Heaven's Light is Our Guide

Computer Science & Engineering

Rajshahi University of Engineering & Technology

Course No.: CSE 4204

Course Title: Sessional based on CSE 4203

Experiment No. 4

Name of the Experiment:

a. Design and implementation of Kohonen Self-organizing Neural Networks algorithm.

b. Design and implementation of Hopfield Neural Networks algorithm.

Course Outcomes: CO1

Learning Domain with Level: Cognitive (Applying, Analyzing, Evaluating & Creating)

Contents

1 Dataset

I've Selected Diabetes Dataset from Kaggle [1] .This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective is to predict based on diagnostic measurements whether a patient has diabetes.

1.1 Dataset Analysis

The Dataset have 8 attributes and 768 instances. The Features are :

1. **Pregnancies:** Number of times pregnant
2. **Glucose:** Plasma glucose concentration a 2 hours in an oral glucose tolerance test
3. **BloodPressure:** Diastolic blood pressure (mm Hg)
4. **Insulin:** 2-Hour serum insulin (mu U/ml)
5. **BMI:** Body mass index (weight in $kg/(heightinm)^2$)
6. **DiabetesPedigreeFunction:** Diabetes pedigree function
7. **Age:** Age (years)
8. **Outcome:** Class variable (0 or 1)

Class Distribution: class value 1 is interpreted as "tested positive for diabetes". In Class 1 there were 268 instances and in class 0 , there were 500 instances.

1.2 Training-Test Ratio

80% of data were used for training and 20% of data for validating .

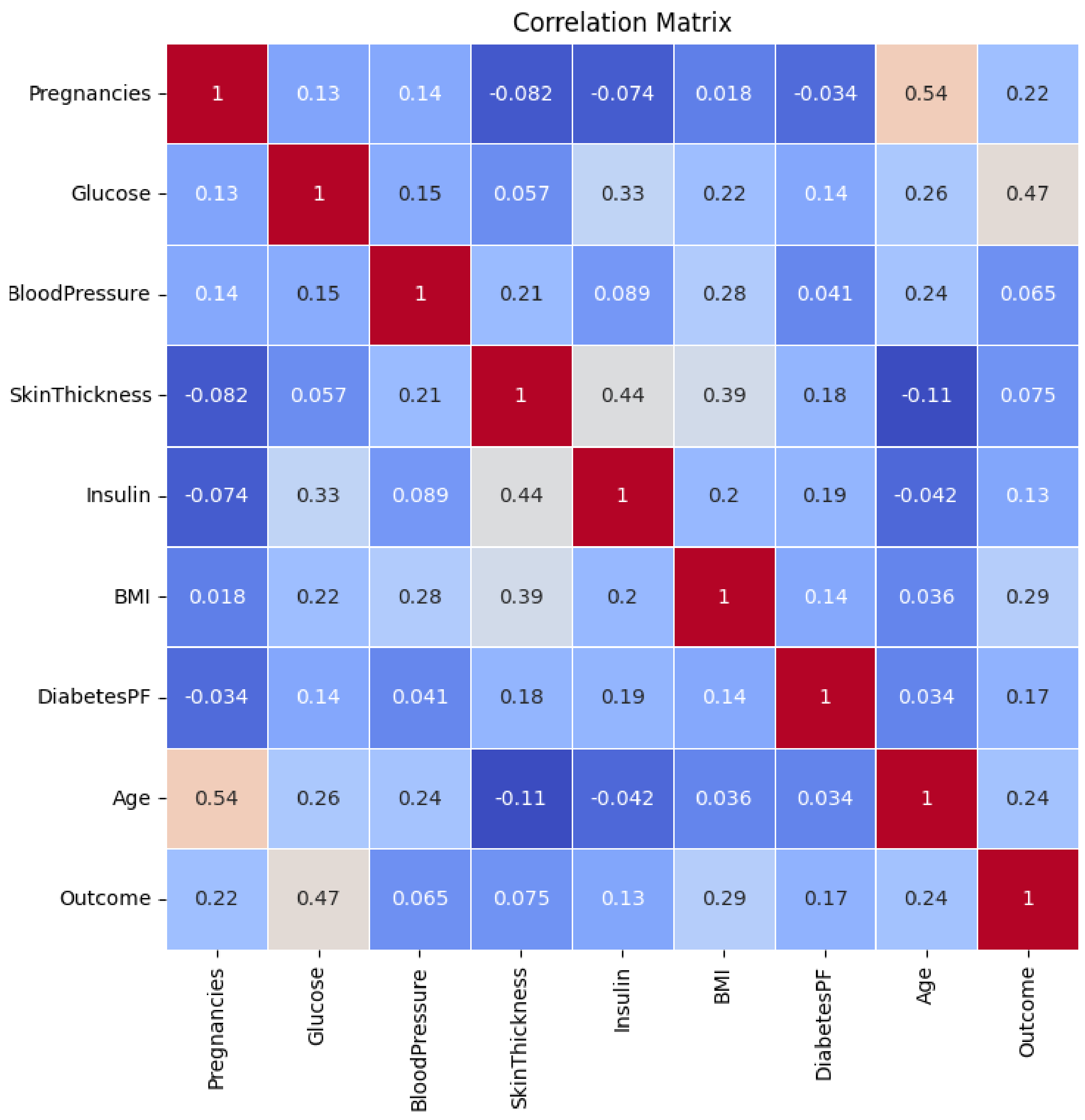
2 Kohonen Self-organizing Neural Network

