REPORT

Table of Contents

[Team: = (3 members) 2](#_Toc23106359)

[Software Used: 2](#_Toc23106360)

[Analysis Process Used: 4](#_Toc23106361)

[Data Preparation & Data Cleaning: = 5](#_Toc23106362)

[Approaches Used and Reviews: 8](#_Toc23106363)

[Evaluation and Accuracy: 9](#_Toc23106364)

[References: 13](#_Toc23106365)

# Team: = (3 members)

|  |  |
| --- | --- |
| 1. Vidhan Dholakia | (20151326) |

# Software Used:

We have used Python 3.7 for the coding purpose with the PyCharm IDE. We also used TPU and GPU from Google Colab for acceleration purpose

Instructions on how to Download & Install the software and the data sets:

How to Download Python & Install

Step 1: = Go to <https://www.python.org/downloads/>

Step 2: = Select our version and directly download as per the requirement and system adaptability.



Step 3: = The installation process is quite simple and straightforward. Just need to follow the obvious steps. We have used PyCharm IDE for execution of Python.

**Using Google Colab:**

Step 1: Go to [https://colab.research.google.com](https://colab.research.google.com/)

**Installing Pandas, Numpy, Keras, Matplotlib:**

Pandas:

Install Python on your PC. Open the command prompt and type

*pip install pandas*

Numpy and Matplotlib can be similarly installed by using the following commands:

*pip install numpy*

*pip install Matplotlib*

For Keras, we need to first install the TensorFlow

*pip install --upgrade TensorFlow*

Now we can install Keras. All we have to do is import it.

*import Keras*

How to Download the Data Set?

We have been given the data set of the diabetes patients over a decade (1999 – 2008) at 130 US hospitals. It includes over 50 features representing patient and hospital outcomes.

# Analysis Process Used:

We have the data (train and test) about the diabetes patients over a decade (1999 – 2008) of clinical care at 130 US hospitals and integrated delivery network. We have various categorical as well nominal data in the data set.

Initially, we had the total data of 90767 items! This data set is not useful yet as it is not clean, and it is totally unfit for the data modelling processes.

