May 12, 2021

Using AI to Help Predict Hospital Readmission in Patients with Diabetes

Yang Pan
AI CS 4300
University of Missouri-St. Louis

Abstract

This project is done to see if we can predict which patients are likely to be readmitted to the hospital for diabetes within 30 days of the initial visit. This project also aims to identify key features contributing to hospital readmission within 30 days.

1 Data Description

In this section I will discuss the dataset that I am working with, the source of this data is at here; from the UCI.edu website.

1.1 Dataset Label

The label we are interested in is "Readmitted" column. It shows weather or not the patient was readmitted to the hospital for the same issue.

$$No \tag{1}$$

- was not readmitted

$$> 30 \tag{2}$$

- readmitted after 30 days

$$< 30 \tag{3}$$

-readmitted within the month

1.2 Dataset Features and Brief Explanations

- gender
- age
- `admission_type_id` - id if the reason why patient was admitted, such as trauma, elective, newborn..
- `discharge_disposition_id`- id of where patient was discharged to
- `admission_source_id`- id of medical source of why patient was admitted
- `time_in_hospital` -time in days patient was admitted
- `num_lab_procedures` -number of lab procedures ordered
- `num_procedures` -number of medical procedures in hospital
- `num_medications` -number of medication given to patient in hospital
- `num_outpatient` -number of times patient seen within the year as outpatient prior to now
- `num_emergency`- number of emergencies within the year
- `num_inpatient`-number of times patient was seen as inpatient in the year prior to now
- `number_diagnosis`-total number of diagnosis for patient
- `max_glu_serum` -max blood sugar measured
- A1Cresult -lab measurement of 3month avg blood sugar
- various drugs names
- change - indicates if change in diabetes medication
- diabetesMed -indicates of any diabetes meds were prescribed

2 Output Original and Manipulation

2.1 Output Manipulation

For better accuracy during classification, we merged those that were readmitted greater than 30 days and not readmitted into the same category.

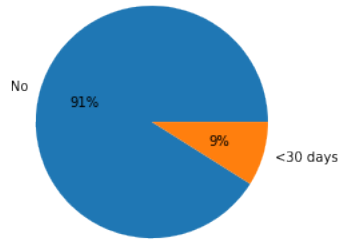

Figure 1: Percentages of patients NOT readmitted before 30 days, and readmitted before 30 days

2.2 Input Manipulation

Gender replaced with 0 or 1, 0 is female, 1 is male

Age changed into age category's median, for example: age 49 is now 45 because 49 is in 40-49 age group, age 12 is 15, because 12 is in 10-19 age group...etc.

`admission_type_id` regrouped:

Category Number	Explanation
1	Any admission from emergency
5	No information known
3	Patient chose to come electively

`discharge_disposition_id` regrouped:

Category Number	Explanation
1	Discharged to a facility considered Home
2	Discharged to another medical facility
10	Discharged for outpatient services
11	Hospice
18	UNKNOWN

`admission_source_id` regrouped:

Category Number	Explanation
1	Referral from medical staff
4	Transfer patient
9	UNKNOWN
11	Anything to do with pregnancy or babies

For the columns with medications, numeric values were assigned to weather or not patient is taking a certain medication, and whether or not the dose was increased or decreased.

`Medications` regrouped:

Dose Change	Assigned numeric value
Up	10
Down	-10
Discontinued	-20
Steady	0

A1c was converted to medical category based on whether or not it is within normal range.

`A1C` regrouped:

A1C	Assigned numeric value
>7	7
>8	8
Norm	5
Other	0

Max Glucose was also relabeled and categorized based on medical guidelines

`Max Glucose` regrouped:

Max Serum Glucose	Assigned numeric value
>200	200
>300	300
Norm	100
Other	0

3 Normalization Technique

Normalization Technique used was Re-scaling

$$Y = Y/Y.max() \tag{4}$$

After Normalization of Input features:

