Diabetes Onset Prediction Using Deep Learning

A project report submitted for the partial fulfillment of the

Bachelor of Technology Degree

in

Computer Science & Engineering

under

Maulana Abul Kalam Azad University of Technology

BY

**SHRADDHA KANTAL**

(ROLL NO : 10400116090, REGISTRATION NO : 161040110157)

**&**

**SOUJANNYA ROY**

(ROLL NO : 10400216052, REGISTRATION NO: 161040110320)

Under the Guidance of:

**Prof. Srijita Basu**

Department of Computer Science

For the Academic Year 2016-2020.



Department of Computer Science and Engineering

Institute of Engineering & Management

Y-12, Salt Lake, Sector 5, Kolkata, Pin 700091, West Bengal, India

Affiliated To



Maulana Abul Kalam Azad University of Technology

BF 142, BF Block, Sector 1, Kolkata, West Bengal 700064

May 2019



CERTIFICATE

TO WHOM IT MAY CONCERN

This is to certify that the project report titled “Diabetes Onset Prediction Using Deep Learning”, submitted by

1. Shraddha Kantal

(Registration No. 161040110157 of 2016-2020 Roll no. 10400116090),

2. Soujannya Roy

(Registration No. 161040110320 of 2016-2020 Roll no. 10400216052)

,students of Institute of Engineering & Management in partial fulfillment of requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering, is a bona fide work carried out under the supervision of Prof. Srijita Basu during the final year of the academic session of 2016-2020. The content of this report has not been submitted to any other university or institute for the award of any other degree.

It is further certified that the work is entirely original and the performance has been found to be satisfactory.

Prof. Srijita Basu Prof.(Dr.) Sourav Saha

Project Manager H.O.D.

Department of Computer Science and Engineering Department of Computer Science and Engineering

Institute of Engineering & Management Institute of Engineering & Management

Prof.(Dr.) Amlan Kusum Nayak

Principal

Institute of Engineering & Management



i

INSTITUTE OF ENGINEERING & MANAGEMENT



DECLARATION

FOR NON-COMMITMENT OF PLAGIARISM

We, Shraddha Kantal, Soujannya Roy, students of B.Tech. in the Department of Computer Science and Engineering, Institute of Engineering & Management have submitted the project report in partial fulfillment of the requirements to obtain the above noted degree. We declare that we have not committed plagiarism in any form or violated copyright while writing the report and have acknowledged the sources and/or the credit of other authors wherever applicable. If subsequently it is found that we have committed plagiarism or violated copyright, then the authority has full right to cancel/reject/revoke our degree.

Name of the Student: SHRADDHA KANTAL

Full Signature:

Name of the Student: SOUJANNYA ROY

Full Signature:

Date:

ii

Abstract

Diabetes is considered as one of the deadliest and chronic diseases which causes an increase in blood sugar. Many complications occur if diabetes remains untreated and unidentified. The tedious identifying process results in visiting of a patient to a diagnostic centre and consulting doctor.

Various traditional methods, based on physical and chemical tests, are available for diagnosing diabetes. However, early prediction of diabetes is quite challenging task for medical practitioners due to complex interdependence on various factors as diabetes affects human organs such as kidney, eye, heart, nerves, foot etc. Data science methods have the potential to benefit other scientific fields by shedding new light on common questions. One such task is to help make predictions on medical data. But the rise in machine learning approaches solves this critical problem.

Machine learning is one of the aspects of artificial intelligence that allows the development of computer systems that have the ability to learn from experiences without being the need of programming it for every instance. Machine learning is dire need of today’s scenario to eliminate human effort as well as come up with higher automation with less errors. The motive of this study is to design a model which can prognosticate the likelihood of diabetes in patients with maximum accuracy. Experiments are performed on Pima Indians Diabetes Database (PIDD) which is sourced from UCI machine learning repository in a systematic manner.

iii

Acknowledgements

We must not forget to acknowledge everyone who has provided constant support to us during our B.Tech course. First and foremost, we would like to express sincere gratitude to our supervisor Prof. Srijita Basu for her continuous support and motivation in fuelling the pursuance of carrying out this project endeavour. Without her guidance and persistent encouragement, this project work would not have been possible. She has been a tremendous mentor for us throughout this academic journey. Many of her academic advises about our career growth have been priceless.

We would like to convey sincere gratitude to Prof. Sourav Saha for providing us constant inspiration to stand firm against several setbacks throughout the course. Additionally, we would like to thank all the technical, non-technical and office staffs of our department for extending facilitating cooperation wherever required. We also express gratitude to all of our friends in the department for providing the friendly environment to work on the project work.

We would also like to thank our Director Prof. Satyajit Chakraborti for providing us an outstanding platform in order to develop our academic career. In addition, we also preserve a very special thankful feeling about our Principal Prof. Amlan Kusum Nayak for being a constant source of inspiration.