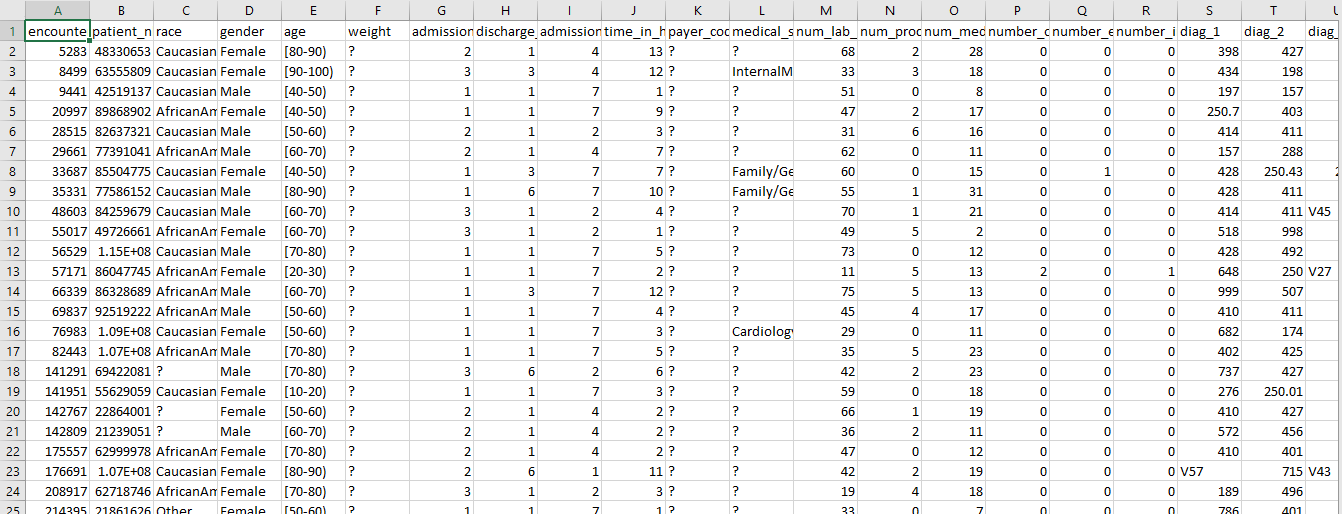


Fig. Trained Data Un-Cleaned

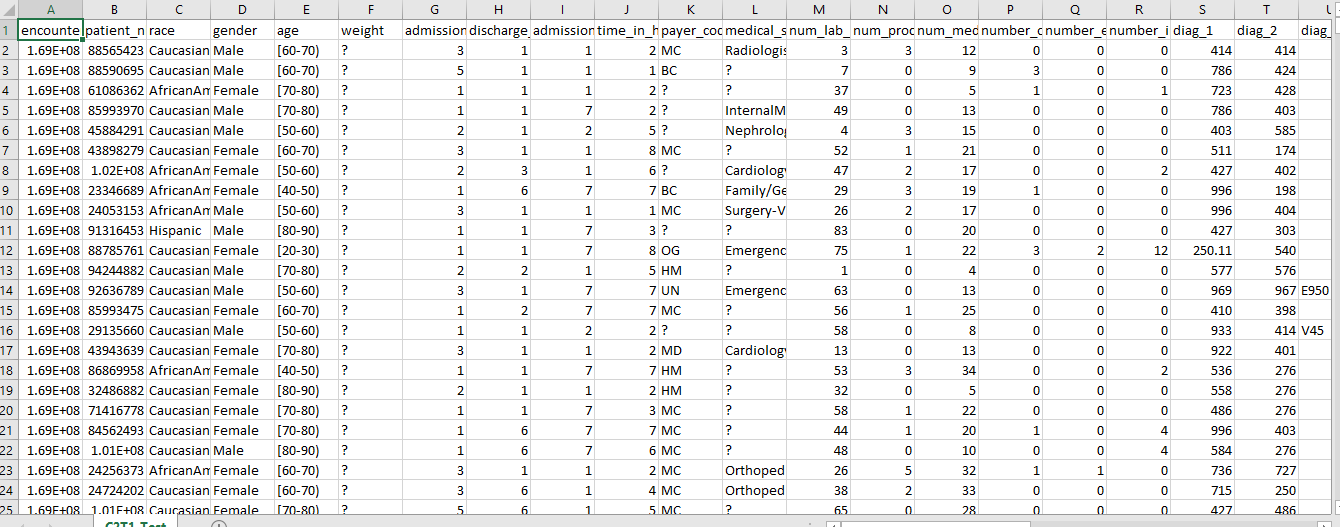


Fig. Test Data Un-Cleaned

# Data Preparation & Data Cleaning: =

We have updated our data as per our requirements. Firstly, we have cleaned the data by dealing with the missing values and bifurcating between the relevant and irrelevant features.

We changed the ‘change’ column values to 0 if value is ‘No’ else 1 if value is ‘Ch’

Discharge\_Disposition\_ID column value were changed from

* (1,6,7,8,13) were changed to 3
* (2,5,10,14,16,22,23,24,27,28,29,30) were changed to 1
* (3,4,9,12,15,17) were changed to 2
* (11,18,19,20,21,25,26) were changed to 4

*Admission\_type\_id* column values were changed from

* 1,2,7,3,4,5,6,8 to 1

*Admission\_Source\_Id* column values were changed from

* 2,3 to 1
* 5,6,10,18,22,25,26 to 2
* 9,15,17,20,21,11,13,14 to 3

Fore the *drugs* value, we changed

* None -99
* Norm to 0
* >200 to 1
* >300 to 1
* >7 to 1
* >8 to 1
* No to 0
* Steady to 1
* Up to 1
* Down to 1

The *age* range were change from

* 10-20 to 15
* 20-30 to 25
* 30-40 to 35
* 40-50 to 45
* 50-60 to 55
* 60-70 to 65
* 70-80 to 75
* 80-90 to 85
* 90-100 to 95

We also changed the *race* categorical values to numerical values as shown below:

* Asian' to 2
* Other to 3
* Hispanic' to 4
* AfricanAmerican to 5

In the *diag\_1, diag\_2, diag\_3* columns, if any value started with V, we assigned those values as 0

If the *diag\_1* value were

* >=390 and <460 or ==785: changed to 1
* >=460 and <520 or ==786: changed to 2
* >=520 and <580 or ==787: changed to 3
* >=800 and <999 changed to 5
* >=710 and <740 changed to 6
* >=580 and <630 or ==788: changed to 7
* >=140 and <240 or ==780/781/784: changed to 8
* >=790 and <800 : changed to 8
* >=240 and <280 : changed to 8
* >=680 and <710 or ==782 /781/784: changed to 8

Same changes were applied to *diag\_2* and *diag\_3* as well.

The above operations were applied to the test as well as train data:

As you can compare from the diagram 2 and diagram 4, we have removed the irrelevant features like “encounter id”, “weight”, “payer code” as they had many values as missing etc. The records were also eliminated who had missing values from the columns having few missing values as these columns couldn’t be removed.

Also, we have applied various feature engineering approaches on the relevant features as shown in the diagram 4. We have standardized our data set on both categorical as well numeric data. However, before this we had converted our categorical to numerical data.

The labels were initially labeled as 0 for patients not admitted. For patients admitted within 30 days and after 30 days, we labelled them as 1.

We also removed the records where patients were not admitted and then in the new data, we labelled the patients admitted before 30 days as 2 and after 30 days as 1. In this way, we won’t leak data of those patients who won’t be admitted into the hospital again. This helped us to reduce data and time as well and achieve accuracy higher than with compiled data. We also added 3 new columns. Which were the product of

* *time\_in\_hospital* and *num\_medications*
* *time\_in\_hospital* and *num\_lab\_procedures*
* *time\_in\_hospital* and number\_diagnoses

The data was then standardized for 5 columns namely

* *time\_in\_hospital*
* *num\_lab\_procedures*
* *num\_medications*
* *number\_diagnoses*
* *num\_procedures*

The above procedures were carried out in MS-Excel and were not done in Python. As we can compare from the diagram 3 and diagram 5, we have removed the irrelevant features like “weight”, “payer code”, etc.