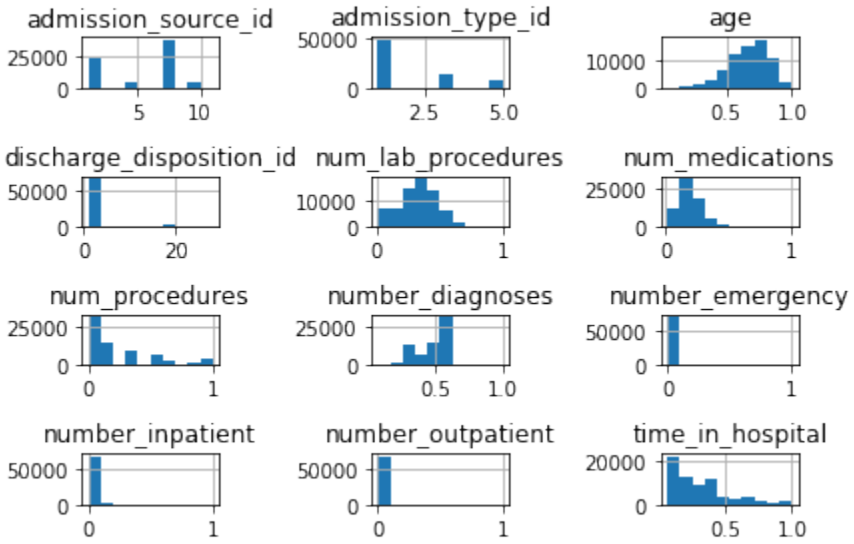

Figure 2: Initial input features histogram after normalization, note: The graphs with id are not normalized as the numeric values is only used for categorical purposes

Phase II

4 Model selection and Evaluation

4.1 Splitting data into training and test

Data was split with from sklearn's model selection importing the train test split, 80 percent of rows in data set was used training, while 20 percent was used for testing.

4.2 Dealing with imbalanced data

The number of patients that were readmitted within 30 days was greatly smaller than the number of patients that were not, close to a 9:1 ratio, therefore having imbalance in the binary classification. Techniques tried were SMOTE, ADASYN, SMOTEENN, after testing, synthetic data was made to counteract the imbalance in data. SMOTEENN performed the best.

Without oversampling, accuracy was higher, however, the model would miss every minority classification

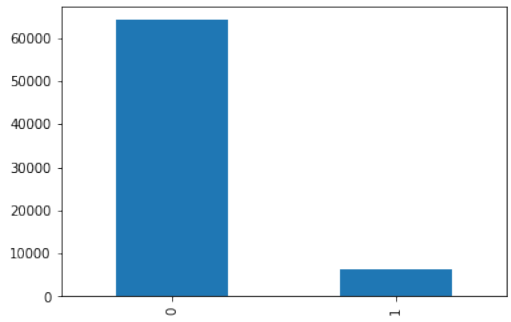

Figure 3: distribution of label before oversampling

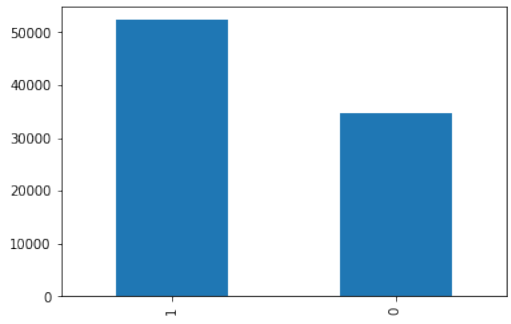

Figure 4: distribution of label after SMOTEENN synthetic samples

5 Baseline Model Performance

Num of hidden Layers	Accuracy	Loss
0 Layer	61.03 percent	162.79 percent
1 Layer	60.62 percent	67.06 percent
2 Layers	60.58 percent	67.08 percent
3 Layers	60.93 percent	66.65 percent

I chose to use 3 hidden layers for the non baseline tests, although it is slightly less accurate than 0 hidden layers in accuracy, it has less variability