A special thank is due to our family. Words cannot express how grateful we are to our parents for all the sacrifices that they have made while giving us necessary strength to stand on our own feet.

Finally, we would like to thank everybody who has provided assistance, in whatever little form, towards successful realization of this project but with an apology that we could not mention everybody’s name individually.

iv

Contents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [List of](#page6) | | [Figures](#page6) |  | vi |
| [List of Tables](#page7) | | |  | vii |
| [1](#page9) | [Introduction](#page9) | |  | 1 |
|  | [1.1](#page9) | [Motivation](#page9) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 1 |
|  | [1.2](#page10) | [Related Work](#page10) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 2 |
|  | [1.3](#page11) | [Objective of the Proposed Work](#page11) . . . . . . . . . . . . . . . . . . . . . . | | 4 |
|  | [1.4](#page11) | [Organization of the Project Report](#page11) . . . . . . . . . . . . . . . . . . . . . | | 4 |
| [2](#page13) | [Proposed Framework](#page13) | | | 5 |
|  | [2.1](#page13) | [Overview of Proposed Framework](#page13) . . . . . . . . . . . . . . . . . . . . . | | 5 |
|  | [2](#page13).2 | Background study . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 5 |
|  | [2.3](#page13) | [Methods and Materials](#page13) . . . . . . . . . . . . . . . . . . . . . . . . . . | | 6 |
|  |  | [2.3.1](#page14) | [Data Collection](#page14) . . . . . . . . . . . . . . . . . . . . . . . . . .. . . . . . . | 7 |
|  |  | [2.3.2](#page17) | [Data Preparation](#page17) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 7 |
|  |  | 2.3.3 | Implement Deep Neural Network . . . . . . . . . . . . . . . . . . | 7 |
|  |  | [2.3.](#page21)4 | [Predict the Output](#page21) . . . . . . . . . . . . . . . . . . . . . . . . | 8 |
| [3 Experimental Results and Discussion](#page28) | | | | 10 |
|  | [3.1](#page28) | [Experimental Setup](#page28) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 10 |
|  | 3.2 | Implementation . .. . . .. . . . . . . . . . . . . . . . . . . . . . . .. . . . | |  |
|  | [3.3](#page28) | [Experimental Results](#page28) . . . . . . . . . . . . . . . . . . . . . . . . . . . . | | 17 |
|  |  |  | |  |
| [4](#page33) | [Conclusion](#page33) | |  | 18 |
|  | | |  |  |
| [Bibliography](#page36) | | |  | 19 |

v

List of Figures

|  |  |  |
| --- | --- | --- |
| [1.](#page12)2 | Age vs Passing from Hypertension. . . . . . . . . . . . . . . . . . . | 2 |
| [2.](#page14)2 | [Deep Neural Network](#page14) . . . . . . . . . . . . . . . . . . . | 6 |
| [2.](#page14)1 | Flowchart. . . . . . . . . . . . . . . . . | 9 |
| [3.3](#page20) | [Prediction on Example dataset](#page20) . . . . . . . . . . . . . . . . . . . . | 17 |

vi

List of Tables

|  |  |  |
| --- | --- | --- |
| [3](#page26).3 | Prediction Accuracy. . . . . . . . . . . . . . . . . . . . . . | 17 |

vii

1

Introduction

**1.1 Motivation**

Diabetes is a chronic disease or group of metabolic disease where a person suffers from an extended level of blood glucose in the body, which is either the insulin production is inadequate, or because the body’s cells do not respond properly to insulin. The constant hyperglycaemia of diabetes is related to long-haul harm, brokenness, and failure of various organs, particularly the eyes, kidneys, nerves, heart, and veins. The objective of this research is to make use of significant features, design a prediction algorithm using Deep learning and find the optimal classifier to give the closest result comparing to clinical outcomes. The normal identifying process is that patients need to visit a diagnostic centre, consult their doctor, and sit tight for a day or more to get their reports. Moreover, every time they want to get their diagnosis report, they have to waste their money in vain. DM is one of the most common endocrine disorders, affecting more than 200 million people worldwide. The onset of diabetes is estimated to rise dramatically in the upcoming years. DM can be divided into several distinct types. However, there are two major clinical types, type 1 diabetes (T1D) and type 2 diabetes (T2D), according to the etiopathology of the disorder. T2D appears to be the most common form of diabetes (90% of all diabetic patients), mainly characterized by insulin resistance. The main causes of T2D include lifestyle, physical activity, dietary habits and heredity, whereas T1D is thought to be due to auto immunological destruction of the Langerhans islets hosting pancreatic-β cells. T1D affects almost 10% of all diabetic patients worldwide, with 10% of them ultimately developing idiopathic diabetes. Other forms of DM, classified on the basis of insulin secretion profile and/or onset, include Gestational Diabetes, endocrinopathies, MODY (Maturity Onset Diabetes of the Young), neonatal, mitochondrial, and pregnancy patients at risk of having Diabetes Mellitus based on patient demographic data and the laboratory results during their visits to medical facilities. Machine learning is

1

**1.2. RELATED WORK**

one of the aspect of artificial intelligence that allows the development of computer systems that have the ability to learn from experiences without being the need of programming it for every instance. Machine learning is dire need of today’s scenario to eliminate human effort as well as come up with higher automation with less errors.

