# Handwritten Digits Clustering Using SOM

In this code snippet, we are going to use the light weight Python library, `SimpSOM` (https://github.com/fcomitani/SimpSOM), for Kohonen self-organizing map to cluster the handwritten digit images provided by `scikit-learn` (https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_digits.html) package.

In [1]:
```
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split
from SimpSOM import somNet
import numpy as np
from utils import plot_2D_samples
```

### Step 1. Load Data

The handwritten image dataset in the scikit-learn package contains 1797 samples of 10 digits (around 180 samples per class). We use `load_digits` (https://scikitlearn.org/stable/modules/generated/sklearn.datasets.load_digits.html) function to load the dataset. Then, the `train_test_split` shuffles the dataset and splits it randomly into two sets of train and test.

If `test_size` is less than 1, it is considered as the test set percentage; otherwise, it shows the exact desired number of test samples.

In [2]:
```
X, y = load_digits(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=100)
```

### Step 2. Build, Train & Save the Model

This step contains building, training and saving the model as follows:

- Build a network 20x20 with a weights format taken from the train set.
- Train the network for 10000 epochs and with initial learning rate of 0.01.
- Save the weights to file

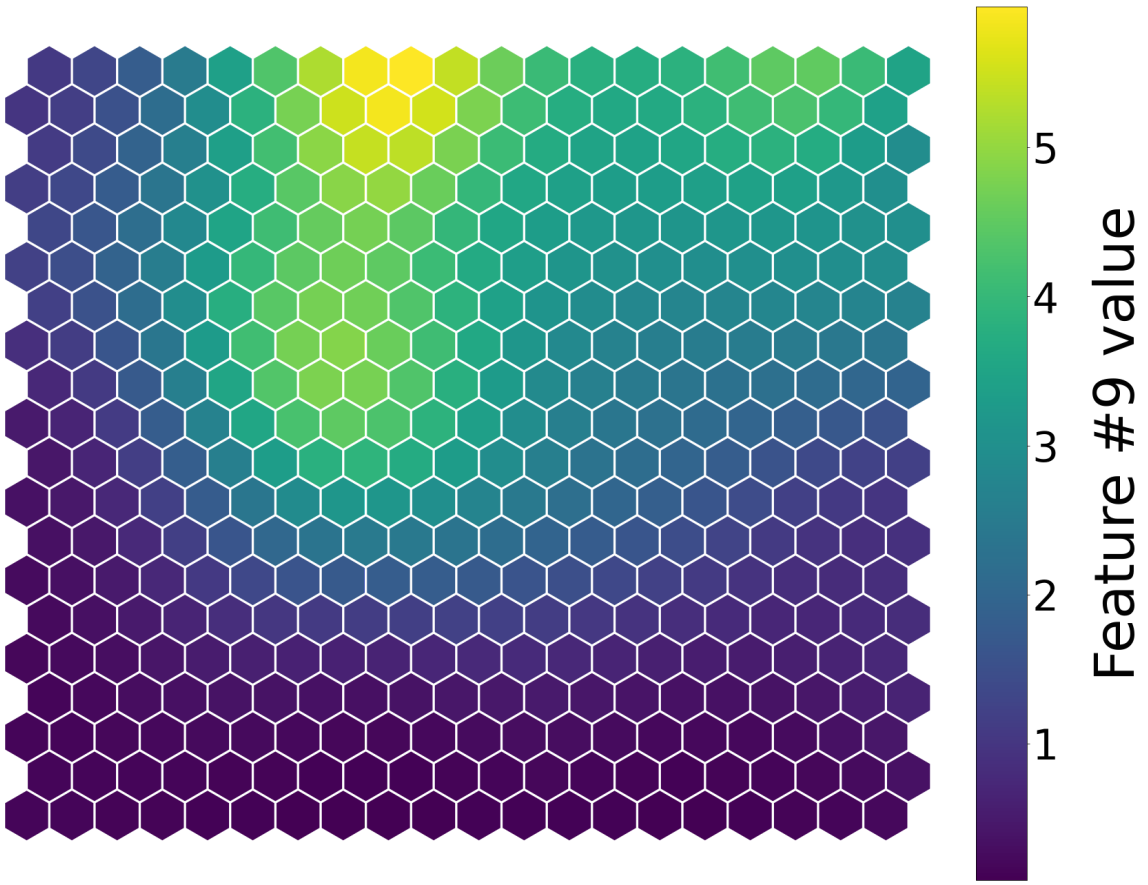
In [3]:
```
net = somNet(20, 20, X_train)
net.train(0.01, 10000)
net.save('filename_weights')
```

