Neural Gas demo

This notebook contains a short example about the usage of the NeuGas class.

You can download the code from Taracchi repository.

In [ ]:
```
from sklearn import datasets
from neugas import NeuGas, quantization_error
import numpy as np
import time

from bokeh.plotting import figure, show
from bokeh.io import output_notebook
from bokeh.palettes import Vibrant3

output_notebook()
np.random.seed(1976)
```

BokehJS 3.3.2 successfully loaded.

Parameters

Some parameters about the tests and the adopted Neural Gas:

- number of points in the dataset
- number of particles of the NeuGas
- uprate value

In [ ]:
```
NUM_POINTS=5000
NG_PARTICLES=NUM_POINTS//50
UPRATE_VALUE=0.5
```

Datasets creation

In [ ]:
```
datasetbag=[]
datasetbag.append(datasets.make_moons(n_samples=NUM_POINTS,noise=0.05)[0])
datasetbag.append(datasets.make_blobs(n_samples=NUM_POINTS, n_features=2,cluster_std=1.0)[0])
datasetbag.append(datasets.make_circles(n_samples=NUM_POINTS, factor=0.5, noise=0.05)[0])
```

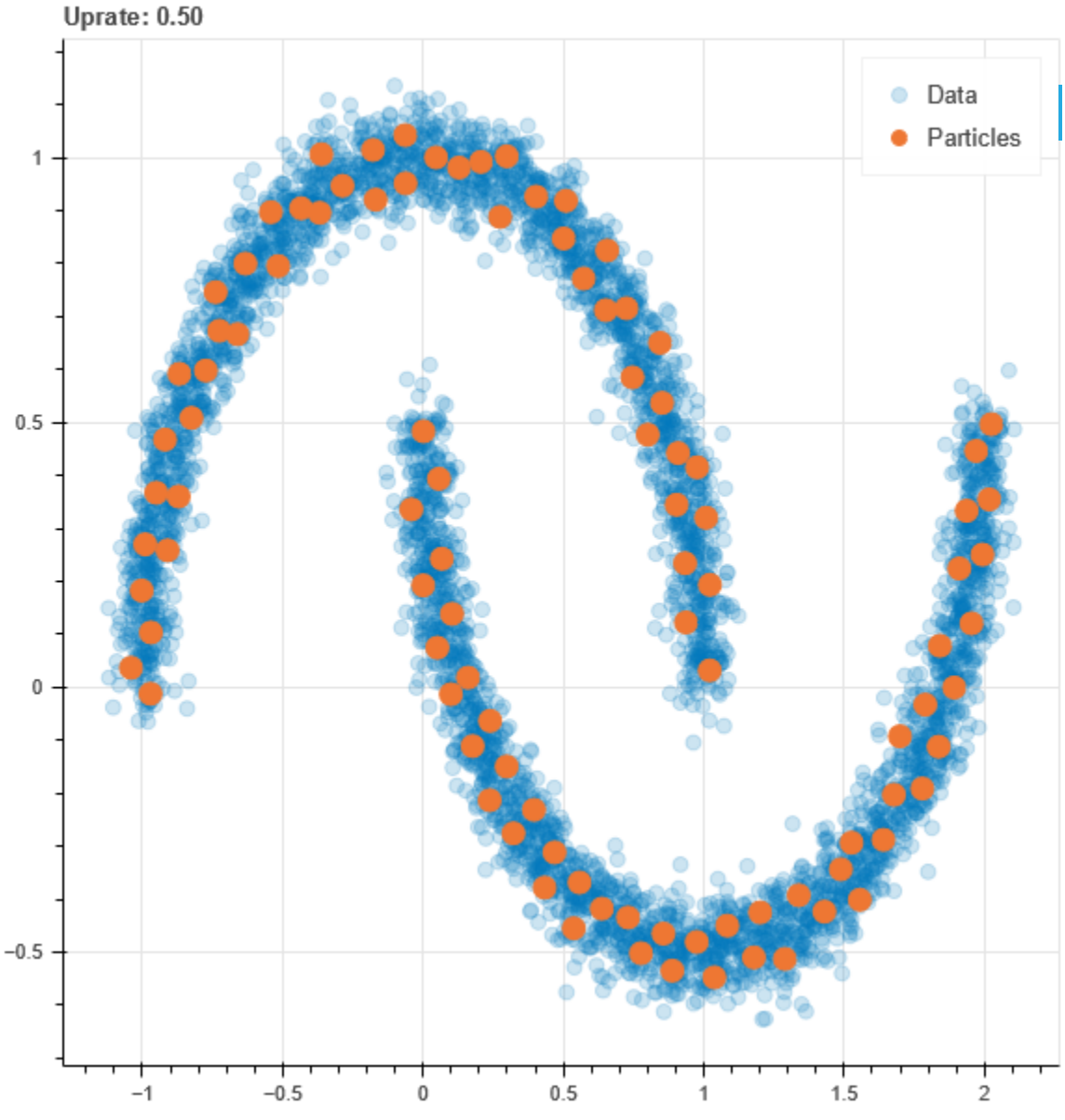
NeuGas training

... and visualization of the results on the three datasets

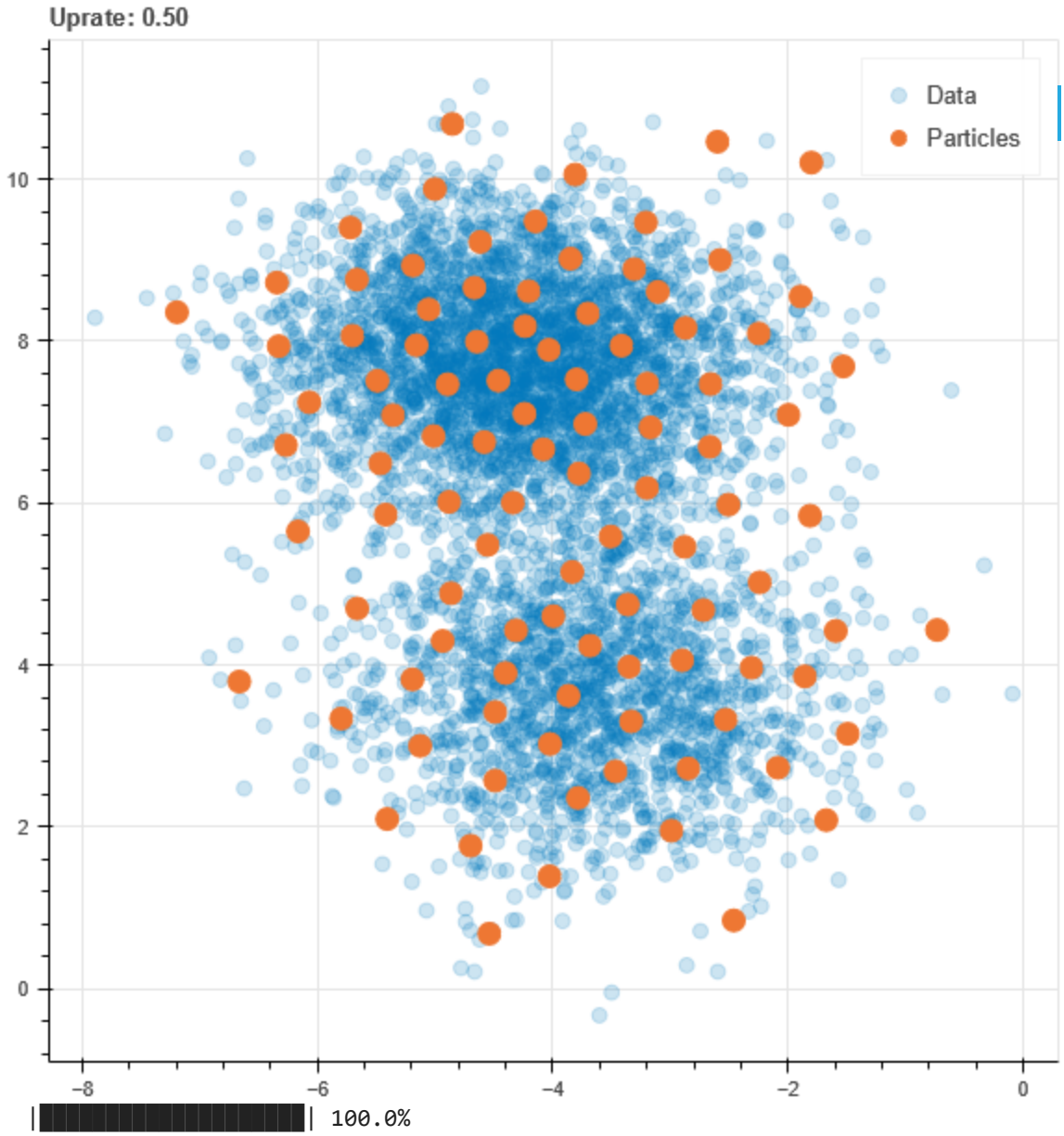
In [ ]:
```
for i,data in enumerate(datasetbag):

    ng = NeuGas(data, NG_PARTICLES)

    start_time=time.time()
    ng.fit(data,
           iterations=2*NUM_POINTS,
```

```
            uprate=UPRATE_VALUE,
            delta_coeff_min_update=0,
            lr0=1)
    elapsed_time=time.time()-start_time
    _,individual_qe=ng.quantize(data)
    print('Dataset %d training time %.2g'%(i+1,elapsed_time))
    print('Quantization error: %.2g'%np.mean(individual_qe))


    p = figure(title='Uprate: %.2f'%(UPRATE_VALUE),width=600,height=600)


    p.circle(x=data[:, 0], y=data[:, 1],
           legend_label='Data',
           color=Vibrant3[1],alpha=0.2, size=8)

    p.circle(x=ng.particles[:, 0], y=ng.particles[:, 1],
           legend_label='Particles',
           color=Vibrant3[0],alpha=1, size=12)

    p.legend.click_policy="hide"
    show(p)
```