2.1 Preprocessing

Kohonen Self-organizing Neural Networks algorithm was applied for only 2 selected features for viewing the clusters.

2.1.1 Correlation Matrix

For Feature selection a correlation matrix was printed for understating the correlation between different features . From the correlation matrix it is shown that the 'Glucose' attribute is highly correlated with the class variable. Then 'BMI' is more correlates with class variable than others . The correlation matrix is shown below:



	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPF	Age	Outcome
Pregnancies	1	0.13	0.14	-0.082	-0.074	0.018	-0.034	0.54	0.22
Glucose	0.13	1	0.15	0.057	0.33	0.22	0.14	0.26	0.47
BloodPressure	0.14	0.15	1	0.21	0.089	0.28	0.041	0.24	0.065
SkinThickness	-0.082	0.057	0.21	1	0.44	0.39	0.18	-0.11	0.075
Insulin	-0.074	0.33	0.089	0.44	1	0.2	0.19	-0.042	0.13
BMI	0.018	0.22	0.28	0.39	0.2	1	0.14	0.036	0.29
DiabetesPF	-0.034	0.14	0.041	0.18	0.19	0.14	1	0.034	0.17
Age	0.54	0.26	0.24	-0.11	-0.042	0.036	0.034	1	0.24
Outcome	0.22	0.47	0.065	0.075	0.13	0.29	0.17	0.24	1

Figure 1: Correlation Matrix

Based of Correlation Matrix It was shown that "Glucose " ,"BMI" Attributes are more correlated with output than others. After Plotting (Figure 2) it was clear that the Dataset was not fully linearly separable . There were overlapping and Noise in the Dataset.

2.1.2 Normalization

A z-score normalization was done by using this formula:

$$Z = \frac{X - \mu}{\sigma}$$

where:
Z is the is the standardized value,
X is the original value of the feature,
μ is the mean of the feature,