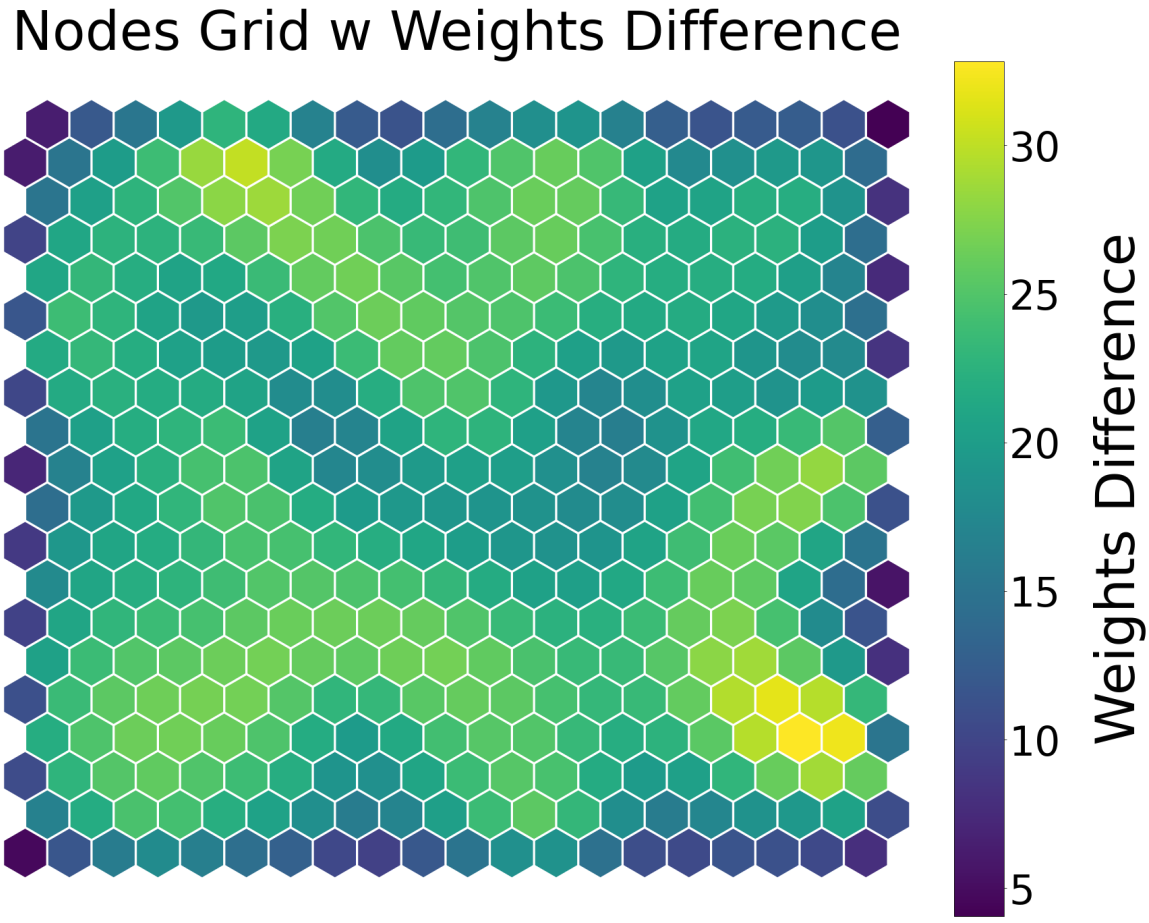
```
Periodic Boundary Conditions inactive.
The weights will be initialised randomly.
Training SOM... done!
```

### *Node Graph Plots*

We print a map of the network nodes and color them according to the tenth feature (column number 9) of the dataset, and then according to the distance between each node and its neighbours. The second graph is called **heatmap**.

