An Assignment Report

On

Building forecasting/predicting model for the given scenario.

by

Aditya Sharma
AU18B1009

Under the guidance of

Ashish Bansal

School of Engineering

Avantika University, Ujjain

2020-2021

Content: **Page**

Title : To Build forecasting/predicting model for the given scenario.

Statement :

To predict diabetes using the given dataset. The dataset represents the attributes of patients and whether they have diabetes or not. Looking at the parameters available in the dataset, train a neural network model to classify patients that might have diabetes than others. Use BPN for the purpose.

A. Identification of the Dataset

I. Type of the Dataset : Multivariate & Structured Dataset

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

The datasets consist of several medical predictor (independent) variables and one target (dependent) variable, **Outcome**. Independent variables include the number of **pregnancies** the patient has had, their **BMI**, **insulin** level, **age**, **BloodPressure, SkinThickness, Glucose, Diabetes pedigree function.**

II. Data Quality and Analysis :

Independent Variables:

Pregnancies -> The number of pregnancies the patient has or had

Glucose -> Plasma glucose concentration a 2 hours in an oral glucose tolerance test

Blood Pressure -> Diastolic blood pressure (mm Hg)

Skin Thickness -> Triceps skin fold thickness (mm)

Insulin -> 2-Hour serum insulin (mu U/ml)

BMI -> Body mass index (weight in kg/(height in m)^2)

Age -> Age (years)

Diabetes Pedigree Function -> Diabetes pedigree function. It provided some data on diabetes mellitus history in relatives and the genetic relationship of those relatives to the patient.

Dependent Variable:

Outcome -> Class variable (0 or 1) 268 of 768 are 1, the others are 0

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

Fig. 1 Diabetes Dataset

```
data.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

Fig. 2 Some basic statistical details

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

Fig. 3 Concise summary of the dataframe

```
In [2506]: data['Outcome'].value_counts()
```

```
Out[2506]: 0    500
           1    268
           Name: Outcome, dtype: int64
```

Visulization

```
In [2507]: ## count plot for target veriable
           sn.countplot(data['Outcome'], palette=['green', 'red'])
```

```
Out[2507]: <AxesSubplot:xlabel='Outcome', ylabel='count'>
```

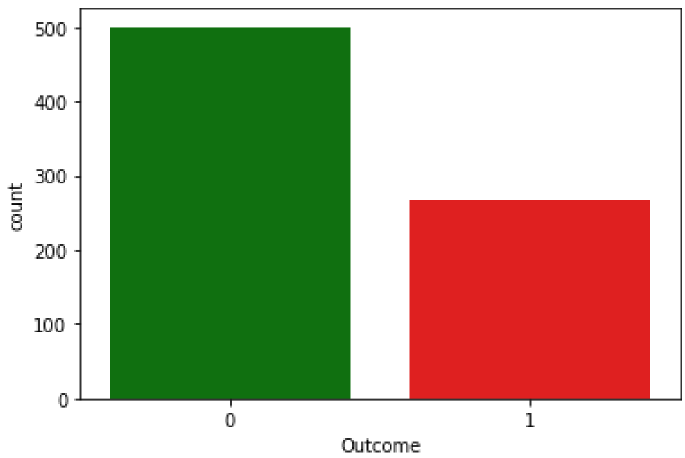

Fig. 4 Dependent Variable Visualization

```
plt.title('Ratio of Healthy and Diabetic Patients in Dataset')
plt.pie(data['Outcome'].value_counts(),autopct='%.2f')
plt.legend(['Healthy','Diabetic'],)
plt.show()
```

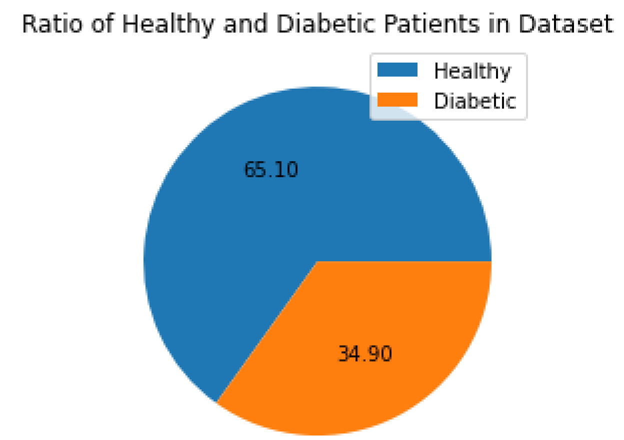

Fig. 5 Ration of healthy and Diabetic Patients

