Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

ЗВІТ

про виконання лабораторної роботи №3
з дисципліни «Інтелектуальний аналіз даних»

Виконала:

Студентка III курсу
Групи КА-76
Хиленко В.В.

Перевірила:
Недашківська Н.І.

Київ – 2020

Практикум 3. Кластеризація засобами бібліотеки Scikit-Learn Python

Постановка завдання
Варіант 17

Агломеративний алгоритм AgglomerativeClustering. Дослідити методи розрахунку відстані між кластерами: ward, single, average, complete. Побудувати матриці внутрішньокласових відстаней, використовуючи metrics.pairwise_distances.

Чи є розбиття стабільним після вилучення окремих об'єктів?

Метрики якості: Estimated number of clusters, Homogeneity, Completeness, V-measure.

Hierarchical clustering algorithms group similar objects into groups called clusters.
Agglomerative — Bottom up approach. Start with many small clusters and merge them together to create bigger clusters.

In [124]:
```
import numpy as np
```

In [125]:
```
np.random.seed(0)
X1 = np.random.randn(300, 2)
Y1 = np.logical_xor(X1[:, 0] > 0 , X1[:, 1] > 0)

from sklearn.datasets import make_moons
X2, Y2 = make_moons(n_samples=100, noise=0.1)
```

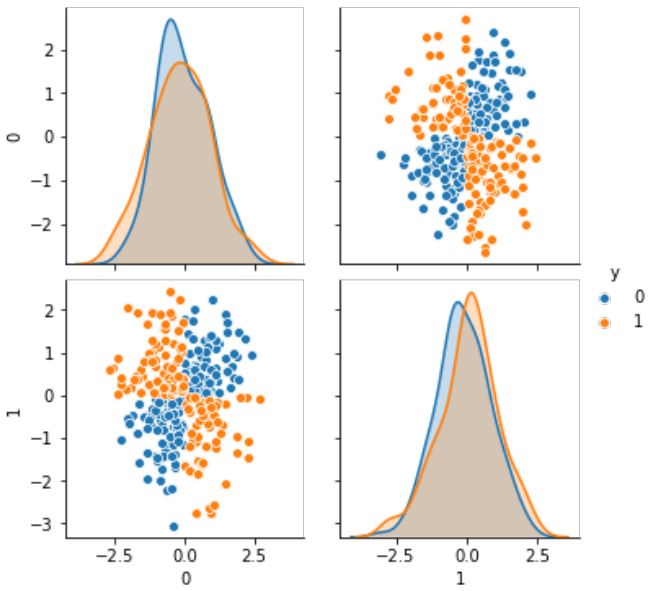
1. Представити початкові дані графічно.

XOR Dataset

In [126]:
```
%matplotlib inline
import seaborn as sns
import pandas as pd
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
```

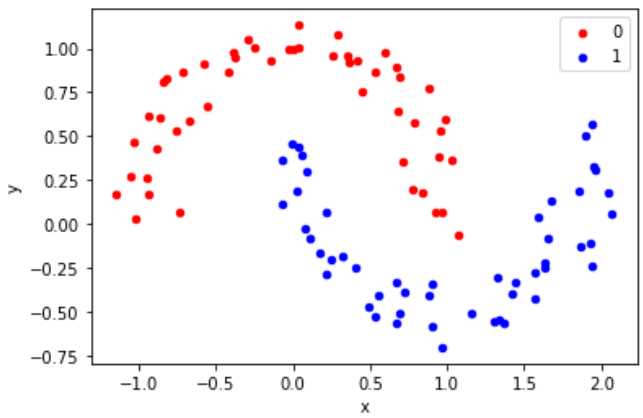
In [127]:
```
df1 = pd.DataFrame(X1)
df1['y'] = Y1.astype("int")

sns.pairplot(df1, hue='y')
```

Out[127]:
```
<seaborn.axisgrid.PairGrid at 0x7fd13cd4b550>
```



make_moons

In [128]:
```
df2 = pd.DataFrame(dict(x=X2[:,0], y=X2[:,1], label=Y2))
colors = {0:'red', 1:'blue'}
fig, ax = plt.subplots()
grouped = df2.groupby('label')
for key, group in grouped:
    group.plot(ax=ax, kind='scatter', x='x', y='y', label=key, color=colors
[key])
plt.show()
```



2.-4. Побудувати модель кластеризації згідно з варіантом. Виконати кластеризацію даних на основі моделі. Представити розбиття на кластери графічно, наприклад, різними кольорами.

