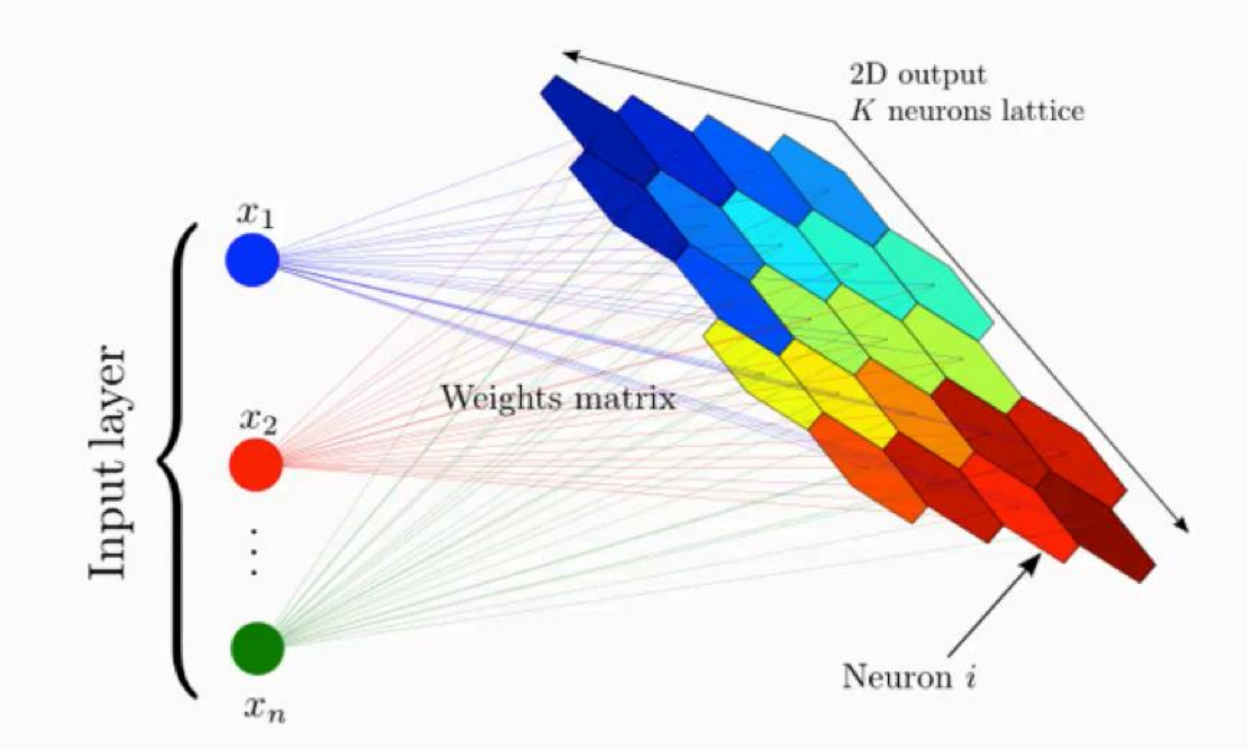
Visualizing Data using SOMs

Module	CS5613 - Neural Networks
Date	12/09/2020
Index	209337M
Name	Jayasuriya D.P.

About Self Organizing Maps - SOMs

Self-organizing maps is a type of artificial neural network that is trained using unsupervised methods in order to visualize high dimensional data in a map. Basically, this allows you for a better visualization, reducing the dimensions of the data. Not like traditional neural networks this used competitive learning instead of error correction methods to preserve the topological features of input space.

This is achieved by projecting multivariate data into a 2-dimensional map in a such a way that data items close to each other in high-dimensional data space are close to each other on the map. This can be utilized to visualize clusters in the data set to support the user in understanding the inherent structure of the data.



Question 1 - Literature Review

Self-Organizing Maps with vector fields

A visualization technique for Self-Organizing Maps with vector fields to obtain the cluster structure at desired levels of detail

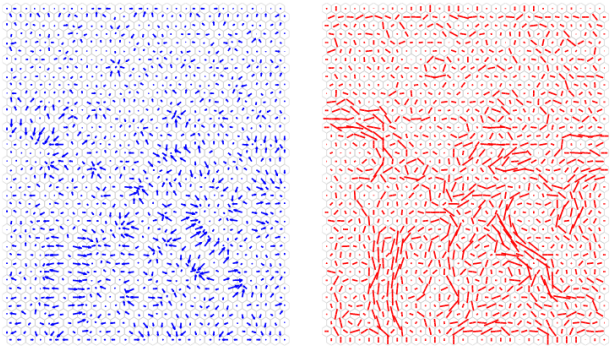
Link: http://www.ifs.tuwien.ac.at/~poelzlbauer/publications/Poe05IJCNN.pdf

This paper proposed a method which used SOM to visualize the cluster structure based on the similarity area of the underlying component planes of codebook. The result can be then plotted on the top of the map lattice with the arrows point to the closest cluster center, this is analogous to flow and vector field visualizations.