**1.2 Related Work**

Data mining techniques have been largely explored in predicting diabetes mellitus. As medical records are more in number and keep growing rapidly due to the emergence of various diseases, extraction of knowledge from a large database has become a typical process. It is important to choose appropriate algorithms for prediction purpose, especially in the case of clinical data analysis. In several works, it has been reported that decision tree achieves higher accuracy in deriving useful information from multiple sources. However, it is also reported that decision trees do not yield the desired accuracy in the presence of sparse data. Few authors have performed a comparative study on decision trees and KNN for diabetes prediction, and it was demonstrated KNN is advantageous in the case of large datasets. The simplicity of the KNN algorithm has also made it an attractive candidate for building predictive models. However the drawbacks of KNN associated with the lazy learning principle should be alleviated for the algorithm to be of practical utility for prediction of diabetes.

The annual report of World Health Association, add up to the number of individuals experiencing diabetes is 422 million the year (Write from which year to which year it is telling the statistics of 422 million). Consistently, there is a significant increment in the number individuals experiencing diabetes in different healing centre. The world health organization (WHO) reports on “Diabetes Care 2018” by American Diabetes Association and Standards for Medical care in Diabetes, a study for correlation diverse races and their pay.

### **This Figure demonstrates the diverse individuals (gender and wage) matured between 29 and 70 years, level of passing because of hypertension**

Fig 1.2: individuals (gender and wage) matured between 29 and 70 years vs level of passing because of hypertension

2

**1.2 RELATED WORK**

The author in [12] evaluated the performance of the machine learning algorithms for the prediction of diabetes mellitus. The algorithms used by the systems are support vector machine, artificial neural network, logistic regression, classification tree, and K-nearest neighbour. The performance of the system is appraised in terms of accuracy, specificity, sensitivity, precision, negative predictive value, false positive rate, rate of misclassification, F1 measure and receiver operating characteristic (roc) curve. The highest accuracy of 78% and rate of misclassification 0.22 is obtained by the system using Logistic Regression. The better precision and negative predictive value are of 82% and 73% using Naïve Bayes and Logistic Regression respectively. The dataset is split into tenfold cross-validation manner. Author in [13] compared multiple classification algorithms for the classification of diabetes in Iran. The algorithms that have been used by the system are artificial neural network, Bayesian network, decision tree, support vector machine, and nearest neighbors. The accuracy of 97.44% is obtained by the system using artificial neural network that demonstrated the best performance. The accuracy of 81.19, 90.85, 95.03, and 91.60 % is obtained by support vector machine, 5- nearest neighbor, decision tree, and Bayesian network respectively. The dataset that has been used for the system is for 2536 cases screened for type 2 diabetes, within the metropolis of Tabriz, Iran that was gathered from the Tabriz University of Medical Sciences during a three-month program in spring 2010. The authors used WEKA as a simulation tool.

Author in [14] proposed a prediction framework for the diabetes mellitus using deep learning approach where the overfitting is diminished by using the dropout method. There are two fully connected layers each trailed by a dropout layer. The decision is found from the output layer with a single node. The system is applied to the Pima Indian diabetes dataset and the highest accuracy obtained by the system is 88.41%. Author at [15] proposed a system using multiple classifiers and improved the accuracy of complex disease prediction like diabetes. They proposed a dynamic weighted voting scheme for that system. The system is tested on T2DM data sets and Pima Indian diabetes dataset. The highest accuracy obtained by the system is 93.45% using MFWC with k=10 on Pima Indian diabetes dataset.

Author at [16] used data mining techniques to predict diabetes mellitus. They extract knowledge from the dataset and understandable description of patterns. The system obtained the highest accuracy 99.51% using Bayesian network. Author in [17] proposed a system to predict the diabetes diagnosis using K-Means, Genetic algorithm, and SVM (Support Vector Machine). The system trailed the following steps. First step, update all the missing values with the mean. Second step, the cleaned dataset is clustered using K-Means to eliminate outliers and unnecessary data and select the optimal feature using genetic algorithm to reduce the features. The highest accuracy of the system is 98.82% using SVM.