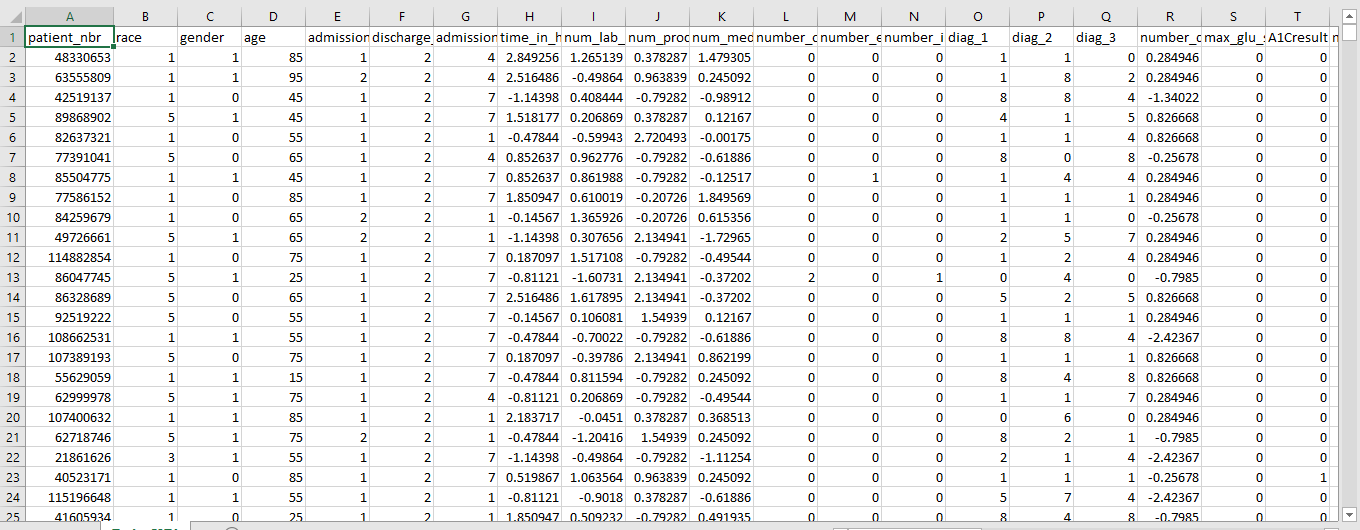


Fig. Cleaned Train Data

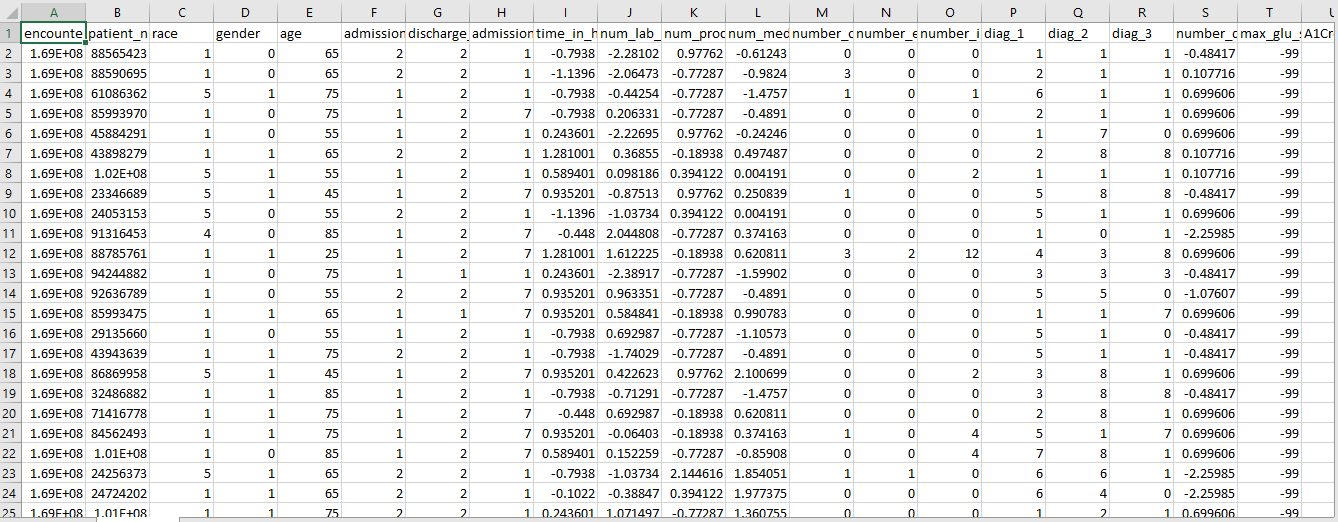


Fig. Cleaned Test Data

# Approaches Used and Reviews:

Our approach was divided into two categories.

* We trained the model to predict whether a patient will be admitted or not. This was done by setting the column ‘readmitted’ as 0 for ‘NO’ and 1 for ‘>30’ OR ‘<30’. Initially, we were only concerned with predicting whether a patient will go to hospital or not. To predict this, we used Random Forest algorithm with 10 trees with depth of 55. The impurity criteria were ‘gini’. With the Random forest algorithm, we were able to get a maximum accuracy of 87% as shown in the screenshot below. The output of the code whether a patient was admitted or not is saved as ***Admitted Or Not.csv***
* To predict whether a patient was admitted before 30 or after 30 days, we separated the patients who were admitted from the training data. This way, we won’t get any leaked data from the patients who didn’t go the hospital at all. This also gave us benefit of reduced data which inherently reduced our time. The new dataset had 40266 rows only. Now we trained the new dataset using a neural network. We have 45 features, which is the input layer. We decided to go with two hidden layers. The first layer consisted of 250 neurons and the second layer consisted with 140 neurons. We ran the code for 380 epochs with a batch size of 180. With the above approach, we got an output accuracy of 91% in average and the highest accuracy was 94.52%. The patients readmitted were saved into a separate file named ***Patients\_admit\_30\_days.csv*** The labels for these were save separately in a file named ***Patients\_admit\_30\_days\_labels.csv.*** We have added the files in the zip as well. The python code for the above process is named ***Within 30 Code Neural Network.py.*** The output of the code is saved as ***Output\_2\_Within\_30\_days.csv. The output label 1 indicates patients admitted after 30 days and output label 2 indicates admitted within 30 days.***