5.1 Code snippet of baseline models used

```
0]: ## Baseline Model #1 , with 2 hidden layer
    model1 = Sequential()
    model1.add(Dense(16, input_dim = len(X_train[0, :]), activation='relu'))
    model1.add(Dense(8, activation='relu'))
    model1.add(Dense(4, activation='relu'))
    model1.add(Dense(1, activation='sigmoid'))

1]: ## Baseline Model #2 , 1 hidden layer
    model2 = Sequential()
    model2.add(Dense(8, input_dim = len(X_train[0, :]), activation='relu'))
    model2.add(Dense(6, activation='relu'))
    model2.add(Dense(1, activation='sigmoid'))

2]: ## Baseline Model #3 , no hidden layer
    model3 = Sequential()
    model3.add(Dense(32, input_dim = len(X_train[0, :]), activation='relu'))
    model3.add(Dense(1, activation='sigmoid'))

3]: ## Baseline Model #4
    model4 = Sequential()
    model4.add(Dense(32, input_dim = len(X_train[0, :]), activation='relu'))
    model4.add(Dense(16, activation='relu'))
    model4.add(Dense(8, activation='relu'))
    model4.add(Dense(4, activation='relu'))
    model4.add(Dense(1, activation='sigmoid'))
```

Figure 5: Baseline models tested

After testing rmsprop, sgd, nadam, adam, gd, adam performed the best

5.2 Code snippet of models tested

```
In [34]: ## Model using sigmoid activation (only last neuron)
modelA = Sequential()
modelA.add(Dense(32, input_dim = len(X_train[0, :]), activation='relu'))
modelA.add(Dense(16, activation='relu'))
modelA.add(Dense(8, activation='relu'))

modelA.add(Dense(4, activation='relu'))
modelA.add(Dense(1, activation='sigmoid'))
```

```
In [35]: ## Model using sigmoid activation (all neurons)
modelB = Sequential()
modelB.add(Dense(32, input_dim = len(X_train[0, :]), activation='sigmoid'))
modelB.add(Dense(16, activation='sigmoid'))
modelB.add(Dense(8, activation='sigmoid'))
modelB.add(Dense(4, activation='sigmoid'))
modelB.add(Dense(1, activation='sigmoid'))
```

```
In [36]: ## Model using linear activation (all neurons)
modelC = Sequential()
modelC.add(Dense(32, input_dim = len(X_train[0, :]), activation='linear'))
modelC.add(Dense(16, input_dim = len(X_train[0, :]), activation='linear'))
modelC.add(Dense(8, input_dim = len(X_train[0, :]), activation='linear'))
modelC.add(Dense(4, activation='linear'))
modelC.add(Dense(1, activation='linear'))
```

```
In [37]: modelD = Sequential()
modelD.add(Dense(32, input_dim = len(X_train[0, :]), activation='relu'))
modelD.add(Dense(16, activation='relu'))
modelD.add(Dense(8, activation='relu'))
modelD.add(Dense(4, activation='relu'))
modelD.add(Dense(1, activation='linear'))
```

Figure 6: models that were tested

5.3 Graph of Learning Curve of Chosen Baseline Model

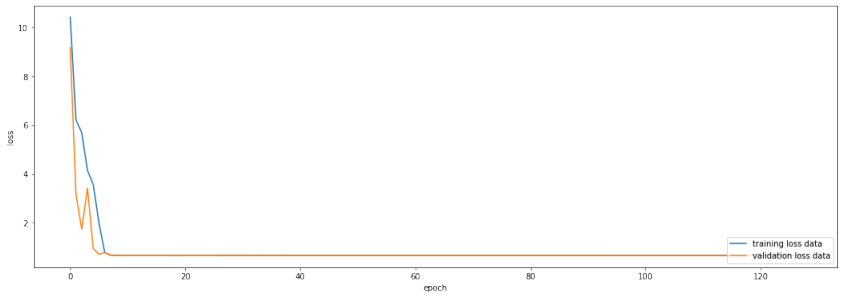

Figure 7: Curve of baseline model showing the epoch and loss

6 Comparison of Linear vs Sigmoid Functions

Activation Function	Accuracy	MAE	Loss
Sigmoid all layers	60.90 percent	-	66.83 percent
Sigmoid last layer	61.12 percent	-	66.40 percent
Linear all layers	-	45.84 percent	23.15 percent
Linear last layer	-	61.23 percent	62.25 percent

