did increase the recall but the best result for my recall was to use the balancedrandomforest classifier from the imbalanced learn package. This model automatically undersamples for each tree. This model gave a 64% recall for the readmitted class. The issue with this is that the accuracy of this model was 60% and the recall for the non readmitted class was 58%. In our test set with 12,376 not readmitted samples, 5,226 were classified as would be readmitted within 30 days. There are only 5,504 samples in the readmitted within 30 days class and only 3501 were correctly classified as readmitted. So while we are correctly classifying more people who will be readmitted we are also incorrectly classifying more people who would not be readmitted.