Fourth Industrial Summer School

Module 4: ML

Unsupervised Learning

Outlines

- ✓ ML Unsupervised Learning
  - ✓ What is Clustering?
  - ✓ Access data using Sci-kit learn (sklearn)

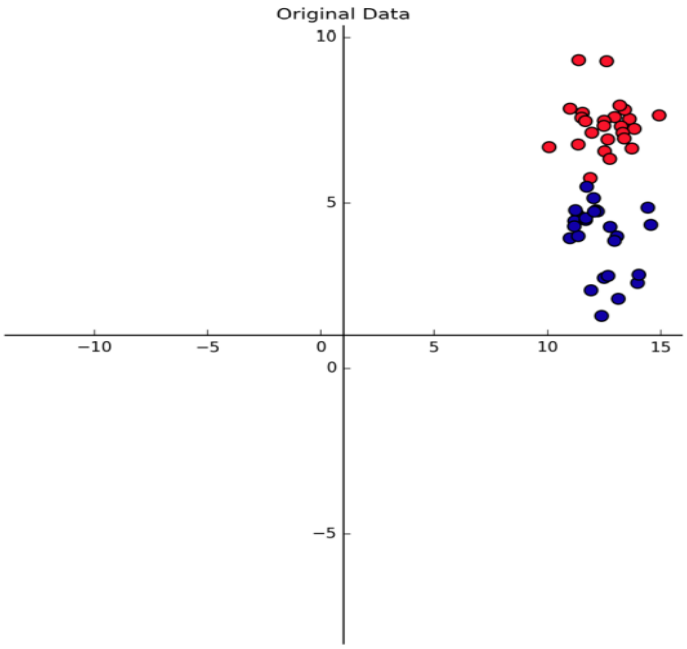
What is unsupervised learning?

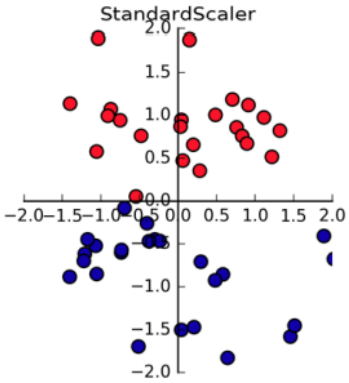
- Unsupervised learning refers to algorithms that learn patterns from unlabeled data **(no known output)**

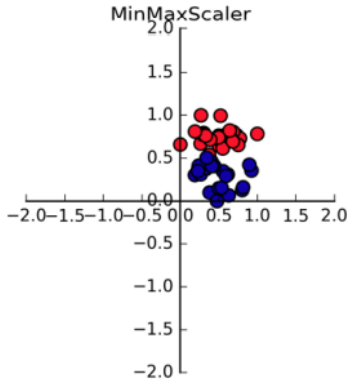
- So, in unsupervised, the learning algorithm is just fed with input data and asked **to extract some knowledge (patterns)**

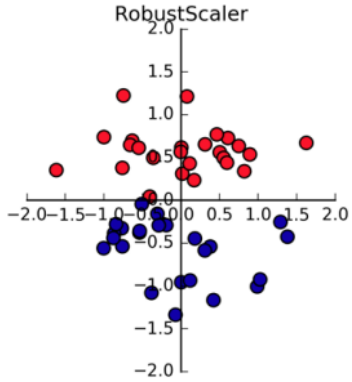
- There are two types of unsupervised learning in this view
  - Data Transformation such as PCA
  - Data Clustering analysis