6.1 Graph of Learning Curve of all Sigmoid

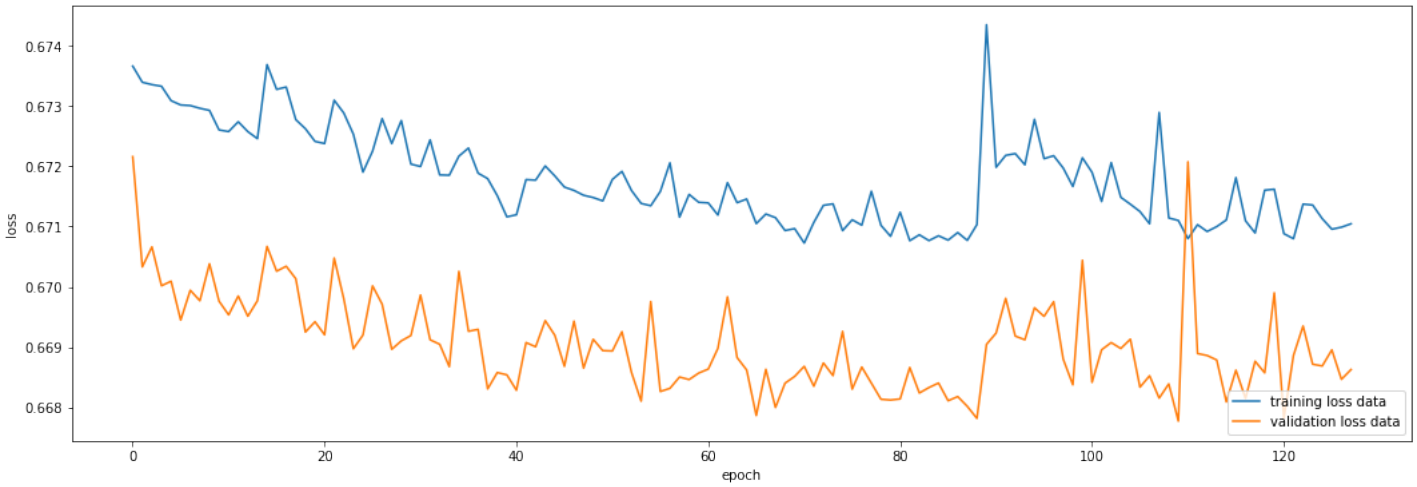

Figure 8: graph of all sigmoid functions in layers showing the epoch and loss

6.2 Graph of Learning Curve of Last Layer as Sigmoid

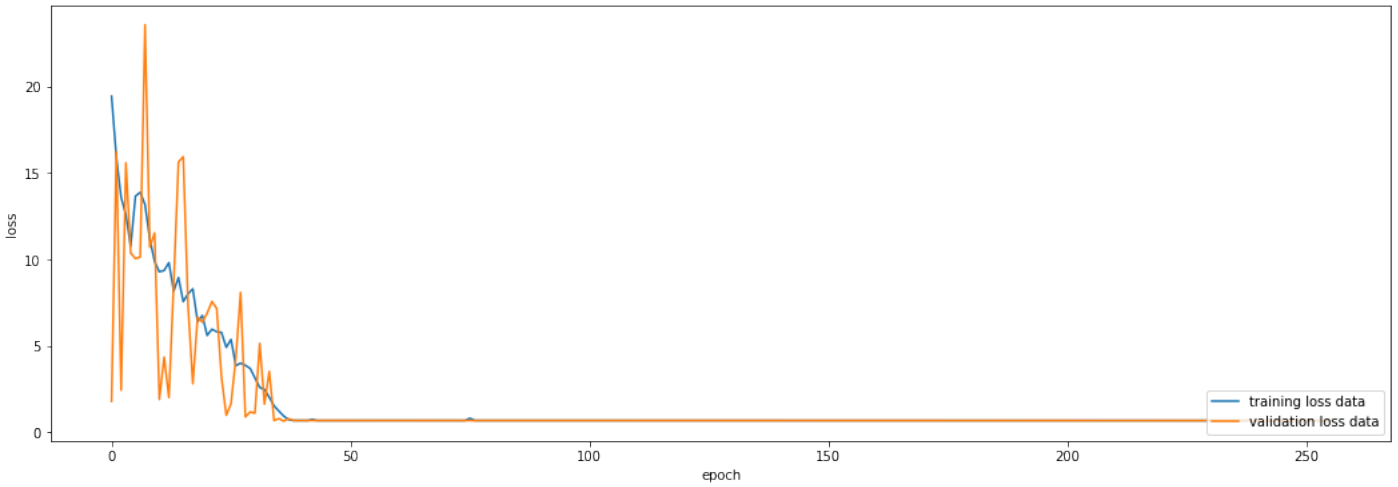

Figure 9: graph of sigmoid in last layer

6.3 Graph of Learning Curve of all Linear

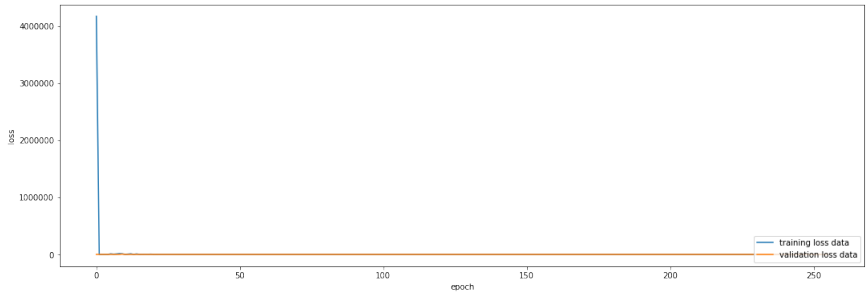

Figure 10: graph of all Linear functions in layers showing the epoch and loss

