Report

On

Contagion Prediction

Prepared by

Nisarg Chauhan (18012011010)

Guided By

Prof. Ketan J. Sarvakar

B.Tech Semester VII

Nov-Dec -2021

Subject: Deep Learning

Subject code: 2CEIT78PE1

Department of Computer Engineering

U. V. Patel College of Engineering

Ganpat University, Ganpat Vidyanagar – 384012

Abstract:

Our project object is to detect whether patients have heart disease or not by given a number of features from patients. The motivation of our project is to save human resources in medical centers and improve accuracy of diagnosis. In our project we use Logistic Regression to detect the heart disease and to find the patient is suffering from heart disease or Not.

Notebook used:

Google Colab - Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.

Tools and Libraries

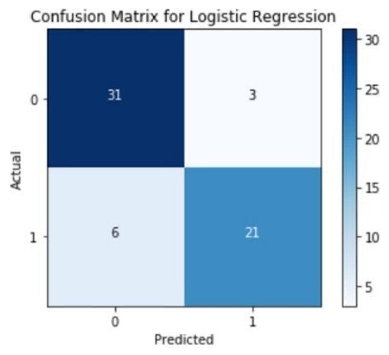
Tools and Libraries	Usage
Keras	Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras actsas an interface for the TensorFlow library.
seaborn	is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis. Seaborn works easily with dataframes and the Pandas library. The graphs created can also be customized easily
Numpy	We are using it for the Image matrix handling
Logistic Regression	Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

Logistic Regression is a supervised learning that computes the probabilities for classification problems with two outcomes. It can also be extended to predict several classes. In Logistic Regression model, we apply the sigmoid function, which is

$$\sigma(\mathrm{z}) = \frac{\&}{\& \cdot e|)^{*}}.$$

This function successfully maps any number into the value between 0 and 1 and we can regard this value as the probability of predicting classes. For example, we have two classes and they are presence of heart disease and absence of disease. If we set the threshold as 0.5, applying the sigmoid function gives us a value of 0.7, which means the man has the 70% probability of having heart disease so we will predict that he has heart disease.

Here is the confusion matrix of the Logistic Regression:



I used the L2 penalty, the square of the magnitude of coefficients, supported by Logistic Regression to avoid overfitting. The train accuracy is 83.88% and test accuracy is 85.25%. It performs well but not the best for us. The advantage of the Logistic Regression is that it does not need too much computational resources and it is highly interpretable. So it is easy and sufficient to apply Logistic Regression. However, thelimitation of Logistic Regression is that it assumes linearity between the features of the dataset. In the real world, the data is rarely separable, neither as our dataset. That is why we cannot reach a very high accuracy of 90 %.

Work Flow:

Step 1: Import Requires Libraries

Step 2: Get the data

Step 3: Split the data into train and test

Step 4: Split features from labels Train and test

Step 5: Normalize the data

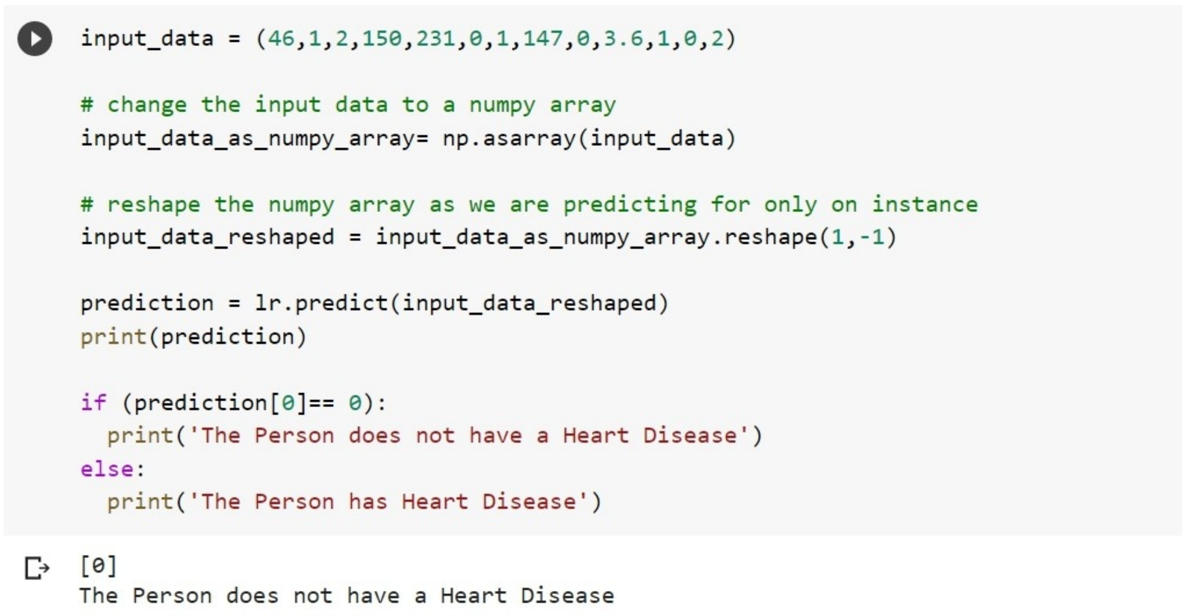
Step 6: Build the model

Step 7: Inspect the model

Step 8: Train the model for 300 epochs

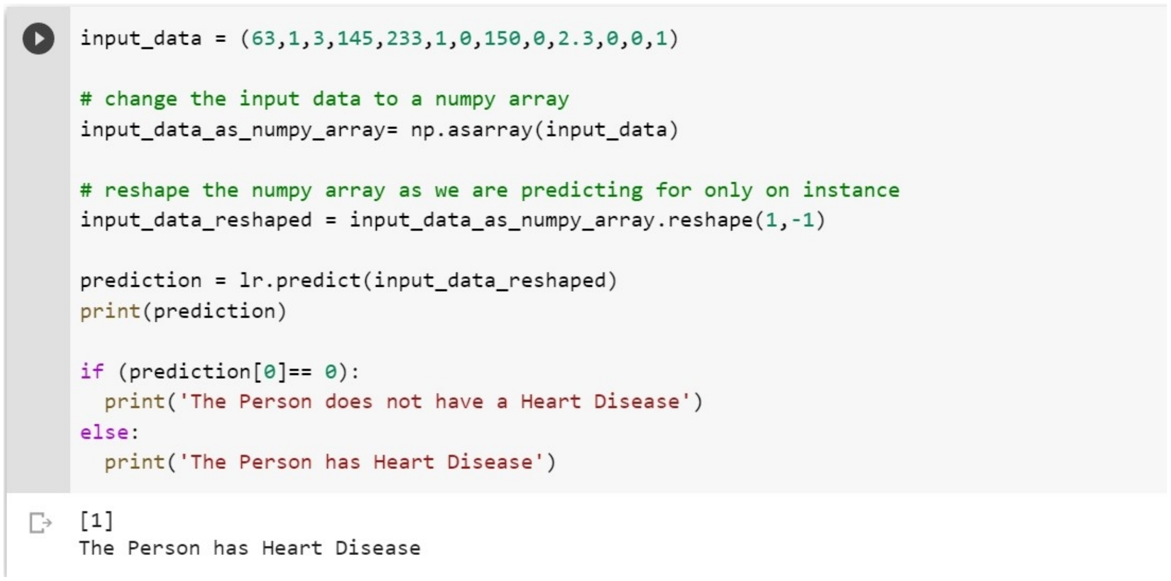
Step 9: Result Prediction

Output:



```
input_data = (46,1,2,150,231,0,1,147,0,3.6,1,0,2)

# change the input data to a numpy array
input_data_as_numpy_array= np.asarray(input_data)

# reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = lr.predict(input_data_reshaped)
print(prediction)

if (prediction[0]== 0):
  print('The Person does not have a Heart Disease')
else:
  print('The Person has Heart Disease')
```

```
[0]
The Person does not have a Heart Disease
```



```
input_data = (63,1,3,145,233,1,0,150,0,2.3,0,0,1)

# change the input data to a numpy array
input_data_as_numpy_array= np.asarray(input_data)

# reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = lr.predict(input_data_reshaped)
print(prediction)

if (prediction[0]== 0):
  print('The Person does not have a Heart Disease')
else:
  print('The Person has Heart Disease')
```

```
[1]
The Person has Heart Disease
```

Dataset Name:

heart.csv