```
sn.relplot(x='Age', y='Insulin', data=data, hue='Outcome', kind='scatter')
plt.show()
```

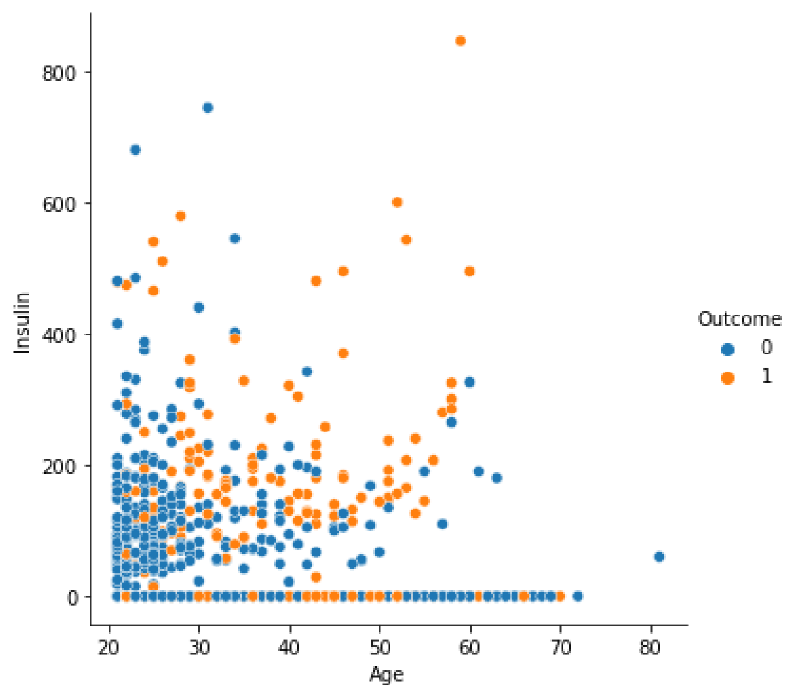

Fig. 6 Age and Insulin, how both of this are related to make a patient a diabetic

III. Features Pre-Processing :

```
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

Fig. 7 Checking for the null values

```
sn.heatmap(data.corr(), annot=True, linewidths=.5)

<AxesSubplot:>
```

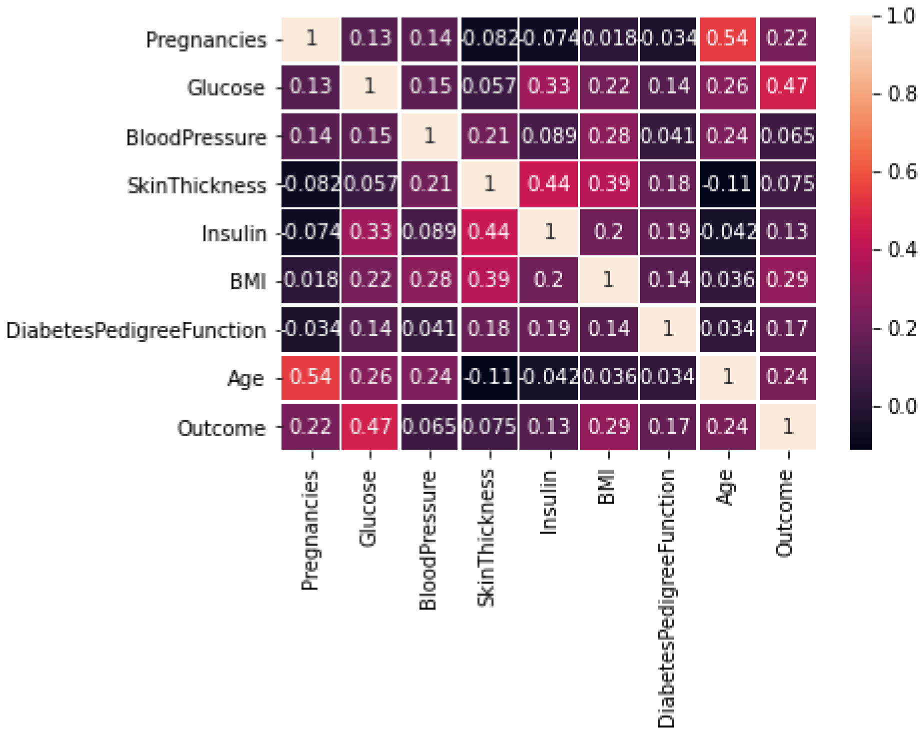

Fig. 8 Correlation of the independent variables.