6.4 Graph of Learning Curve of Last Layer as Linear

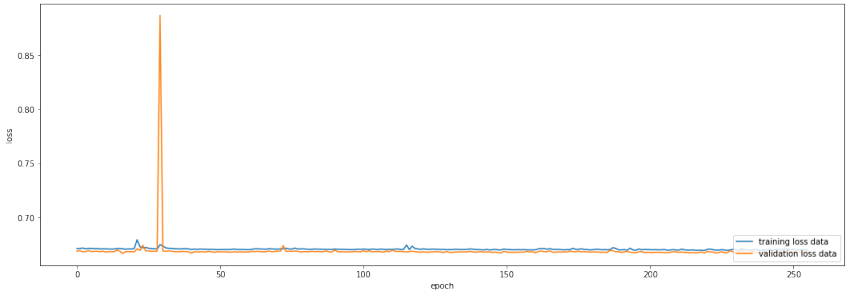

Figure 11: graph of linear function in last layer

6.5 Graph of Over-fitting example

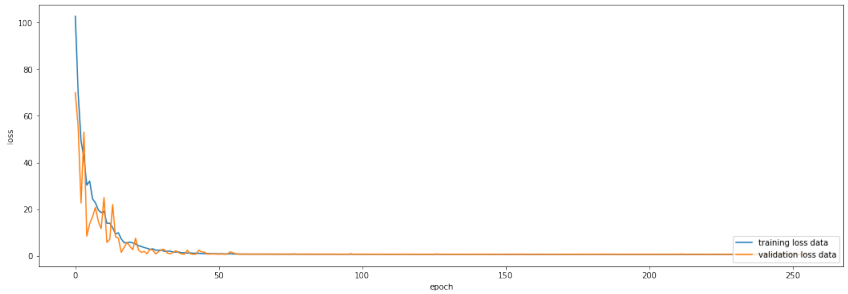

Figure 12: graph of over fitting example with 80 neurons

Although accuracy was much higher in the overfitting example, at 71.91 percent with 80 neurons, it is unknown whether or not it will perform good in the real world context