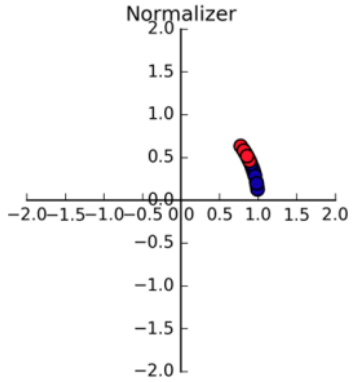
Other Data Transformation







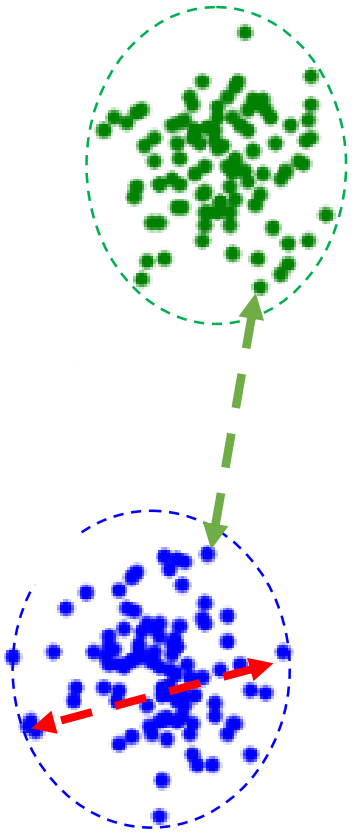




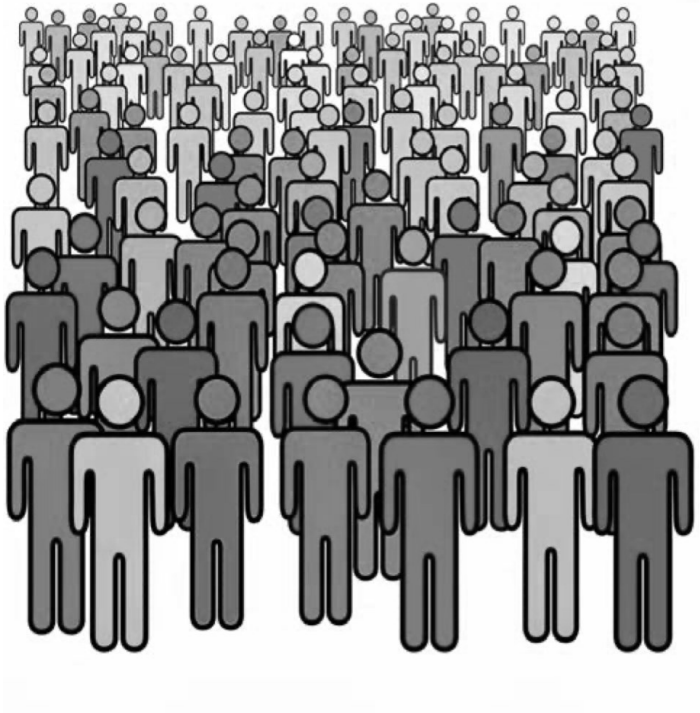
- sklearn.preprocessing.StandardScaler
- sklearn.preprocessing.MinMaxScaler
- sklearn.preprocessing.RobustScaler
- sklearn.preprocessing.Normalizer
- etc

What is clustering ?

- **Clustering** is a technique used to group similar data points together.
- Each point in a group share two main properties
  1. It is similar to samples within a group (i.e., related), and
  2. It is different to samples of the other groups (i.e., unrelated)

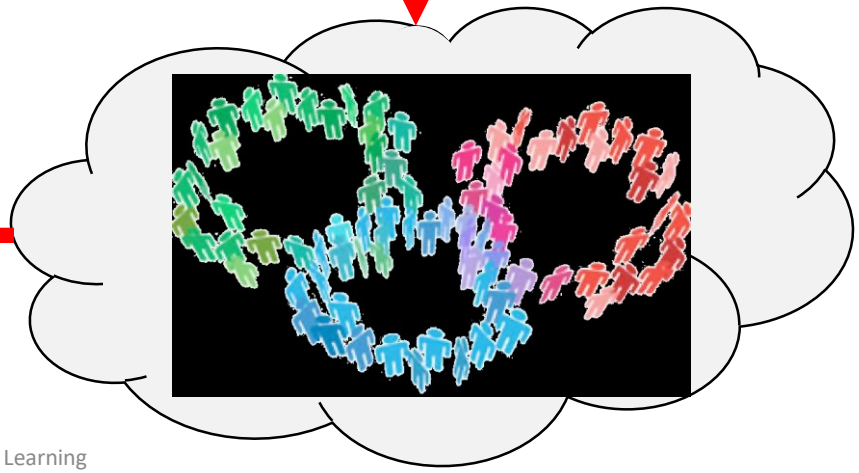


Marketing (Example)



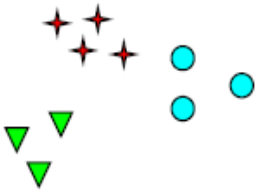
Clustering Task

Customers Groups

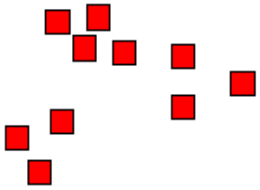


Clusters is tough to find

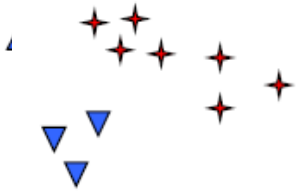
How many clusters?