In [129]:
```
from sklearn.cluster import AgglomerativeClustering
```

linkage{“ward”, “complete”, “average”, “single”}, default=”ward”

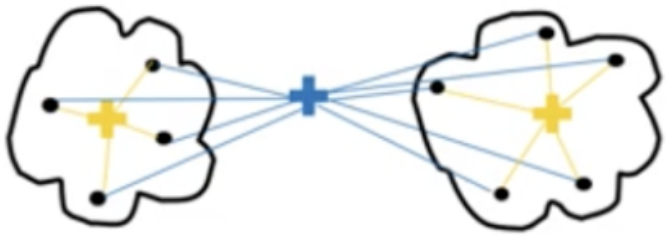
Which linkage criterion to use. The linkage criterion determines which distance to use between sets of observation. The algorithm will merge the pairs of cluster that minimize this criterion.

- ward minimizes the variance of the clusters being merged.
- average uses the average of the distances of each observation of the two sets.
- complete or maximum linkage uses the maximum distances between all observations of the two sets.
- single uses the minimum of the distances between all observations of the two sets.

- `single`: $d(U, V) := \min_{u \in U, v \in V} d(u, v)$
- `complete`: $d(U, V) := \max_{u \in U, v \in V} d(u, v)$
- `average`: $d(U, V) := \sum_{u \in U, v \in V} \frac{d(u,v)}{|U||V|}$
- `ward`: tries to minimize the variance in each cluster

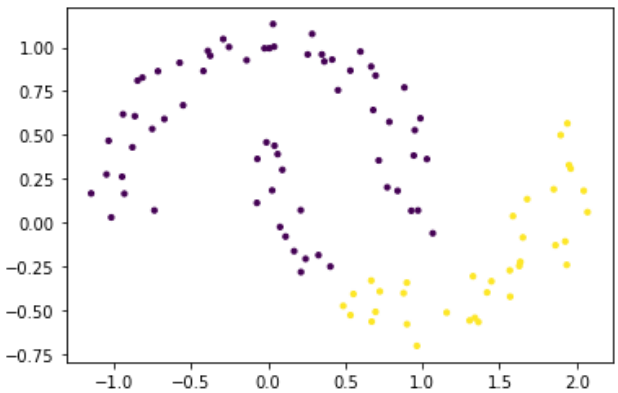
linkage ="ward"

The distance between clusters is the sum of squared differences within all clusters



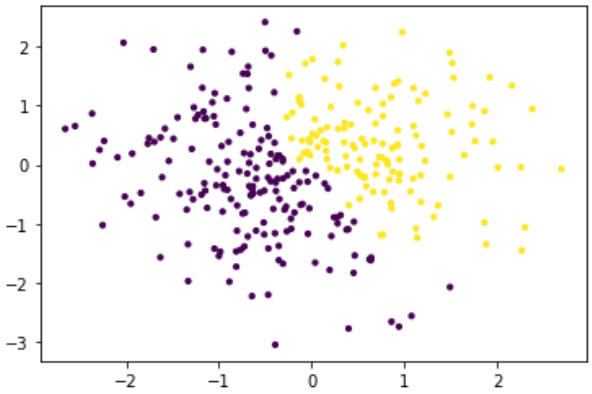
In [130]:
```
agg_clust = AgglomerativeClustering(n_clusters=2)
assigned_clust = agg_clust.fit_predict(X2)
plt.scatter(X2[:, 0], X2[:, 1], c=assigned_clust, s=10)
```

Out[130]:
```
<matplotlib.collections.PathCollection at 0x7fd134eaf080>
```



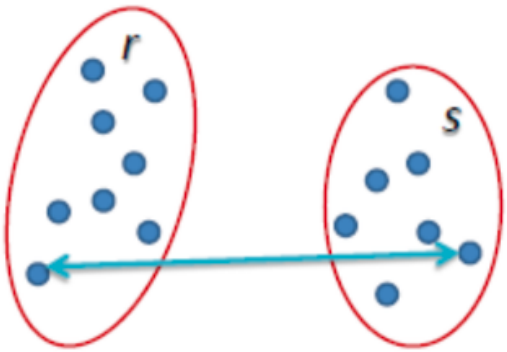
In [131]:
```
agg_clust = AgglomerativeClustering(n_clusters=2)
assigned_clust = agg_clust.fit_predict(X1)
plt.scatter(X1[:, 0], X1[:, 1], c=assigned_clust, s=10)
```

Out[131]:
```
<matplotlib.collections.PathCollection at 0x7fd12ee24780>
```



linkage ="complete"

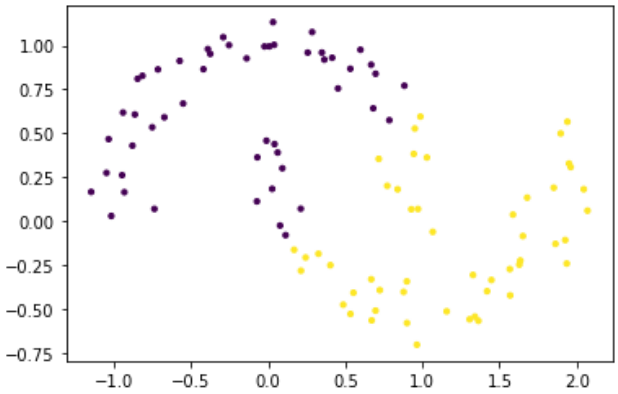
The distance between two clusters is the longest distance between two points in each cluster



$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

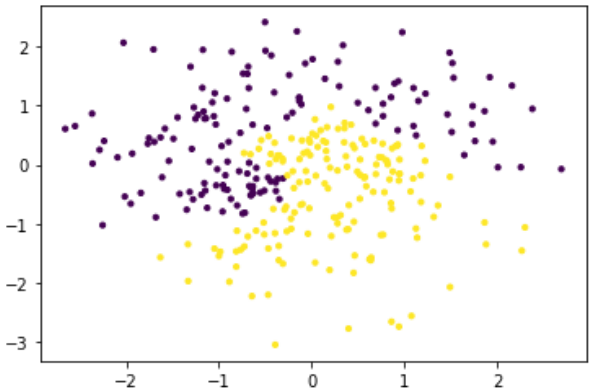
In [132]:
```
agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="complete")
assigned_clust = agg_clust.fit_predict(X2)
plt.scatter(X2[:, 0], X2[:, 1], c=assigned_clust, s=10)
```

Out[132]: <matplotlib.collections.PathCollection at 0x7fd12e75f320>



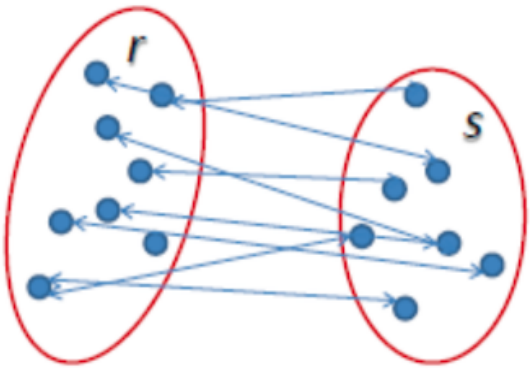
In [133]:
```
agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="complete")
assigned_clust = agg_clust.fit_predict(X1)
plt.scatter(X1[:, 0], X1[:, 1], c=assigned_clust, s=10)
```

Out[133]: <matplotlib.collections.PathCollection at 0x7fd12e72ac18>



linkage ="average"

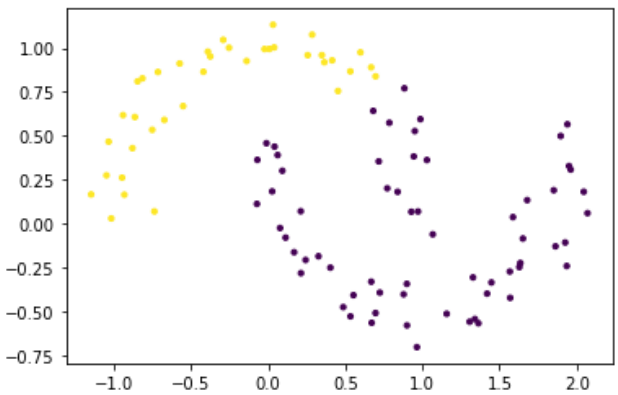
The distance between clusters is the average distance between each point in one cluster to every point in other cluster



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

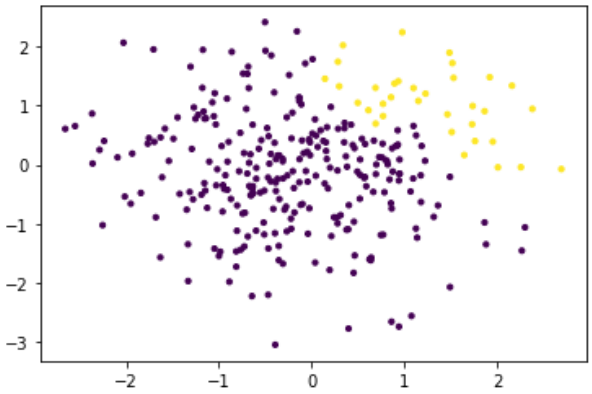
In [134]:
```
agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="average")
assigned_clust = agg_clust.fit_predict(X2)
plt.scatter(X2[:, 0], X2[:, 1], c=assigned_clust, s=10)
```

Out[134]: <matplotlib.collections.PathCollection at 0x7fd12e68c4a8>



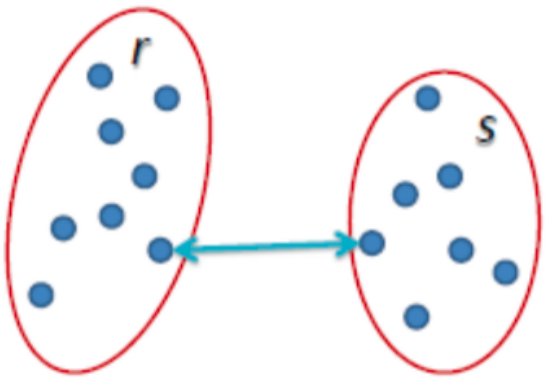
In [135]:
```
agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="average")
assigned_clust = agg_clust.fit_predict(X1)
plt.scatter(X1[:, 0], X1[:, 1], c=assigned_clust, s=10)
```

Out[135]:
```
<matplotlib.collections.PathCollection at 0x7fd12e673860>
```



linkage ="single"

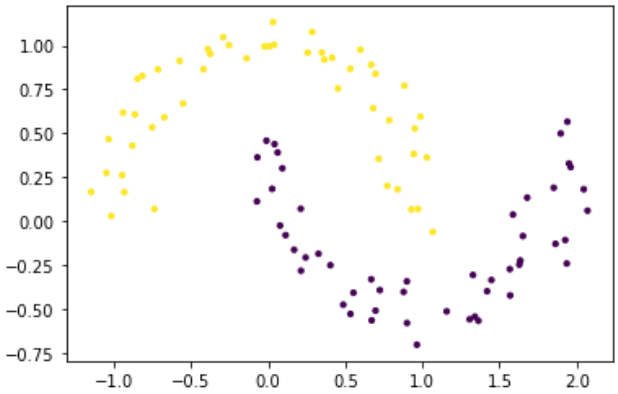
The distance between two clusters is the shortest distance between two points in each cluster



$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

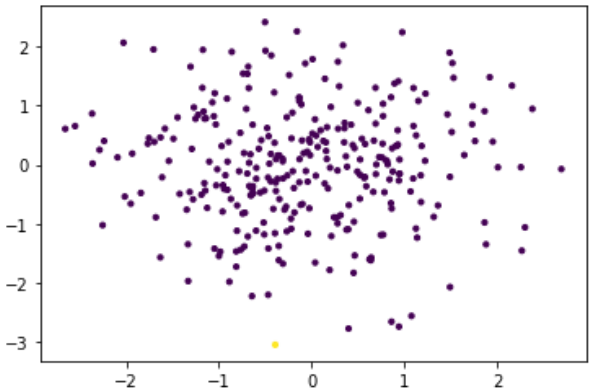
In [136]:
```
agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="single")
assigned_clust = agg_clust.fit_predict(X2)
plt.scatter(X2[:, 0], X2[:, 1], c=assigned_clust, s=10)
```

Out[136]: <matplotlib.collections.PathCollection at 0x7fd12e5db080>



In [137]:
```
agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="single")
assigned_clust = agg_clust.fit_predict(X1)
plt.scatter(X1[:, 0], X1[:, 1], c=assigned_clust, s=10)
```

Out[137]: <matplotlib.collections.PathCollection at 0x7fd12e53e470>



Для make_moons найбільш ефективним розрахунком відстані є single, для xor - complete.

5. Розрахувати додаткові результати кластеризації згідно з варіантом.

In [138]:
```
from sklearn.metrics import pairwise_distances
```

Матриця внутрішньокласових відстаней для make_moons

In [139]:
```
dataf2 = pd.DataFrame(pairwise_distances(X2))
dataf2
```

Out[139]:

	0	1	2	3	4	5	6	7	8	9	..
0	0.000000	1.123844	1.832366	1.770773	1.590234	1.888931	0.211950	0.914110	1.787266	0.820441	..
1	1.123844	0.000000	2.956083	2.869712	2.698483	2.997193	1.293514	0.726741	2.895192	0.584242	..
2	1.832366	2.956083	0.000000	0.453831	0.470519	0.359811	1.678289	2.590740	0.356614	2.569354	..
3	1.770773	2.869712	0.453831	0.000000	0.829323	0.159977	1.659555	2.408506	0.098296	2.417166	..
4	1.590234	2.698483	0.470519	0.829323	0.000000	0.794720	1.405584	2.437041	0.750112	2.385030	..
...	...	...	...	...	...	...	...	...	...	...	..
95	0.761338	0.368116	2.593030	2.521458	2.330420	2.644864	0.925772	0.644951	2.542941	0.467274	..
96	1.206290	0.313335	3.017521	2.977034	2.718514	3.094303	1.339533	1.013870	2.992813	0.854390	..
97	0.144534	1.012284	1.948044	1.862275	1.722703	1.986514	0.356377	0.769823	1.884544	0.677189	..
98	0.830379	1.713063	1.579795	1.731837	1.173207	1.789032	0.632670	1.722814	1.701873	1.601379	..
99	0.745801	1.484872	1.798419	1.556299	1.746722	1.708140	0.861936	0.880882	1.611864	0.935325	..

100 rows × 100 columns

Матриця внутрішньокласових відстаней для XOR

In [140]:
```
dataf1 = pd.DataFrame(pairwise_distances(X1))
dataf1
```

Out[140]:

	0	1	2	3	4	5	6	7	8	9
0	0.000000	2.001257	1.381319	0.983212	1.867300	1.932767	1.040957	1.321862	0.662791	1.917943
1	2.001257	0.000000	3.338656	2.392422	2.126172	1.146946	2.130371	1.980802	2.499749	3.165766
2	1.381319	3.338656	0.000000	1.234462	2.410428	2.980427	1.559514	1.935330	0.857703	1.559363
3	0.983212	2.392422	1.234462	0.000000	1.193839	1.796596	0.332095	0.701085	0.546645	0.948492
4	1.867300	2.126172	2.410428	1.193839	0.000000	1.072565	0.911272	0.552464	1.711875	1.331445
...	...	...	...	...	...	...	...	...	...	...
295	1.221360	1.853715	1.938967	0.716965	0.647156	1.090921	0.384873	0.144888	1.148457	1.313476
296	2.477393	3.534265	2.089828	1.495969	1.499155	2.564218	1.554116	1.565251	1.925856	0.580125
297	1.412427	1.964394	2.034136	0.799688	0.456079	1.094925	0.483559	0.102738	1.283107	1.233939
298	2.317690	2.852868	2.458203	1.421625	0.726702	1.780266	1.276736	1.066842	1.964146	1.009693
299	2.755668	3.081846	2.909486	1.880442	1.015366	1.954519	1.717643	1.466405	2.424381	1.414891

300 rows × 300 columns

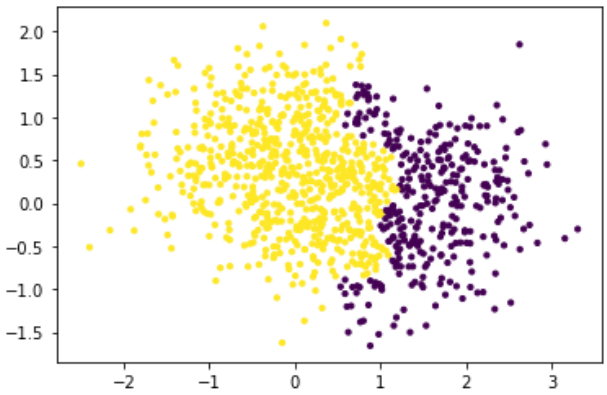
6. Побудувати декілька альтернативних моделей:

- шляхом зміни значень параметрів основної моделі,
- використати різні функції відстані,
- задати різні значення кількості кластерів, в алгоритмах де кількість кластерів - параметр.

make_moons, linkage ="complete", n_samples=1000, noise=0.5

In [141]:
```
X2_1, Y2_1 = make_moons(n_samples=1000, noise=0.5)
agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="complete")
assigned_clust = agg_clust.fit_predict(X2_1)
plt.scatter(X2_1[:, 0], X2_1[:, 1], c=assigned_clust, s=10)
```

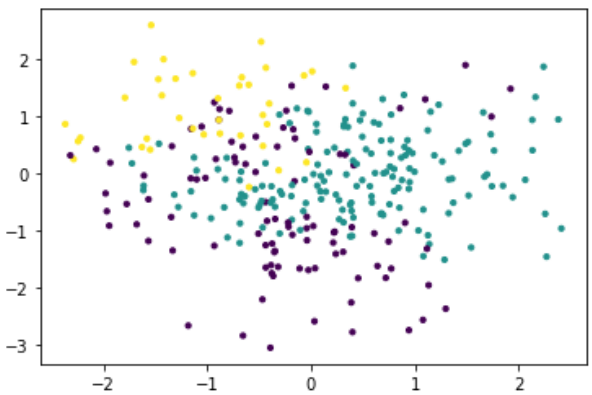
Out[141]: <matplotlib.collections.PathCollection at 0x7fd12e526198>



XOR, n_clusters=3, linkage ="complete"

In [142]:
```
np.random.seed(0)
X1_1 = np.random.randn(300, 3)
Y1_1 = np.logical_xor(X1_1[:, 0] > 0 , X1_1[:, 1] > 0)

agg_clust = AgglomerativeClustering(n_clusters=3, linkage ="complete")
assigned_clust = agg_clust.fit_predict(X1_1)
plt.scatter(X1_1[:, 0], X1_1[:, 1], c=assigned_clust, s=10)
```

Out[142]: <matplotlib.collections.PathCollection at 0x7fd12e47cfd0>



7. Для кожної альтернативної моделі розрахувати метрики якості кластеризації: Estimated number of clusters, Homogeneity, Completeness, V-measure.

- homogeneity: each cluster contains only members of a single class.
- completeness: all members of a given class are assigned to the same cluster.
- V-Measure: the harmonic mean of homogeneity h and completeness c of the clustering.

In [143]:
```
from sklearn.metrics import homogeneity_score, completeness_score, v_measure
_score
```

In [144]:
```
def metric(true_labels, labels):
    # Number of clusters in labels, ignoring noise if present.
    n_clusters_ = len(set(labels)) - (1 if -1 in labels else 0)

    print('Estimated number of clusters: %d' % n_clusters_)
    print("Homogeneity: %0.3f" % homogeneity_score(true_labels, labels))
    print("Completeness: %0.3f" % completeness_score(true_labels, labels))
    print("V-measure: %0.3f" % v_measure_score(true_labels, labels))
```

make_moons, linkage ="complete", n_samples=1000, noise=0.5

In [145]:
```
metric(Y2_1, agg_clust.fit(X2_1).labels_)
```

```
Estimated number of clusters: 3
Homogeneity: 0.153
Completeness: 0.127
V-measure: 0.139
```

XOR, n_clusters=3, linkage ="average"

In [146]:
```
metric(Y1_1, agg_clust.fit(X1_1).labels_)
```

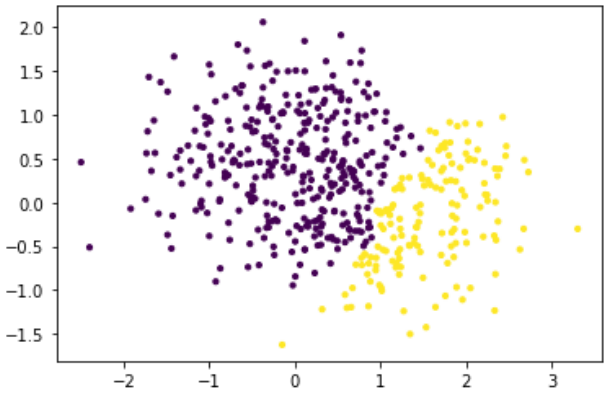
```
Estimated number of clusters: 3
Homogeneity: 0.080
Completeness: 0.058
V-measure: 0.067
```

8. Виконати аналіз результатів кластеризації. Чи є розбиття стабільним після вилучення окремих об’єктів?

make_moons, linkage ="complete", n_samples=1000, noise=0.5

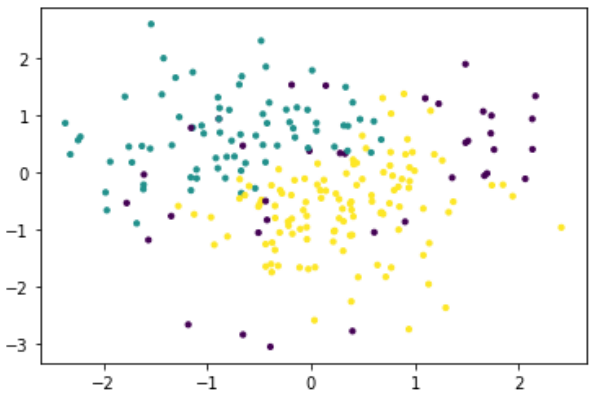
In [147]:
```
idx_not_to_delete = np.array([x for x in range(len(X2_1))])
np.random.shuffle(idx_not_to_delete)
idx_not_to_delete = idx_not_to_delete[500:]
idx_not_to_delete
X2_2 = X2_1[idx_not_to_delete]

agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="complete")
assigned_clust = agg_clust.fit_predict(X2_2)
plt.scatter(X2_2[:, 0], X2_2[:, 1], c=assigned_clust, s=10)
```

Out[147]: <matplotlib.collections.PathCollection at 0x7fd12e460630>



XOR, n_clusters=3, linkage ="complete"

In [148]:
```
idx_not_to_delete = np.array([x for x in range(len(X1_1))])
np.random.shuffle(idx_not_to_delete)
idx_not_to_delete = idx_not_to_delete[70:]
idx_not_to_delete
X1_2 = X1_1[idx_not_to_delete]

agg_clust = AgglomerativeClustering(n_clusters=3, linkage ="complete")
assigned_clust = agg_clust.fit_predict(X1_2)
plt.scatter(X1_2[:, 0], X1_2[:, 1], c=assigned_clust, s=10);
```



Розбиття не є стабільним після вилучення окремих об’єктів, бо як ми бачимо з графіків, розбиття змінилось.

9. Зробити висновки про якість роботи моделей на досліджених даних. Дослідити різні значення параметрів основної моделі, різні функції відстані та різну кількість кластерів в алгоритмах, де кількість кластерів слугує параметром.

Виконаємо решітчатий пошук, для знаходження оптимальних параметрів моделі.

v_measure: score between 0.0 and 1.0. 1.0 stands for perfectly complete labeling

In [149]:
```
num = [1, 2, 3, 4]
linkage = ["ward", "complete", "average", "single"]
```

In [150]:
```
def grid_search(X, labels_true):
    param=[]
    esimated_cluster_num = []
    v_measure = []
    for n in num:
        for link in linkage:
            agg_clust = AgglomerativeClustering(n_clusters = n, linkage = li
nk).fit(X)
            param.append([n, link])
            labels = agg_clust.labels_
            n_clusters_ = len(set(labels)) - (1 if -1 in labels else 0)
            esimated_cluster_num.append(n_clusters_)
            v_measure.append(v_measure_score(labels_true, labels))

    return pd.DataFrame({"params[n, linkage]": param, "v_measure_score": v_m
easure})
```

make_moons dataset

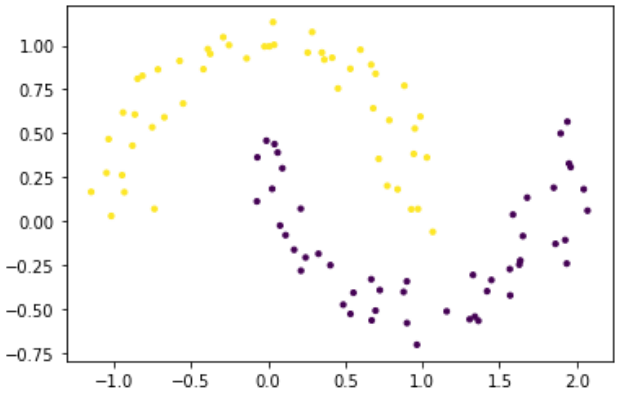
In [151]:
```
agg_clust2 = grid_search(X2, Y2).set_index("params[n, linkage]")
pd.options.display.float_format = '{:.6f}'.format
agg_clust2_sort = agg_clust2.sort_values(["v_measure_score"], ascending=Fals
e)
agg_clust2_sort.head()
```

Out[151]:

params[n, linkage]	v_measure_score
[2, single]	1.000000
[3, single]	0.800185
[4, single]	0.723562
[2, average]	0.550886
[2, ward]	0.510243

In [152]:
```
agg_clust = AgglomerativeClustering(n_clusters=2, linkage ="single")
assigned_clust = agg_clust.fit_predict(X2)
plt.scatter(X2[:, 0], X2[:, 1], c=assigned_clust, s=10)
```

Out[152]: <matplotlib.collections.PathCollection at 0x7fd12e3a0a90>



XOR dataset

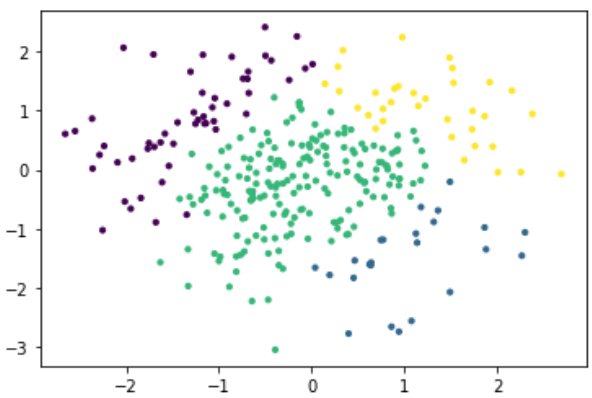
In [153]:
```
agg_clust1 = grid_search(X1, Y1).set_index("params[n, linkage]")
pd.options.display.float_format = '{:.6f}'.format
agg_clust1_sort = agg_clust1.sort_values(["v_measure_score"], ascending=Fals
e)
agg_clust1_sort.head()
```

Out[153]:

params[n, linkage]	v_measure_score
[4, average]	0.190266
[3, average]	0.126124
[4, ward]	0.109957
[2, average]	0.084607
[4, complete]	0.056164

In [155]:
```
agg_clust = AgglomerativeClustering(n_clusters=4, linkage ="average")
assigned_clust = agg_clust.fit_predict(X1)
plt.scatter(X1[:, 0], X1[:, 1], c=assigned_clust, s=10)
```

Out[155]: <matplotlib.collections.PathCollection at 0x7fd12e2e3c88>



10. Оцінити результати кластеризації на основі метрик якості та на основі неформальних методів. Для кожного набору даних вибрати найкращу модель.

1. Найкращі параметри для AgglomerativeClustering:

Датасет make_moons

n_clusters=2, linkage ="single"

XOR датасет

n_clusters=4, linkage ="average"

2. Вилучення окремих об’єктів у датасеті, впливає на якість кластеризації.

3. Висновок

Я обирала найкращу модель завдяки метриці V-Measure

$$V_\beta = \frac{(1+\beta)hc}{\beta h+c}$$

h - homogeneity c - completeness
Чим більше ця метрика, тим краща модель. Здійснивши решітчатий пошук можна зробити висновок про те які параметри є оптимальними. З графіків видно, що це дійсно так.