This method is based on the neighborhood kernel function and on aggregation of distances in the proximity of each codebook vector. This kernel function determines the influence of the prototype vectors among each other. The kernel depends on is the neighborhood radius σ which controls the width of the kernel function. The kernel function was used as a weighting factor that allows to compute the similarity in terms of input space distance of map units that are close to each other on the map.

Here they have mentioned two ways of visualizing it.

- Gradient field
  Arrows points towards the closest cluster center.
- Border visualization
  How grave the transition is between neighboring regions?



(a) (b)

(a)– Vector Field representation (b) – Border representation

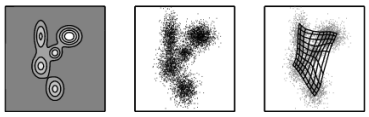
Smoothed Data Histograms in SOMs

Paper: Using Smoothed Data Histograms for Cluster Visualization in Self-Organizing Maps

Link: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.18.2118&rep=rep1&type=pdf

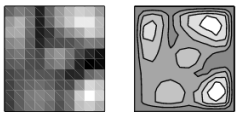
In this paper smoothed data histograms were used to visualize the clusters. The idea behind to use this method is that clusters are areas in the data space with a high density of data items. This uses SOM as the basis for a smoothed data histogram. The map units are interpreted as bins. The bin centers in the data space are defined by the model vectors and the varying bin widths are defined through the distances between the model vectors

The membership degree of a data item to a specific bin is governed by the smoothing parameters and calculated based on the rank of the distances between the data item and all bin centers. In particular, the membership degree is s/cs to the closest bin, (s−1)/cs to the second, (s−2)/csto the third, and so forth. The membership to all but the closest bins is 0. The constants=∑s−1i=0s−iensures that the total membership of each data item adds up to 1.



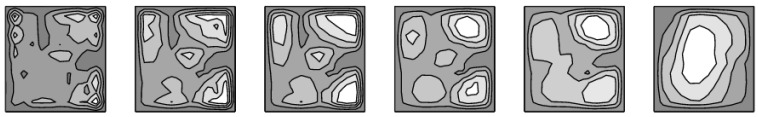
(a) (b) (c)

Fig. 1. The 2-dimensional data space. (a) The probability distribution from which (b) the sample was drawn and (c) the model vectors of the SOM.



(a) (b)

Fig. 2. The SDH ($s = 8$) visualized using (a) gray shadings and (b) contours.



(a) $s = 1$ (b) $s = 3$ (c) $s = 5$ (d) $s = 15$ (e) $s = 20$ (f) $s = 50$

Fig. 3. Different values for the smoothing parameters s and their effects on the SDH.

Visualizing SOMs using U-Matrix

Paper: Exploiting the structures of the U-matrix

Link: https://www.researchgate.net/publication/279446061_Exploiting_the_Structures_of_the_U-Matrix

This method is widely used to represent SOMs. The sum of distances between n and the neurons in N(n) in the high-dimensional space is shown on a U-matrix as a height value (U-height) at neuron n. Large U-heights mean that there is a large gap in the data space. The method is shown to detect and visualize meaningful cluster structures on difficult artificial and real-life data. This has become a standard visualization of self-organizing feature maps (SOM).