Six Clusters



Two Clusters



Four Clusters

Key Challenge in Unsupervised Learning

- **Evaluation (validation)** is a challenge in unsupervised learning!
  - Data samples are not associated with label information!
  - Clustering algorithms always produce answers!
  - The right answer is unknown??

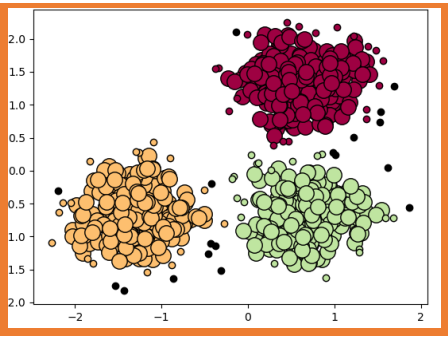
Clustering Applications

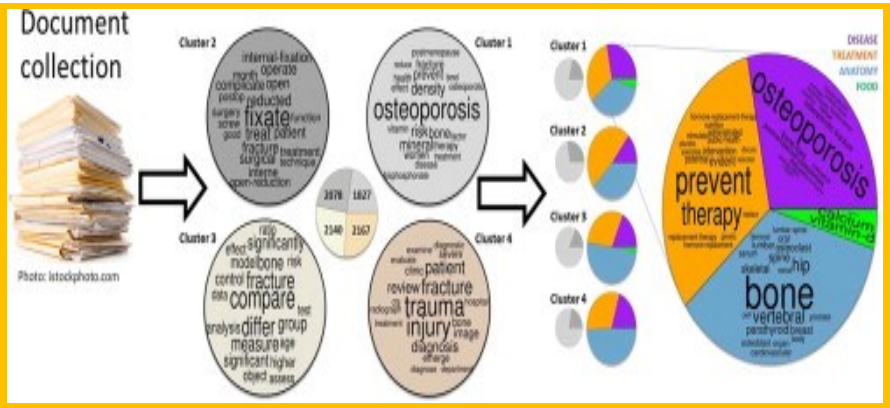
Image segmentation:

Clustering can be used for object detection or image segmentation as showed in the figure

Outlier detection

The black dots showed in this Figure is outliers detected by a clustering algorithm