3

**1.3. OBJECTIVE OF PROPOSED WORK**

Author in [18] proposed a system that uses recursive feature elimination and principal component analysis for prediction of diabetes. They classify diabetes using deep neural networks and artificial neural networks. Using deep neural network their accuracy was 82.67% and using artificial neural network their accuracy was 78.62%. Goncalves et al. [19] introduced a system to predict diabetes using hierarchical Neuro-Fuzzy BSP method. They propose a new hierarchical neuro-fuzzy binary space partitioning (BSP) model devoted to pattern classification and rule extraction. They found the accuracy of 80.08% in the training set and 78.26% in the testing set. Han et al. [20] introduced the pair-wise and size-constrained K-means method to screen the high-risk population of diabetes mellitus.

**1.3 Objective of the Proposed Work**

The objective of this study was to build an effective predictive model with high sensitivity and selectivity to better identify patients at risk of having Diabetes Mellitus based on patient demographic data and the laboratory results during their visits to medical facilities. Next, we have to define grid search parameters. The goal of training is to answer a question or make a prediction correctly as often as possible .After compiling the model, we have to build and fill GridSearchCV. With that we find best batch size and ideal number of epochs. GridSearchCV lets us combine an estimator with a grid search preamble to tune hyper-parameters. The method will pick the optimal parameter from the grid search and use it with the estimator selected by the user. The goal is then to train the model and evaluate it e.g. using cross-validation. Cross-validation is a method for robustly estimating test-set performance (generalization) of a model. Grid-search is a way to select the best of a family of models, parameterized by a grid of parameters. Then,we evaluate the model to "measure" objective performance of the model .Testing the model is against previously unseen data. Next, further (test set) data which have, until this point, been withheld from the model (and for which class labels are known), will be used to test the model to get a better approximation of how the model will perform in the real world.

**1.4 Organization of the Project Report**

Rest of the report is organized as follows. In Chapter 2, the proposed technique is detailed focusing on how the project is performed on Jupyter Notebook using Keras library. The subsequent chapter to the detailed description has aimed to produce a report on how efficiently our framework has performed in obtaining the prediction. The concluding chapter summarizes the proposed work.

4

2

Proposed Framework

**2.1 Overview of Proposed Framework**

This section discusses our proposed framework in detail.

Classification is one of the most important decision making techniques in many real world problem. In this work, the main objective is to classify the data as diabetic or non-diabetic and improve the classification accuracy. For many classification problem, the higher number of samples chosen but it doesn’t leads to higher classification accuracy. In many cases, the performance of

algorithm is high in the context of speed but the accuracy of data classification is low. The main objective of our model is to achieve high accuracy. Classification accuracy can be increase if we use much of the data set for training and few data sets for testing. This survey has analyzed various classification techniques for classification of diabetic and non-diabetic data. Thus, it is observed that techniques like Support Vector Machine, Logistic

Regression, and Artificial Neural Network are most suitable for implementing the Diabetes prediction system.

Fig. [2.1](#page14) intuitively demonstrates the overall flow of the proposed strategy. The description of each step of the overall flow is detailed next.

**2.2 Background Study**

A deep neural network (DNN) is an [artificial neural network](https://en.wikipedia.org/wiki/Artificial_neural_network) (ANN) with multiple layers between the input and output layers. The DNN finds the correct mathematical manipulation to turn the input into the output, whether it be a [linear relationship](https://en.wikipedia.org/wiki/Linear_relationship) or a non-linear relationship. The network moves through the layers calculating the probability of each output. The user can review the results and select which probabilities the network should display (above a certain threshold, etc.) and return the proposed label. Each mathematical manipulation as such is considered a layer, and complex DNN have many layers, hence the name "deep" networks.

5

**2.3 METHODS AND MATERIALS**

DNNs can model complex non-linear relationships. DNN architectures generate compositional models where the object is expressed as a layered composition of [primitives](https://en.wikipedia.org/wiki/Primitive_data_type). The extra layers enable composition of features from lower layers, potentially modeling complex data with fewer units than a similarly performing shallow network.

Deep architectures include many variants of a few basic approaches. Each architecture has found success in specific domains. It is not always possible to compare the performance of multiple architectures, unless they have been evaluated on the same data sets.

DNNs are typically feedforward networks in which data flows from the input layer to the output layer without looping back. At first, the DNN creates a map of virtual neurons and assigns random numerical values, or "weights", to connections between them. The weights and inputs are multiplied and return an output between 0 and 1. If the network did not accurately recognize a particular pattern, an algorithm would adjust the weights.That way the algorithm can make certain parameters more influential, until it determines the correct mathematical manipulation to fully process the data.

In deep neural network, activation function performs a vital role. Activation functions are required to implement complicated mapping features that are non-linear in order to bring in the plenty-needed non-linearity property that permits them to approximate any function. Activation functions are also critical for squashing the limitless linearly weighted sum from neurons. It is also necessary to keep away from massive values gathering high up the processing hierarchy.

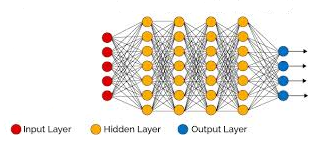


Fig 2.2: Deep Neural Network

**2.3 Methods and Materials**

To predict diabetes mellitus using deep learning, we

follow the following steps.

1. Data collection,
2. Data preparation,
3. Implement deep neural network and
4. Predict the output.

6

**2.3 METHODS AND MATERIALS**

**2.3.1 Data Collection**

The Pima Indian diabetes dataset is retrieved from the UCI machine learning repository database. Within the dataset, all the patients are female and minimum of 21 years old. There are 768 samples and also the sample is split into 8 attributes. Finally, the 9th attribute is the class distribution. Class 1 indicate that the diabetes test is positive and class 0 indicate the opposite. In the dataset, the number of instances of each class is (i) Class 0: Number of instances 500 (65.1%) (ii) Class 1: Number of instances 268 (34.9%). In Fig. 2 and Fig. 3 indicates the correlation between the eight parameters of “absence” and “present” samples where the high correlation between the parameters of these two classes of samples. We easily found out that in Fig. 2 that the “number of pregnant” parameter is negatively correlated with “triceps skin”, “serum insulin”, and “pedigree function”. All other parameters are positively correlated with the “number of pregnant”. Similarly, all the other parameters positively or negatively correlated when the class is “absence”. As an example, “number of pregnant” is positively correlated with all the parameters except “plasma glucose”, triceps skin”, “serum insulin”, “body mass”, and “pedigree function” when the class is “present”.

**2.3.2. Data Preparation**

The performance of any system depends on the standard of the data. So, we check the dataset that there any missing value is present or not. Then we randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data.

After that we convert dataset to **NumPy** array which is a general-purpose array-processing package. Then data is split into input x and output y sets.

Then we used StandardScalar to the data. The idea behind StandardScaler is that it will transform the data such that its distribution will have a mean value 0 and standard deviation of 1. Given the distribution of the data, each value in the dataset will have the sample mean value subtracted, and then divided by the standard deviation of the whole dataset.

**2.3.3. Implement Deep Neural Network**

We create and compile the model and prepare it for training.

•The goal of training is to answer a question or make a prediction correctly as often as possible. Each iteration of process is a training step.

7

**2.3. METHODS AND MATERIALS**

• First, we have to define grid search parameters. We build and fill GridSearchCV. With that we find best batch size and ideal number of epochs. GridSearchCV lets us combine an estimator with a grid search preamble to tune hyper-parameters. The method picks the optimal parameter from the grid search and uses it with the estimator selected by the user. The goal is then to train the model and evaluate it e.g. using cross-validation. Cross-validation is a method for robustly estimating test-set performance (generalization) of a model. Grid-search is a way to select the best of a family of models, parameterized by a grid of parameters.

**2.3.4 Predict the output**

Next, further (test set) data which have, until this point, been withheld from the model (and for which class labels are known), are used to test the model to get a better approximation of how the model will perform in the real world.

8

**2.3 METHODS AND MATERIALS**

Start

Get Data

Clean Data

Randomize data

Create and compile model

Find best batch size and ideal no. of epochs using GridSearchCV

Choose model

Use StandardScalar on data

Normalize using sklearn

Training data set seyset

Split into input and output sets

Testing data set

Convert dataset to NumPy array

Test the model

Train model

End

Figure 2.1: Flowchart

9

3

Experimental Results and Discussion

**3.1 Experimental Setup**

The project is performed on Jupyter Notebook.

JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones.

We use Keraswhich is a model-level library, providing high-level building blocks for developing deep learning models. At this time, Keras has two backend implementations available: the TensorFlow backend and the Theano backend. We have used Theono backend in this project. Experiments are performed on **Pima Indians Diabetes Database (PIDD)** which is sourced from UCI machine learning repository in a systematic manner.

**3.2 Implementation**

import sys

import pandas

import numpy

import sklearn.0

import keras

print('Pandas: {}'.format(pandas.\_\_version\_\_))

print('Numpy: {}'.format(numpy.\_\_version\_\_))

print('Sklearn: {}'.format(sklearn.\_\_version\_\_))

print('Keras: {}'.format(keras.\_\_version\_\_))

10

**3.2 IMPLEMENTATION**

import pandas as pd

import numpy as np

# import the uci pima indians diabetes dataset

url = "http://archive.ics.uci.edu/ml/machine-learning-databases/pima-indians-diabetes/pima-indians-diabetes.data"

names = ['n\_pregnant', 'glucose\_concentration', 'blood\_pressuer (mm Hg)', 'skin\_thickness (mm)', 'serum\_insulin (mu U/ml)',

'BMI', 'pedigree\_function', 'age', 'class']

df = pd.read\_csv(url, names = names)

# Describe the dataset

df.describe()

df[df['glucose\_concentration'] == 0]