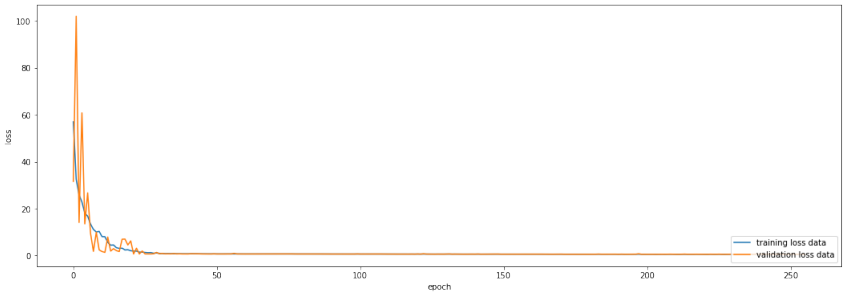

Figure 13: graph of over fitting example with 320 neurons

7 Model Comparison

Model	Precision	Recall	F1 score
Baseline 3 hidden layers	60.68 percent	99.99	0.76
Sigmoid last layer	60.43 percent	100 percent	0.75
Linear all layers	62.89 percent	79.74 percent	0.72
Linear last layer	60.43 percent	100 percent	0.75
Over fitting Mild	80.73 percent	82.16 percent	0.81
Over fitting Severe	100 percent	0.01 percent	0.00

8 Call Backs

Call backs were experimented with, specifically, early stopping. It helped decrease the run time by stopping early when accuracy reaches a plateau and right before it starts to possibly decline, therefore not having to run anymore epochs.

Phase III, Feature Importance and Reduction

9 Feature Importance

Each feature was ran on by itself as the sole variable for determining accuracy, highest accuracy features found were:

- diabetesMeds
- age
- time in hospital
- num inpatient

10 New model using only highest accuracy features

Using only diabetesMed, age, time in hospital and number inpatient as input features:

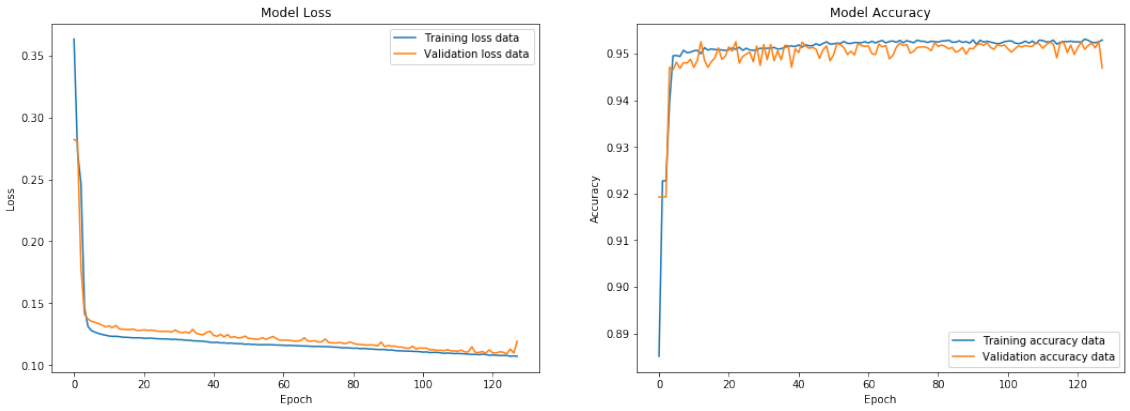

Figure 14: Graph of new model, achieved highest accuracy of **94 percent**, while previous all input model only had accuracy of about **61 percent**

11 Conclusion

There were many features in the data set that contributed to the label, however, it was very difficult to get high accuracy in predictions while using all of the features.
One of the issues I faced was dealing with the minority group in classification. Those that were readmitted were only 10 percent of the total. As a result, the model would just classify every input as "not readmitted" which would get roughly 90 percent accuracy, and would fail to classify a single input into the readmitted category. Many oversampling techniques were used and tested, the SMOTEENN performed the best. With oversampling, the model would make predictions with about 61 percent accuracy.
However, after identifying the key features:

- diabetesMeds -whether or not patient was prescribed diabetes meds
- age
- time in hospital - time in days patient spent in hospital
- num inpatient - number of time patient was seen in patient in this year

Using only the above features, I was able to get a much higher accuracy of 94 percent. Based on this dataset, if hospitals want to identify diabetes patients with high risk of readmission, I would recommend them to focus mostly on the 4 parameters above.

References

https://archive.ics.uci.edu/ml/datasets/diabetes+130-us+hospitals+for+years+1999-2008
https://medium.com/analytics-vidhya/diabetes-130-us-hospitals-for-years-1999-2008-e18d69beea4d
https://github.com/zegster/artificial-intelligence/blob/master/model_selection_and_evaluation/Model_Selection_Evaluation.pdf