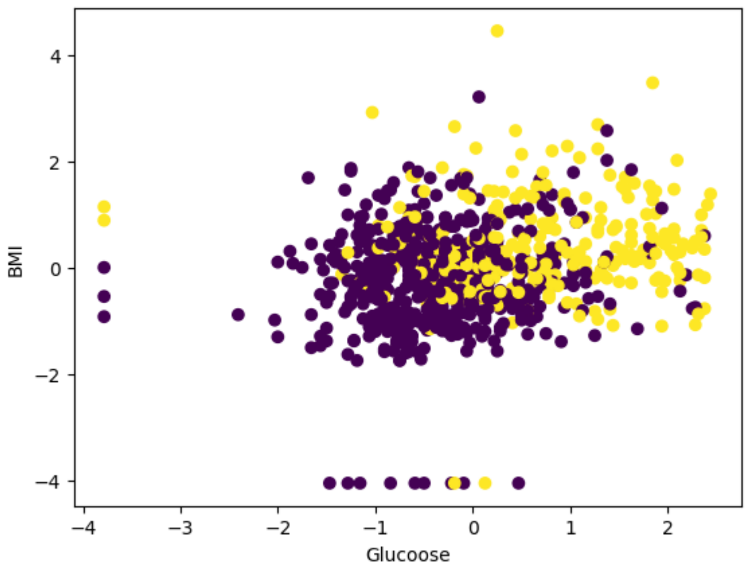


Figure 2: 2D scatter Plot of Glucose , BMI

σ is the standard deviation of the feature.

2.2 Kohonen Network Algorithm

The Algorithm were described as follows[2] :

1. Initialise network
   Define $w_{ij}(t)(0 \leq i \leq n-1)$ to be the weight from input i to node j at time t. Initialise weights from the n inputs to the nodes to small random values. Set the initial radius of the neighbourhood around node j, $N_j(0)$, to be large.

2. Present input Present input $X_p = x_0(t), x_1(t), x_2(t), \ldots, x_{n-1}(t)$ where $x_i(t)$ is the input node i at time t.

3. Calculate Distances
   Compute the distance d_j between the input and each output node j, given by

$$d_j = \sum_{i=0}^{n-1}(x_i(t) - w_{ij}(t))^2$$

4. Select minimum distance
   Designate the output node with minimum d_j to be $j*$.

5. Update weights
   Update weights for node $j*$ and its neighbours, defined by the neighbourhood size $N_{j*}(t)$. New weights are

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)(x_i(t) - wij(t)$$

For j in $N_{j*}(t)$, $0 \leq i \leq n-1$
The term $\eta(t)$ is a gain term ($0 < \eta(t) < 1$) that decreases in time, so slowing the weight adaption. Notice that the neighbourhood $N_{j*}(t)$ decreases in size as time goes on, thus localising the area of maximum activity.

6. Repeat by going to 2

2.3 Performance Analyzing

In selected dataset Kohonen Network Algorithm was applied, At First I applied it for 2 selected feature based on correlation matrix for visualization . Which is shown in Figure 3. The performance was measured by Quantization Error. The quantization error is the average distance between each input vector and its corresponding Best Matching Unit (BMU) on the Self Organizing Map. Which can be expressed as :

$$QE = \frac{1}{N}\sum_{i=1}^{N} d(input_i, BMU_i)$$

The Quantization error is then normalized for represent the average normalized distance between each input vector and its corresponding BMU.

$$QE_{\text{normalized}} = \frac{1}{N}\sum_{i=1}^{N} \frac{d(\text{input}_i, \text{BMU}_i)}{E_{\text{max}}}$$

The Quantization error was 0.96 and the Normalized Quantization error was 0.66 . Also