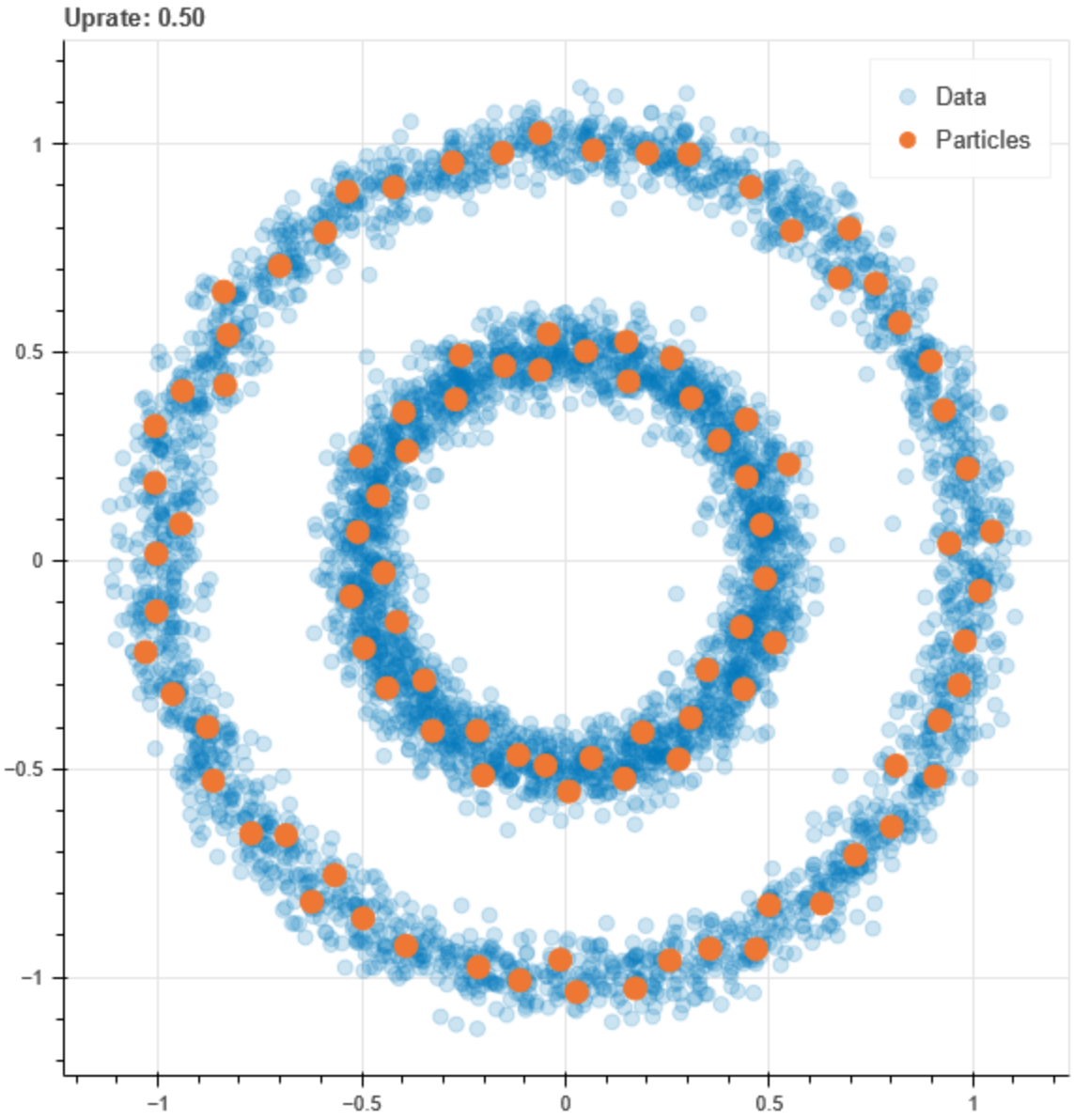
```
|████████████████████| 100.0%
Dataset 1 training time 10
Quantization error: 0.042
```