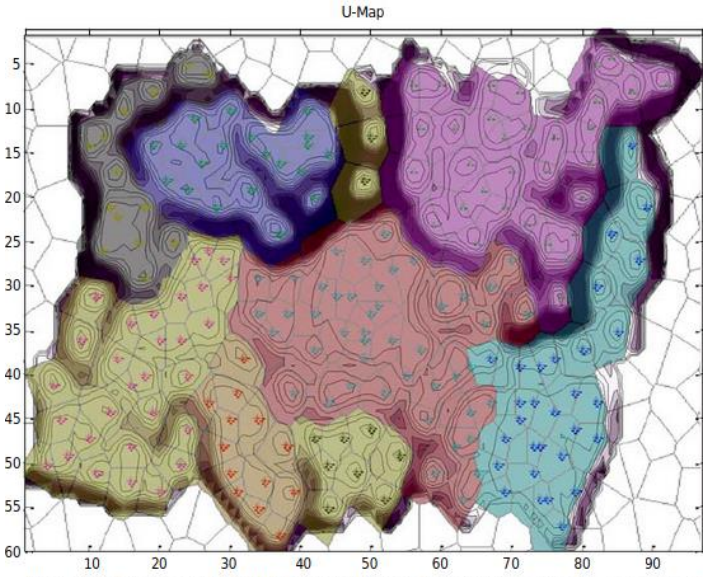


Figure 5: Political map of the Pain data set following clustering into eight classes of subjects with similar pain sensitivity patterns and overlaid onto the U-matrix (Figure 4)

The political map of a U-matrix is a very flexible tool to visualize the result of possible clustering. It allows to easily identify outliers and critical distance structures where the membership of data points to the same or different clusters is debatable.

Question2 – Implement one type of SOM

For this implementation U-Matrix visualization is used.

Implementation details are as below

Import dataset

In [232]:
```
file_name = 'iris.txt'
dataset = pd.read_csv(file_name, header=None)
X = dataset.iloc[:, :-1]
y = dataset.iloc[:, -1]
dataset.head()
```

Out[232]:

	0	1	2	3	4
0	5.1	3.5	1.4	0.2	1
1	4.9	3.0	1.4	0.2	1
2	4.6	3.1	1.5	0.2	1
3	5.0	3.6	1.4	0.2	1
4	4.6	3.4	1.4	0.3	1

Feature Scaling

In [233]:
```
sc = MinMaxScaler(feature_range=(0,1))
X = sc.fit_transform(X)
```

SOM Training

```
som_map = np.random.uniform(size=(rows,cols,no_of_features))
prev_som_map = np.zeros((rows,cols,no_of_features))
convergence = []
pre_cost = 100
for _iter in range(no_of_iterations):
    current_learning_rate = get_current_lr(_iter)
    current_radius = get_current_radius(_iter)
#     print("Current LR: {}".format(current_learning_rate))
#     print("Current Radius: {}".format(current_radius))
    J = np.linalg.norm(som_map - prev_som_map)
    randomly_selected_data = X[np.random.randint(len(X))]
    bmu_index = get_closest_node(randomly_selected_data, som_map)
    prev_som_map = np.copy(som_map)

    for col in range(cols):
        for row in range(rows):
#             print(row, col)
            som_distance = euclidian_distance(bmu_index, np.array([row, col]))
            if som_distance <= current_radius:
                som_map[row][col] = som_map[row][col] + current_learning_rate * ( randomly_selected_data - som_map[row][co
l])
    if pre_cost > J:
        print("New optimum point found, iter {} cost: {}".format(_iter, J))
        pre_cost = J
    convergence.append(J)
```

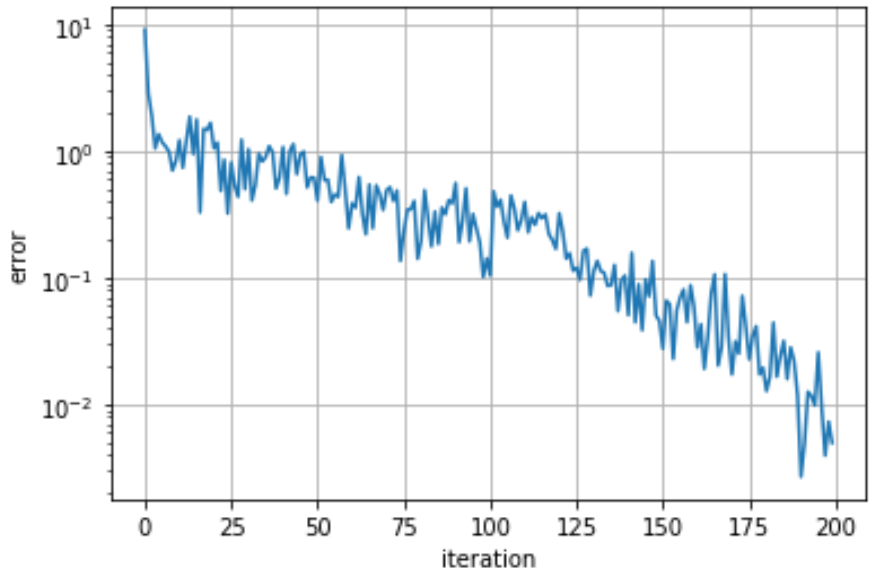
Plotting

```
# from scipy.misc import toimage
from PIL import Image


BMU = np.zeros([2],dtype=np.int32)
result_map = np.zeros([rows,cols,3],dtype=np.float32)

i=0
for i, pattern in enumerate(X):
    BMU = get_closest_node(pattern, som_map)

    row = BMU[0]
    col = BMU[1]

    if y[i] == 1:
        if result_map[row][col][0] <= 0.5:
            result_map[row][col] += np.asarray([0.5,0,0])
    elif y[i] == 2:
        if result_map[row][col][1] <= 0.5:
            result_map[row][col] += np.asarray([0,0.5,0])
    elif y[i] == 3:
        if result_map[row][col][2] <= 0.5:
            result_map[row][col] += np.asarray([0,0,0.5])
    i+=1
result_map = np.flip(result_map,0)

print("Red = Class 1")
print("Blue = Class 2")
print("Green = Class 3")

plt.figure(figsize = (10,10))
plt.imshow(result_map, interpolation='nearest')
```