# Preprocess the data, mark zero values as NaN and drop

columns = ['glucose\_concentration', 'blood\_pressuer (mm Hg)', 'skin\_thickness (mm)', 'serum\_insulin (mu U/ml)', 'BMI']

for col in columns:

df[col].replace(0, np.NaN, inplace=True)

df.describe()

# Drop rows with missing values

df.dropna(inplace=True)

# summarize the number of rows and columns in df

df.describe()

# Convert dataframe to numpy array

dataset = df.values

print(dataset.shape)

# split into input (X) and an output (Y)

X = dataset[:,0:8]

Y = dataset[:, 8].astype(int)

print(X.shape)

print(Y.shape)

print(Y[:5])

# Normalize the data using sklearn StandardScaler

from sklearn.preprocessing import StandardScaler

11

**3.2 IMPLEMENTATION**

scaler = StandardScaler().fit(X)

print(scaler)

# Transform and display the training data

X\_standardized = scaler.transform(X)

data = pd.DataFrame(X\_standardized)

data.describe()

# import necessary sklearn and keras packages

from sklearn.model\_selection import GridSearchCV, KFold

from keras.models import Sequential

from keras.layers import Dense

from keras.wrappers.scikit\_learn import KerasClassifier

from keras.optimizers import Adam

# Start defining the model

def create\_model():

# create model

model = Sequential()

model.add(Dense(8, input\_dim = 8, kernel\_initializer='normal', activation='relu'))

model.add(Dense(4, input\_dim = 8, kernel\_initializer='normal', activation='relu'))

model.add(Dense(1, activation='sigmoid'))

# compile the model

adam = Adam(lr = 0.01)

model.compile(loss = 'binary\_crossentropy', optimizer = adam, metrics = ['accuracy'])

return model

model = create\_model()

print(model.summary())

# Do a grid search for the optimal batch size and number of epochs

# import necessary packages

from sklearn.model\_selection import GridSearchCV

from sklearn.model\_selection import KFold

from keras.models import Sequential

from keras.layers import Dense

from keras.wrappers.scikit\_learn import KerasClassifier

from keras.optimizers import Adam

# Define a random seed

seed = 6

np.random.seed(seed)

12

**3.2 IMPLEMENTATION**

# Start defining the model

def create\_model():

# create model

model = Sequential()

model.add(Dense(8, input\_dim = 8, kernel\_initializer='normal', activation='relu'))

model.add(Dense(4, input\_dim = 8, kernel\_initializer='normal', activation='relu'))

model.add(Dense(1, activation='sigmoid'))

# compile the model

adam = Adam(lr = 0.01)

model.compile(loss = 'binary\_crossentropy', optimizer = adam, metrics = ['accuracy'])

return model

# create the model

model = KerasClassifier(build\_fn = create\_model, verbose = 1)

# define the grid search parameters

batch\_size = [10, 20, 40]

epochs = [10, 50, 100]

# make a dictionary of the grid search parameters

param\_grid = dict(batch\_size=batch\_size, epochs=epochs)

# build and fit the GridSearchCV

grid = GridSearchCV(estimator = model, param\_grid = param\_grid, cv = KFold(random\_state=seed), verbose = 10)

grid\_results = grid.fit(X\_standardized, Y)

# summarize the results

print("Best: {0}, using {1}".format(grid\_results.best\_score\_, grid\_results.best\_params\_))

means = grid\_results.cv\_results\_['mean\_test\_score']

stds = grid\_result.cv\_results\_['std\_test\_score']

params = grid\_results.cv\_results\_['params']

for mean, stdev, param in zip(means, stds, params):

print('{0} ({1}) with: {2}'.format(mean, stdev, param))

# Do a grid search for learning rate and dropout rate

# import necessary packages

from keras.layers import Dropout

# Define a random seed

seed = 6

np.random.seed(seed)

# Start defining the model

def create\_model(learn\_rate, dropout\_rate):

13

**3.2 IMPLEMENTATION**

# create model

model = Sequential()

model.add(Dense(8, input\_dim = 8, kernel\_initializer='normal', activation='relu'))

model.add(Dropout(dropout\_rate))

model.add(Dense(4, input\_dim = 8, kernel\_initializer='normal', activation='relu'))

model.add(Dropout(dropout\_rate))

model.add(Dense(1, activation='sigmoid'))

# compile the model

adam = Adam(lr = learn\_rate)

model.compile(loss = 'binary\_crossentropy', optimizer = adam, metrics = ['accuracy'])

return model

# create the model

model = KerasClassifier(build\_fn = create\_model, epochs = 100, batch\_size = 20, verbose = 0)

# define the grid search parameters

learn\_rate = [0.001, 0.01, 0.1]

dropout\_rate = [0.0, 0.1, 0.2]

# make a dictionary of the grid search parameters

param\_grid = dict(learn\_rate=learn\_rate, dropout\_rate=dropout\_rate)