# Evaluation and Accuracy:

1. Predicting if the person is going to be readmitted or not

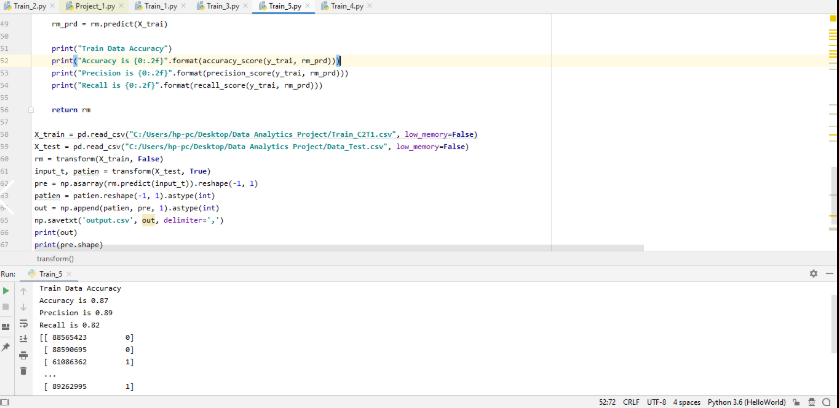
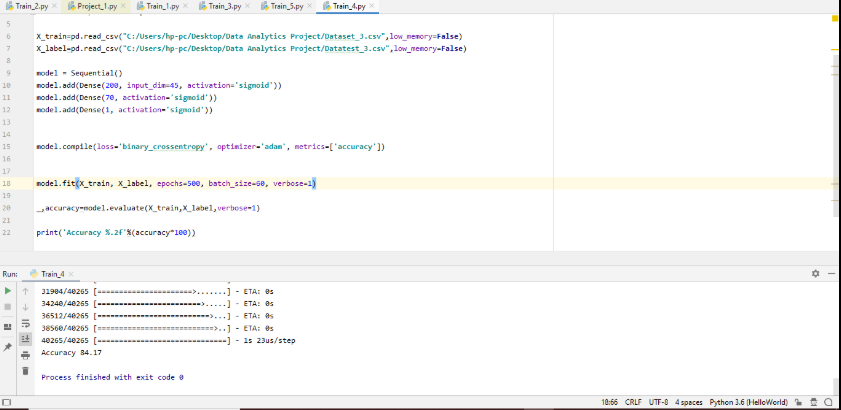


Fig. Training whether a patient will be admitted or not suing Random Forest

The accuracy obtained was 87% whether a person will be admitted or not.

1. Predicting if a person will be readmitted in less than 30 OR Greater than 30 days.



The below snapshot is of the code and the output of the Neural Network we used to predict the patients, whether they were admitted before 30 days or after 30 days. The input layer is of 45 node which are the features of the patients i.e. columns from test data. The first hidden layer node has 250 neurons. The second hidden layer node has 140 neurons. The final layer has 1 output.

A screenshot of a computer screen

Description automatically generated

Fig. Neural Network Code to predict patients within 30 or after 30 days.

A screenshot of a cell phone

Description automatically generated

Fig. Output Of NN with an accuracy of 95%

Training the separate data of admitted patients gave us an accuracy of 94.52% and an average of 91%.

We have 2 output files. The output of the code whether a patient was admitted or not is saved as ***Admitted Or Not.csv.*** The output of the code whether a patient was admitted within 30 or after 30 days is saved as ***Output\_2\_Within\_30\_days.csv***

.

# References:

1. <https://www.python.org/doc/>
2. <https://keras.io/>
3. <https://towardsdatascience.com/predicting-hospital-readmission-for-patients-with-diabetes-using-scikit-learn-a2e359b15f0>
4. <https://medium.com/berkeleyischool/how-to-use-machine-learning-to-predict-hospital-readmissions-part-1-bd137cbdba07>