Please Find the full code here: https://github.com/DulanGit/Visualizing-Data-using-Self-Origanizing-Maps/blob/master/Visualizing%20Data%20using%20SOMs.ipynb

Error Reduction

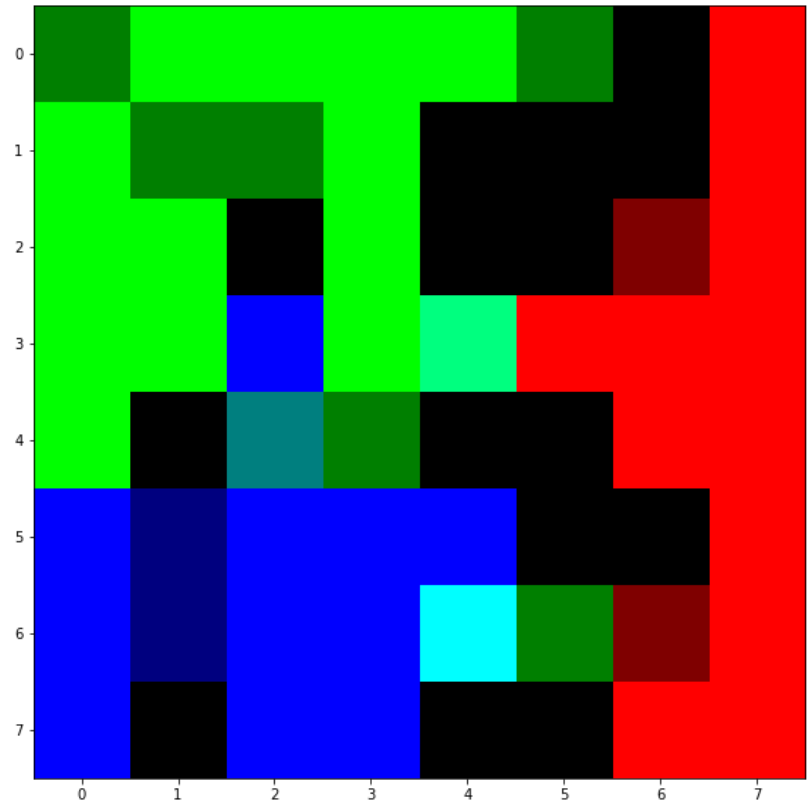
With the iterations error has been significantly reduced.



SOM Classification Results

Here I have used 5x5 matrix to visualize the high dimensional data. It's clear that it was able to correctly classify the data according to the classes.

```
Red = Class 1
Blue = Class 2
Green = Class 3
<matplotlib.image.AxesImage at 0x7fdc0841f0d0>
```



References

https://www.researchgate.net/publication/279446061_Exploiting_the_Structures_of_the_U-Matrix

https://archive.ics.uci.edu/ml/datasets/iris

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.18.2118&rep=rep1&type=pdf

http://www.ifs.tuwien.ac.at/~poelzlbauer/publications/Poe05IJCNN.pdf

https://medium.com/machine-learning-researcher/self-organizing-map-som-c296561e2117

https://medium.com/@abhinavr8/self-organizing-maps-ff5853a118d4