# build and fit the GridSearchCV

grid = GridSearchCV(estimator = model, param\_grid = param\_grid, cv = KFold(random\_state=seed), verbose = 10)

grid\_results = grid.fit(X\_standardized, Y)

# summarize the results

print("Best: {0}, using {1}".format(grid\_results.best\_score\_, grid\_results.best\_params\_))

means = grid\_results.cv\_results\_['mean\_test\_score']

stds = grid\_result.cv\_results\_['std\_test\_score']

params = grid\_results.cv\_results\_['params']

for mean, stdev, param in zip(means, stds, params):

print('{0} ({1}) with: {2}'.format(mean, stdev, param))

# Do a grid search to optimize kernel initialization and activation functions

# import necessary packages

# Define a random seed

seed = 6

np.random.seed(seed)

# Start defining the model

def create\_model(activation, init):

14

**3.2 IMPLEMENTATION**

# create model

model = Sequential()

model.add(Dense(8, input\_dim = 8, kernel\_initializer= init, activation= activation))

model.add(Dense(4, input\_dim = 8, kernel\_initializer= init, activation= activation))

model.add(Dense(1, activation='sigmoid'))

# compile the model

adam = Adam(lr = 0.001)

model.compile(loss = 'binary\_crossentropy', optimizer = adam, metrics = ['accuracy'])

return model

# create the model

model = KerasClassifier(build\_fn = create\_model, epochs = 100, batch\_size = 20, verbose = 0)

# define the grid search parameters

activation = ['softmax', 'relu', 'tanh', 'linear']

init = ['uniform', 'normal', 'zero']

# make a dictionary of the grid search parameters

param\_grid = dict(activation = activation, init = init)

# build and fit the GridSearchCV

grid = GridSearchCV(estimator = model, param\_grid = param\_grid, cv = KFold(random\_state=seed), verbose = 10)

grid\_results = grid.fit(X\_standardized, Y)

# summarize the results

print("Best: {0}, using {1}".format(grid\_results.best\_score\_, grid\_results.best\_params\_))

means = grid\_results.cv\_results\_['mean\_test\_score']

stds = grid\_result.cv\_results\_['std\_test\_score']

params = grid\_results.cv\_results\_['params']

for mean, stdev, param in zip(means, stds, params):

print('{0} ({1}) with: {2}'.format(mean, stdev, param))

# Do a grid search to find the optimal number of neurons in each hidden layer

# import necessary packages

# Define a random seed

seed = 6

np.random.seed(seed)

# Start defining the model

def create\_model(neuron1, neuron2):

15

**3.2 IMPLEMENTATION**

# create model

model = Sequential()

model.add(Dense(neuron1, input\_dim = 8, kernel\_initializer= 'uniform', activation= 'linear'))

model.add(Dense(neuron2, input\_dim = neuron1, kernel\_initializer= 'uniform', activation= 'linear'))

model.add(Dense(1, activation='sigmoid'))

# compile the model

adam = Adam(lr = 0.001)

model.compile(loss = 'binary\_crossentropy', optimizer = adam, metrics = ['accuracy'])

return model

# create the model

model = KerasClassifier(build\_fn = create\_model, epochs = 100, batch\_size = 20, verbose = 0)

# define the grid search parameters

neuron1 = [4, 8, 16]

neuron2 = [2, 4, 8]

# make a dictionary of the grid search parameters

param\_grid = dict(neuron1 = neuron1, neuron2 = neuron2)

# build and fit the GridSearchCV

grid = GridSearchCV(estimator = model, param\_grid = param\_grid, cv = KFold(random\_state=seed), refit = True, verbose = 10)

grid\_results = grid.fit(X\_standardized, Y)

# summarize the results

print("Best: {0}, using {1}".format(grid\_results.best\_score\_, grid\_results.best\_params\_))

means = grid\_results.cv\_results\_['mean\_test\_score']

stds = grid\_result.cv\_results\_['std\_test\_score']

params = grid\_results.cv\_results\_['params']

for mean, stdev, param in zip(means, stds, params):

print('{0} ({1}) with: {2}'.format(mean, stdev, param))

# generate predictions with optimal hyperparameters

y\_pred = grid.predict(X\_standardized)

print(y\_pred.shape)

print(y\_pred[:5])

# Generate a classification report

from sklearn.metrics import classification\_report, accuracy\_score

print(accuracy\_score(Y, y\_pred))

print(classification\_report(Y, y\_pred))

16

**3.3 Results**

We generated a classification report from sklearn.metrics import classification\_report, accuracy\_score

The average F1 score and precision is 0.77 and 0.77 for three-fold cross validation.