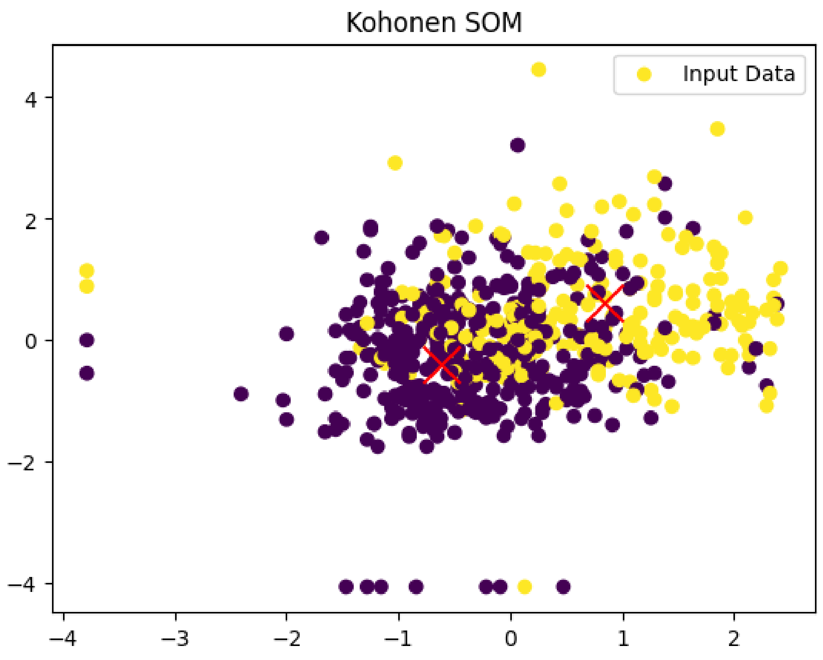


Figure 3: Output of KSOM.

when all 8 features of the data was passed in the algorithm the quantization error was 1.85 and the normalized quantization error was 0.53 .

2.4 Conclusion

In summary, the Kohonen Self-Organizing Algorithm and the Kaggle Diabetes data brought some interesting discoveries. When two features and a small map was selected, the algorithm excelled with a low error of 0.96. Yet, when in all eight features with a bigger map, the error climbed to 1.85, pointing to more complexity.

3 Hopfield Neural Network

3.1 Preprocessing

3.1.1 Normalizing

A Normalizing operation was applied to convert the data between 0 to 1 by following formula:

$$\text{normalizedData} = \frac{data - \text{minVal}}{\text{maxVal} - \text{minVal}}$$

Then value below than 0.05 is converted to -1 otherwise +1 .

3.2 Hopfield Network Algorithm

The Algorithm were described as follows[2] :

1. Assign connection weights

   $$w_{ij} = \begin{cases} \sum_{s=0}^{M-1} x_i^s x_j^s & i \neq j \\ 0 & i = j, 0 \leq i, j \leq M-1 \end{cases}$$

   where w_{ij} is the connection weight between node i and node j, and x_i^s is element i of the exemplar pattern for class s, and is either +1 or -1. There are M patterns, from 0 to M - 1, in total. The thresholds of the units are zero.

2. Initialise with unknown pattern

   $$\mu_i(0) = x_i \qquad 0 \leq i \leq N-1$$

   where $\mu_i(t)$ is the output of node i at time t.

3. Iterate until convergence

   $$\mu_i(t+1) = f_h \left[\sum_{i=0}^{N-1} w_{ij}\mu_j(t) \right] \qquad 0 \leq j \leq N-1$$

   The function f_h is the hard-limiting non-linearity, the step function, Repeat the iteration until the outputs from the nodes remain unchanged.

3.3 Performance Analyzing

After initializing the weight matrix. New pattern was applied for testing. The accuracy was calculated by :

$$\text{accuracy} = \frac{\text{total number of converged}}{\text{total value supplied}}$$

This give a accuracy value of 1. That means the algorithm successfully converged for all patterns .

3.4 Conclusion

In conclusion, the Hopfield Network Algorithm was applied to the dataset, and the data was initially normalized to a range between 0 and 1. Subsequently, the patterns were transformed into -1s and +1s based on a threshold value of 0.05. When the network was tested, a perfect accuracy score of 1 was achieved according to the chosen accuracy measure, which considered the ratio of converged patterns to the total number of patterns.

References

[1] Diabetes Dataset, [Online]. Available at: `https://www.kaggle.com/datasets/mathchi/diabetes-data-set`

[2] R. Beale and T. Jackson, *Neural Computing: An Introduction*, CRC Press, 1990.