- Name : Rishabh Patil
- SAP : 60009200056/K2

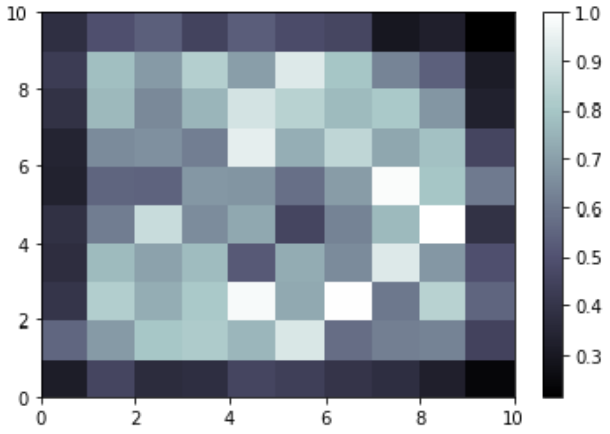
```
!pip install minisom
```

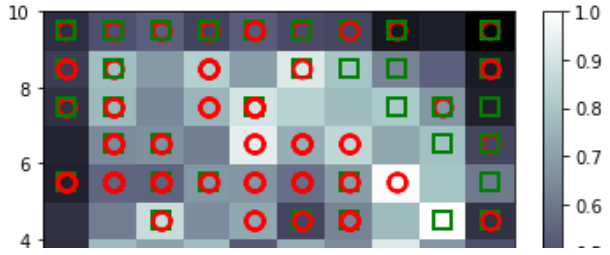
```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/sim
Collecting minisom
  Downloading MiniSom-2.3.0.tar.gz (8.8 kB)
Building wheels for collected packages: minisom
  Building wheel for minisom (setup.py) ... done
  Created wheel for minisom: filename=MiniSom-2.3.0-py3-none-any.whl size=9018 sha256=efb7f8d6
  Stored in directory: /root/.cache/pip/wheels/d4/ca/4a/488772b0399fec45ff53132ed14c948dec4b30
Successfully built minisom
Installing collected packages: minisom
Successfully installed minisom-2.3.0
```

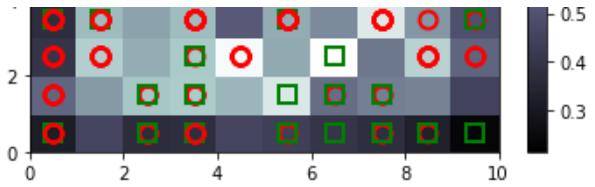
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# Minisom library and module is used for performing Self Organizing Maps
from minisom import MiniSom
# Loading Data
data = pd.read_csv('Credit_Card_Applications.csv')
# X
data
# Shape of the data:
data.shape
# Info of the data:
data.info()
# Defining X variables for the input of SOM
X = data.iloc[:, 1:14].values
y = data.iloc[:, -1].values
# X variables:
pd.DataFrame(X)
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler(feature_range = (0, 1))
X = sc.fit_transform(X)
pd.DataFrame(X)
# Set the hyper parameters
som_grid_rows = 10
som_grid_columns = 10
iterations = 20000
sigma = 1
learning_rate = 0.5
# define SOM:
som = MiniSom(x = som_grid_rows, y = som_grid_columns, input_len=13, sigma=sigma, learning_rate=lear
# Initializing the weights
som.random_weights_init(X)
# Training
som.train_random(X, iterations)
# Weights are:
#wts = som.weights
# Shape of the weight are:
#wts.shape
# Returns the distance map from the weights:
som.distance_map()
from pylab import plot, axis, show, pcolor, colorbar, bone
bone()
```