The performance of the system both for three-fold is represented in Table 3.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | Prediction | |  |  |
|  |  | Precision | recall | F1-score | Support |
|  |  |  |  |  |  |
| Actual | Class 1 | 0.81 | 0.89 | 0.84 | 262 |
| Class |  |  |  |  |  |
|  | Class 0 | 0.71 | 0.57 | 0.63 | 130 |
|  |  |  |  |  |  |
|  | Avg/total | 0.77 | 0.78 | 0.77 | 392 |
|  |  |  |  |  |  |

Table 3.3:Prediction Accuracy

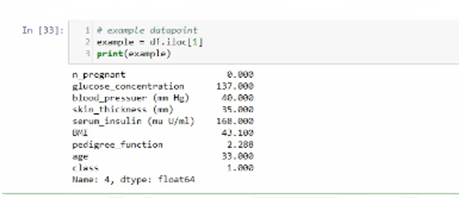




Fig 3.3:Prediction on example dataset

17

4

Conclusion

Diabetes is a chronic ailment that has to be prevented before it distresses people. Diabetes causes a lot of death per year throughout the world. So, the detection of diabetes in its initial stage is very important for the

treatment. This study has implemented deep neural network to predict diabetes. This investigation has engaged deep neural network technique to identity of diabetes based on several medical factors. The technique may also help researchers to develop an accurate and effective tool that will reach at the table of clinicians to help them make better decision about the disease status. We have taken default cross validation as 3.But 5 or maybe 10(any larger value) could be more stable .However that is longer to train. It is good to review whole grid at the same time. We only optimized 2 parameters at once. There can be relation between parameters. Although we set seed for random number generator in NumPy, results are not 100% reproducible. So different people can get slightly different results for same dataset.

.

18

Bibliography

1. <https://www.sciencedirect.com/science/article/pii/S221083271830365X>
2. <https://www.academia.edu/36963831/Diabetes_Prediction_Using_Machine_Learning_Techniques>
3. <https://www.ijert.org/research/a-survey-detection-and-prediction-of-diabetes-using-machine-learning-techniques-IJERTV7IS030167.pdf>
4. <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0175-6>

1. <http://www.ijirst.org/articles/IJIRSTV3I10027.pdf>
2. <https://www.kdnuggets.com/2018/05/general-approaches-machine-learning-process.html>
3. https://stackoverflow.com Komi, Zhai. 2017. Application of Data Mining Methods in Diabetes Prediction
4. Diagnosis and Classification of Diabetes Mellitus, American Diabetes Association, *Diabetes Care*, vol. 33, Jan. 2010.
5. Devi, M. Renuka, and J. Maria Shyla. "Analysis of Various Data Mining Techniques to Predict Diabetes Mellitus." *International* *Journal of Applied Engineering* *Research* 11.1, pp. 727-730, 2016.
6. Giri, Donna, et al. "Automated diagnosis of coronary artery disease affected patients using LDA, PCA, ICA and discrete wavelet transform." Knowledge-Based Systems 37 (2013): 274-282.

1. M. Heydari, M. Teimouri, Z. Heshmati, and S. M. Alavinia, "Comparison of various classification algorithms in the diagnosis of type diabetes in Iran," *International Journal of Diabetes in Developing Countries*, pp. 1-7, 2015.
2. A. Kumar Dwivedi, “Analysis of computational intelligence techniques for diabetes mellitus prediction,”.*Neural Comput. Appl*., vol. 13, no. 3, pp. 1–9, 2017.
3. M. Heydari, M. Teimouri, Z. Heshmati, and S. M. Alavinia, "Comparison of various classification algorithms in the diagnosis of type diabetes in Iran," *International Journal of Diabetes in Developing Countries*, pp. 1-7, 2015.

19

**BIBLIOGRAPHY**

1. A. Ashiquzzaman, A. K. Tushar, M. Islam, J.-M. Kim et al., ``Reduction of overfitting in diabetes prediction using deep learning neural network,'' *arXiv preprint* arXiv:1707.08386, 2017.
2. J. Zhu, Q. Xie, K. Zheng. “An Improved Early Detection

Method of Type-2 Diabetes Mellitus Using Multiple

Classifier Systems”. *Information Sciences*, volume 292, pages 1-14, 2015.

[16] M. Kumari, Dr. R. Vohra, and A. Arora, “Prediction of

Diabetes using Bayesian Network*,” International Journal* *of Computer Science and Information Technologies*, vol. 5,pp. 5174-5178, 2014.

[17] T. Santhanam and M.S Padmavathi, “Application of K-Means and Genetic Algorithms for Dimension Reduction by Integrating SNM for Diabetes Diagnosis,” *Procedia* *Computer Science*, vol. 47, pp. 76-83, 2015.

[18] J. Vijayashree and J. Jayashree, “ An Expert System for the Diagnosis of Diabetic Patients using Deep Neural

NetworksandRecursiveFeatureElimination,”

[19] <https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks>

[20] Pima Indian Diabetes Data Set, [Online]. Available: *https://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes.*

20