|████████████████████| 100.0%
Dataset 2 training time 10
Quantization error: 0.23



|████████████████████| 100.0%
Dataset 3 training time 14
Quantization error: 0.049



Calculation of quantization error

Average Quantization Error (QE) is calculated for this latter data (the *circles*) with respect to the last trained neural gas.

First method is based on the use of the `quantization_error` function within the `NeuGas` package.

In [ ]:
```
qe=quantization_error(ng.particles,data)
print('Quantization error is: %3g'%qe)
```

```
Quantization error is: 0.0494375
```

The same can be done by using the `quantize` **method** of the *NeuGas* object that calculate the individual quantization error of each sample in the dataset together with the *id* of the *Best Matching Unit* (BMU) for each sample.

In [ ]:
```
partcile_id,dist=ng.quantize(data)

print('Particle    Distance')
print('--------------------')
for i in range(20):
    print('%3d          %5.2g'%(partcile_id[i],dist[i]))
```

```
print('\n Average quantization error is: %3g'%np.mean(dist))
```

```
Particle    Distance
--------------------
 79          0.039
 86          0.028
 87          0.044
 51          0.033
 48          0.045
  1          0.039
 21          0.0084
 87          0.051
 47          0.061
 96           0.04
 32          0.033
 99          0.056
 71          0.082
 62          0.026
 40           0.14
 95          0.067
 15          0.064
 88          0.054
 15          0.073
 79          0.043

 Average quantization error is: 0.0494375
```

In [ ]: