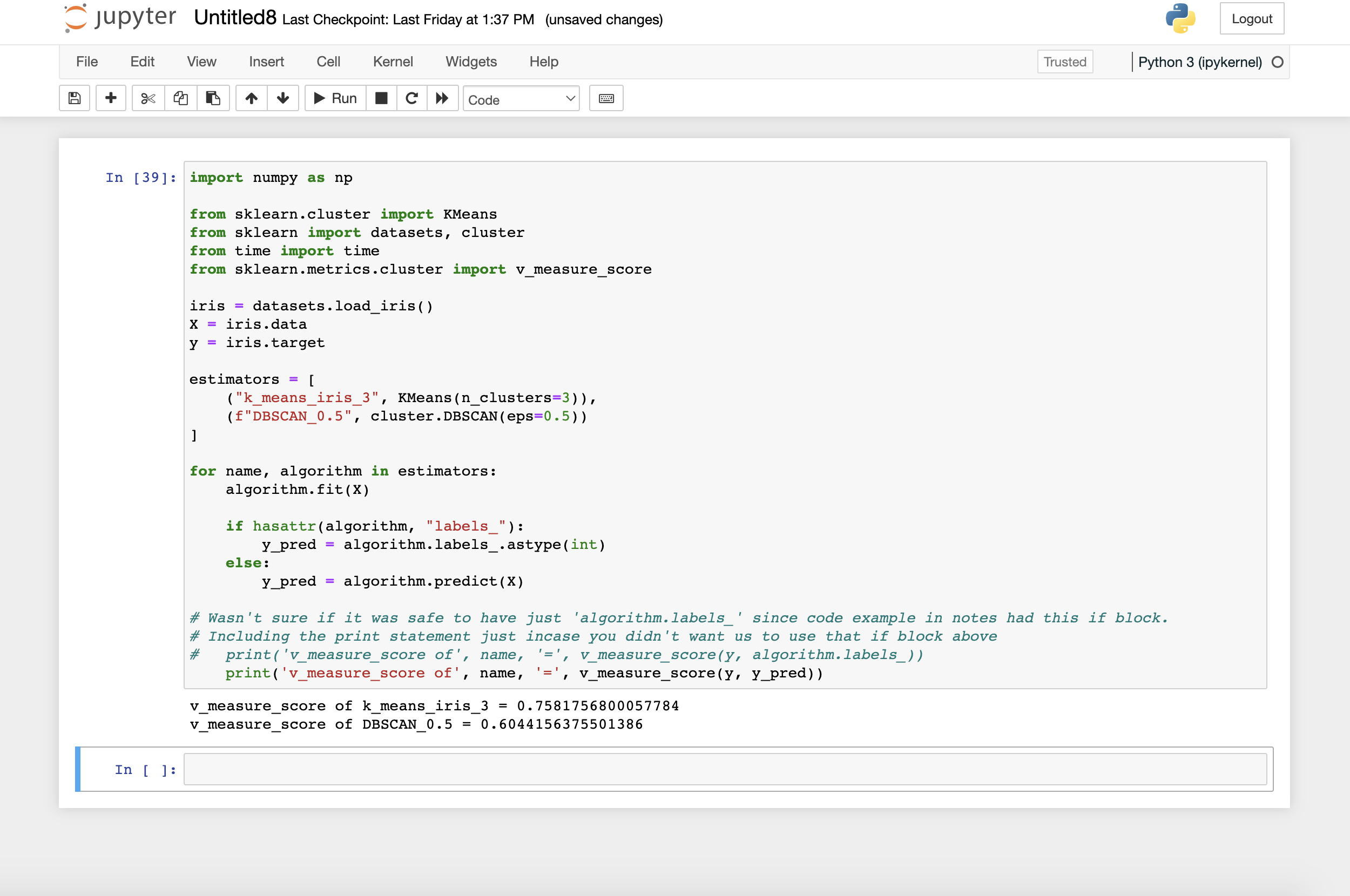
1. **Fit the 2 estimators with X and evaluate their performance by** [**v\_measure\_score**](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.v_measure_score.html)**. Write down the code and compare its performance. (1 point)**

**Output:**



v\_measure\_score of k\_means\_iris\_3 = 0.7581756800057784  
 v\_measure\_score of DBSCAN\_0.5 = 0.6044156375501386

**Code:**

import numpy as np

from sklearn.cluster import KMeans

from sklearn import datasets, cluster

from time import time

from sklearn.metrics.cluster import v\_measure\_score

iris = datasets.load\_iris()

X = iris.data

y = iris.target

estimators = [

("k\_means\_iris\_3", KMeans(n\_clusters=3)),

(f"DBSCAN\_0.5", cluster.DBSCAN(eps=0.5))

]

for name, algorithm in estimators:

algorithm.fit(X)

if hasattr(algorithm, "labels\_"):

y\_pred = algorithm.labels\_.astype(int)

else:

y\_pred = algorithm.predict(X)

# Wasn't sure if it was safe to have just 'algorithm.labels\_' since code example in notes had that if block above.

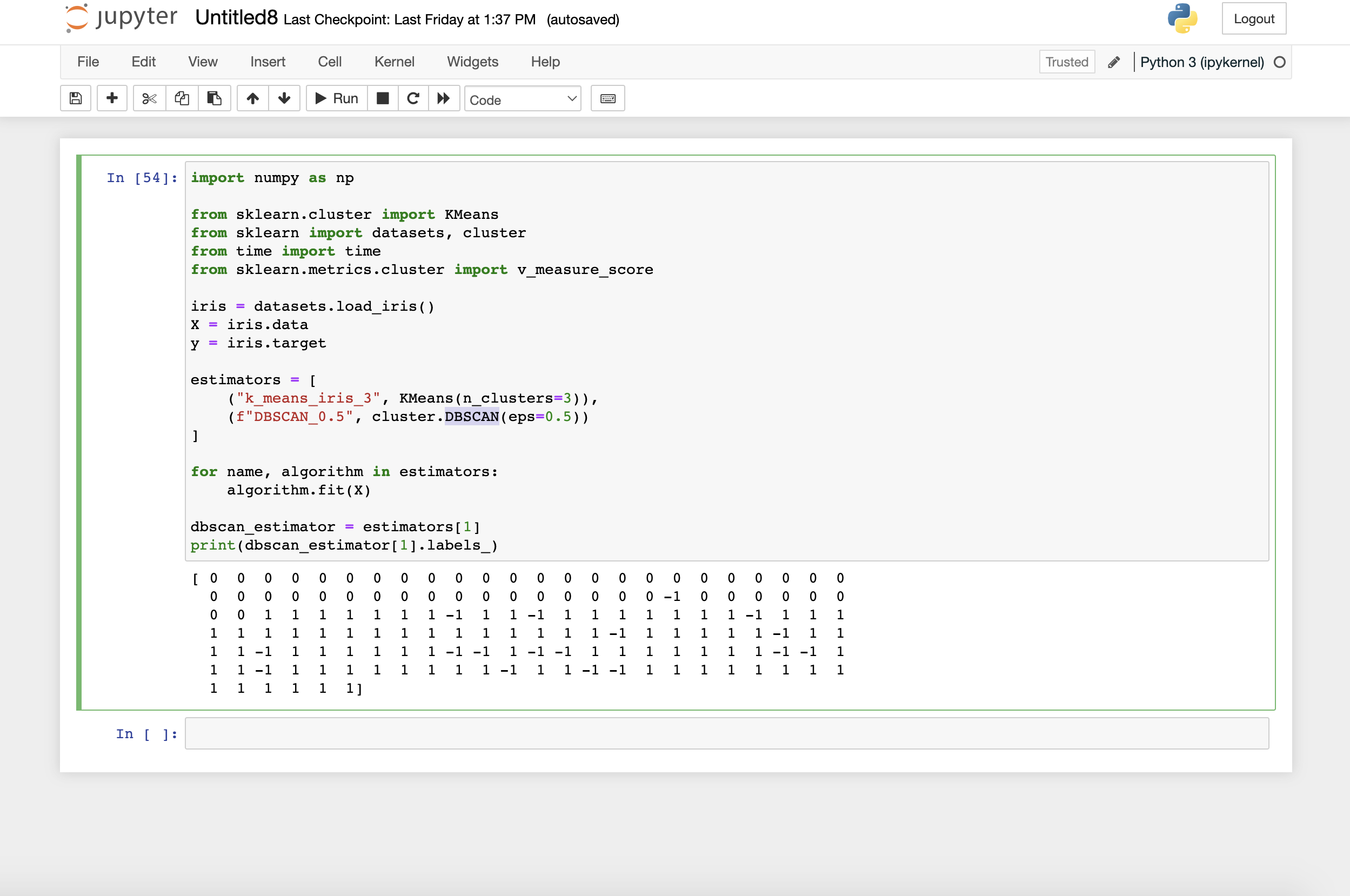
# Including the print statement just in case you didn't want us to use that if block above

# print('v\_measure\_score of', name, '=', v\_measure\_score(y, algorithm.labels\_))

print('v\_measure\_score of', name, '=', v\_measure\_score(y, y\_pred))

1. **After fitting the 2 estimators, print the labels of the DBSCAN estimator. If your DBSCAN estimator is stored in variable est, you can get the labels by running print(est.labels\_). You will see many -1 in the list. What are those data points with the -1 label? If you change eps to 1, do you still see it? Can you explain the change? (2 points)**

**Output:**



Eps=0.5:

[ 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -1 0 0 0 0 0 0  
 0 0 1 1 1 1 1 1 1 -1 1 1 -1 1 1 1 1 1 1 1 -1 1 1 1  
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 -1 1 1 1 1 1 -1 1 1  
 1 1 -1 1 1 1 1 1 1 -1 -1 1 -1 -1 1 1 1 1 1 1 1 -1 -1 1  
 1 1 -1 1 1 1 1 1 1 1 1 -1 1 1 -1 -1 1 1 1 1 1 1 1 1  
 1 1 1 1 1 1]

Eps=1:

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
 1 1]

The –1 values in the first output above are data points that are considered noisy by the algorithm and can be ignored. If a data point doesn’t have enough neighbors within a specified range (eps), it will be classified as noisy by the DBSCAN algorithm and given the –1 value.

Changing the eps parameter from 0.5 to 1 removes the –1 values from the array, as can be seen from the 2nd output above. The reasoning for this is we just increased the maximum distance between 2 samples that would consider one of the data points to be in the neighborhood of the other. The result is that these data points that were noisy at 0.5 are not considered noisy anymore at 1 because they are now within the maximum range.

References:

1. <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html>
2. <https://stackoverflow.com/questions/45313176/what-are-noisy-samples-in-scikits-dbscan-clustering-algorithm#:~:text=They%20are%20simply%20points%20that,the%20points%20into%20the%20clusters>.

**Code:**

import numpy as np

from sklearn.cluster import KMeans

from sklearn import datasets, cluster

from time import time

from sklearn.metrics.cluster import v\_measure\_score

iris = datasets.load\_iris()

X = iris.data

y = iris.target

estimators = [

("k\_means\_iris\_3", KMeans(n\_clusters=3)),

(f"DBSCAN\_0.5", cluster.DBSCAN(eps=0.5))

]

for name, algorithm in estimators:

algorithm.fit(X)

dbscan\_estimator = estimators[1]

print(dbscan\_estimator[1].labels\_)