```
colorbar()
show()
bone()
pcolor(som.distance_map().T)
colorbar() #gives legend
markers = ['o', 's']                # if the observation is fraud then red circular color or else g
colors = ['r', 'g']
for i, x in enumerate(X):
    w = som.winner(x)
    plot(w[0] + 0.5,
         w[1] + 0.5,
         markers[y[i]],
         markeredgecolor = colors[y[i]],
         markerfacecolor = 'None',
         markersize = 10,
         markeredgewidth = 2)
show()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 690 entries, 0 to 689
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   CustomerID  690 non-null    int64
 1   A1          690 non-null    int64
 2   A2          690 non-null    float64
 3   A3          690 non-null    float64
 4   A4          690 non-null    int64
 5   A5          690 non-null    int64
 6   A6          690 non-null    int64
 7   A7          690 non-null    float64
 8   A8          690 non-null    int64
 9   A9          690 non-null    int64
 10  A10         690 non-null    int64
 11  A11         690 non-null    int64
 12  A12         690 non-null    int64
 13  A13         690 non-null    int64
 14  A14         690 non-null    int64
 15  Class       690 non-null    int64
dtypes: float64(3), int64(13)
memory usage: 86.4 KB
```






```
mappings = som.win_map(X)
mappings
mappings.keys()
len(mappings.keys())
mappings[(9,8)]
frauds = np.concatenate((mappings[(4,7)], mappings[(5,8)]), axis = 0)
frauds
# the list of customers who are frauds:
frauds1 = sc.inverse_transform(frauds)
pd.DataFrame(frauds1)
```

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.0 | 19.50 | 0.165 | 2.0 | 11.0 | 4.0 | 0.040 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 380.0 |
| 1 | 0.0 | 29.75 | 0.665 | 2.0 | 9.0 | 4.0 | 0.250 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 300.0 |
| 2 | 0.0 | 21.75 | 11.750 | 2.0 | 8.0 | 4.0 | 0.250 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 180.0 |
| 3 | 0.0 | 50.25 | 0.835 | 2.0 | 6.0 | 4.0 | 0.500 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 240.0 |
| 4 | 0.0 | 26.17 | 2.000 | 2.0 | 5.0 | 3.0 | 0.000 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 276.0 |
| 5 | 0.0 | 22.92 | 1.250 | 2.0 | 11.0 | 4.0 | 0.250 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 120.0 |
| 6 | 0.0 | 24.83 | 4.500 | 2.0 | 9.0 | 4.0 | 1.000 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 360.0 |
| 7 | 0.0 | 18.08 | 0.375 | 3.0 | 13.0 | 1.0 | 10.000 | 0.0 | 0.0 | 0.0 | 1.0 | 1.0 | 300.0 |
| 8 | 0.0 | 45.33 | 1.000 | 2.0 | 11.0 | 4.0 | 0.125 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 263.0 |
| 9 | 0.0 | 23.50 | 1.500 | 2.0 | 9.0 | 4.0 | 0.875 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 160.0 |
| 10 | 0.0 | 27.67 | 2.040 | 2.0 | 9.0 | 4.0 | 0.250 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 180.0 |
| 11 | 0.0 | 32.25 | 1.500 | 2.0 | 8.0 | 4.0 | 0.250 | 0.0 | 0.0 | 0.0 | 1.0 | 2.0 | 372.0 |
| 12 | 0.0 | 24.50 | 0.500 | 2.0 | 11.0 | 8.0 | 1.500 | 1.0 | 0.0 | 0.0 | 0.0 | 2.0 | 280.0 |
| 13 | 0.0 | 28.08 | 15.000 | 1.0 | 10.0 | 9.0 | 0.000 | 1.0 | 0.0 | 0.0 | 0.0 | 2.0 | 0.0 |
| 14 | 0.0 | 40.83 | 10.000 | 2.0 | 11.0 | 8.0 | 1.750 | 1.0 | 0.0 | 0.0 | 0.0 | 2.0 | 29.0 |
| 15 | 0.0 | 18.83 | 4.415 | 1.0 | 8.0 | 8.0 | 3.000 | 1.0 | 0.0 | 0.0 | 0.0 | 2.0 | 240.0 |
| 16 | 0.0 | 25.17 | 2.875 | 2.0 | 14.0 | 8.0 | 0.875 | 1.0 | 0.0 | 0.0 | 0.0 | 2.0 | 360.0 |
| 17 | 0.0 | 20.50 | 11.835 | 2.0 | 8.0 | 8.0 | 6.000 | 1.0 | 0.0 | 0.0 | 0.0 | 2.0 | 340.0 |
| 18 | 0.0 | 24.58 | 0.670 | 2.0 | 6.0 | 8.0 | 1.750 | 1.0 | 0.0 | 0.0 | 0.0 | 2.0 | 400.0 |