Although there arent any null value associated with the dataset but Skin Thickness, Insulin, Blood Pressure, Glucose and BMI has some of the data encoded as 0s.

0's needs to be replaced.

Using the median value associated with that particular column, we will replace the 0's.

```
data.SkinThickness.replace(0, data.SkinThickness.median(), inplace=True)
data.Insulin.replace(0, data.Insulin.median(), inplace=True)
data.Glucose.replace(0, data.Glucose.median(), inplace=True)
data.BloodPressure.replace(0, data.BloodPressure.median(), inplace=True)
data.BMI.replace(0, data.BMI.median(), inplace=True)
```

Fig. 9 Replacing the 0’s with the median.

IV. Format of the Dataset : CSV

B. Neural Network

I. Model building, Training & Testing :

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers,callbacks
from tensorflow.keras.callbacks import EarlyStopping
from sklearn import preprocessing,model_selection
```

Fig. 10 Libraries used

```
x = data.drop('Outcome', axis=1)
y = data['Outcome']
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=2)
```

```
print(X_train.shape,y_train.shape,X_test.shape,y_test.shape)
```

```
(537, 8) (537,) (231, 8) (231,)
```

Fig.11 Split the data into a training set and test set.

Defining the model:

```
Early_Stopping=callbacks.EarlyStopping(min_delta=0.001,patience=20,restore_best_weights=True,)
```

```
model=keras.Sequential([
    layers.Dense(14,activation='relu',input_shape=[8]),
    layers.Dropout(0.3),
    layers.BatchNormalization(),
    layers.Dense(8,activation='relu'),
    layers.Dense(1),])
```

Fig.12 Model definition.

Model Compilation:

```
: from keras.optimizers import Adam
model.compile(loss='binary_crossentropy', optimizer='adam')
```

Fig.13 Compiling the defined model .

Fitting the model:

```
from keras.callbacks import ReduceLROnPlateau, EarlyStopping
reduce_lr = ReduceLROnPlateau()
early_stopping = EarlyStopping(patience=20, min_delta=0.0001)
```

```
history=model.fit(X_train,y_train,validation_data=(X_test,y_test),batch_size=256,epochs=200,callbacks=[Early_Stopping],verbose=0)
```

```
history_df=pd.DataFrame(history.history)
```

II. Model Accuracy, Prediction & Precession :

```
accuracy=model.evaluate(x,y)

24/24 [==============================] - 0s 2ms/step - loss: 0.9434

print(accuracy*100)

94.3443238735199
```

Fig. 15 Model Accuracy

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome	predictions
0	6	148	72	35	30.5	33.6	0.627	50	1	1
1	1	85	66	29	30.5	26.6	0.351	31	0	0
2	8	183	64	23	30.5	23.3	0.672	32	1	1
3	1	89	66	23	94.0	28.1	0.167	21	0	0
4	0	137	40	35	168.0	43.1	2.288	33	1	0
...	...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180.0	32.9	0.171	63	0	0
764	2	122	70	27	30.5	36.8	0.340	27	0	0
765	5	121	72	23	112.0	26.2	0.245	30	0	1
766	1	126	60	23	30.5	30.1	0.349	47	1	1
767	1	93	70	31	30.5	30.4	0.315	23	0	0

Fig. 16 Model Predictions

```
0    0
1    0
0    0
0    0
0    1
0    0
0    0
0    0
0    0
0    0
0    1
0    1
0    1
1    0
0    0
0    0
1    1
1    0
1    1
```

```
1    1
0    0
1    1
0    0
0    1
0    0
0    1
0    0
0    1
0    1
0    0
1    1
0    0
0    1
1    1
0    1
0    1
0    1
0    0
```

```
0    0
0    0
1    1
0    0
0    0
0    0
0    0
1    1
1    0
0    0
0    0
0    0
0    0
0    1
1    1
1    0
0    0
0    0
0    0
```

Fig. 16 Actual values vs. Predicted Value.

C. Key Learning Outcomes :

In this assignment I had learnt how to design a neural network, compiling the neural network, fitting the model and Evaluations. After completion of this assignment I am confident to build deep learning models and neural networks models using keras and tensorflow.