In [4]:
```
net.nodes_graph(colnum=9)
net.diff_graph()
```

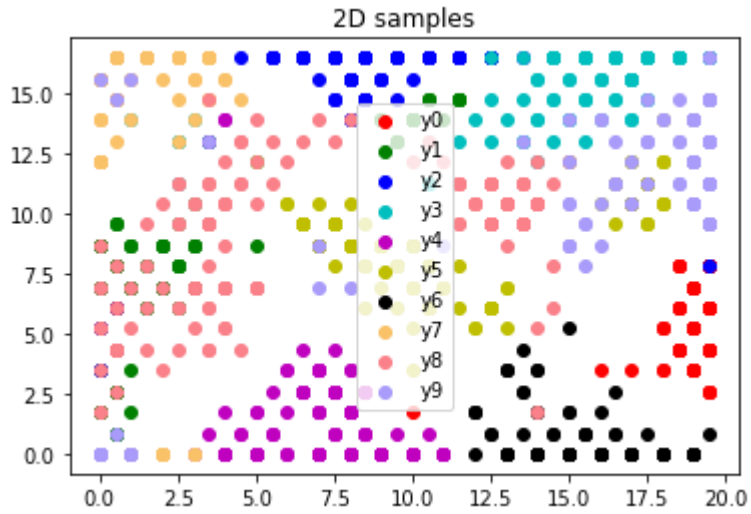

Node Grid w Feature #9
5
4
3
2
1
Feature #9 value


Nodes Grid w Weights Difference
30
25
20
15
10
5
Weights Difference

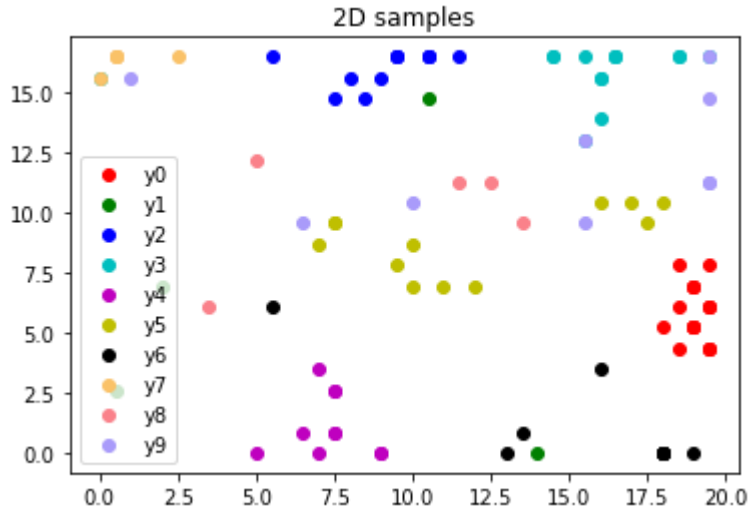
### *Plot Clusters*

To project the datapoints on the new 2D network map, we utilize `project` , the `SimpSOM` built-n function to find out the corresponding activated node; then then we used our `plot_2D_samples` function to visualize it. We can repeat for both train and test sets

In [7]:
```
prj = np.array(net.project(X_train))
plot_2D_samples(prj, y_train)
```




In [8]:
```
prj = np.array(net.project(X_test))
plot_2D_samples(prj, y_test)
```




In [ ]:
```

```