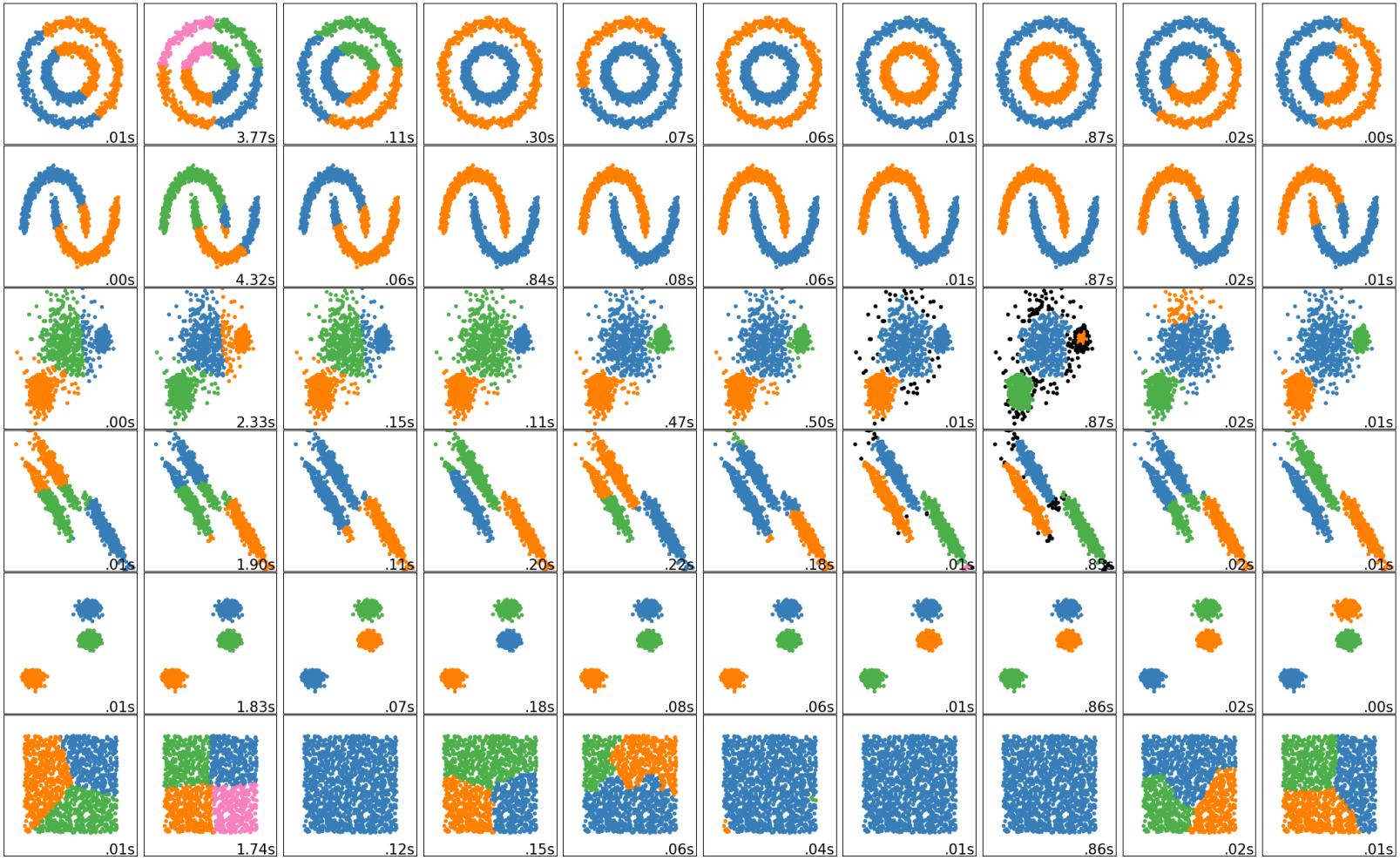


Labeling data

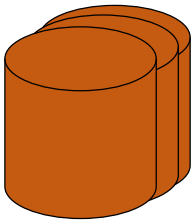
For example, based on documents, analyze content and label by theme

https://www.sciencedirect.com/science/article/pii/S1532046415000556#f0030

2D data shapes



Source: https://scikit-learn.org/stable/modules/clustering.html



Datasets

Generate, load, or access datasets using Scikit-learn

Scikit Learn

- Scikit learn is **machine learning** library for Python
- It supports several algorithms for clustering, classification, and regression problems
- It is adequately integrated Numpy and Scipy libraries
- The library is updated regularly (current stable version is 1.5, check here)

Generate/load datasets

- Scikit learn provides many useful packages that help to generate and get data for learning purposes.

- ***Datasets*** package is our good starting point with Sklearn, we can:
  - Generate different forms of datasets
  - Load predefined benchmarked datasets (i.e., toy datasets)
  - Fetch datasets of larger sizes

Generate datasets

datasets.make_biclusters(shape, n_clusters, *)	Generate an array with constant block diagonal structure for biclustering.
datasets.make_blobs([n_samples, n_features, ...])	Generate isotropic Gaussian blobs for clustering.
datasets.make_checkerboard(shape, n_clusters, *)	Generate an array with block checkerboard structure for biclustering.
datasets.make_circles([n_samples, shuffle, ...])	Make a large circle containing a smaller circle in 2d.
datasets.make_classification([n_samples, ...])	Generate a random n-class classification problem.
datasets.make_friedman1([n_samples, ...])	Generate the "Friedman #1" regression problem.
datasets.make_friedman2([n_samples, noise, ...])	Generate the "Friedman #2" regression problem.
datasets.make_friedman3([n_samples, noise, ...])	Generate the "Friedman #3" regression problem.
datasets.make_gaussian_quantiles(*[, mean, ...])	Generate isotropic Gaussian and label samples by quantile.
datasets.make_hastie_10_2([n_samples, ...])	Generates data for binary classification used in Hastie et al. 2009, Example 10.2.
datasets.make_low_rank_matrix([n_samples, ...])	Generate a mostly low rank matrix with bell-shaped singular values.
datasets.make_moons([n_samples, shuffle, ...])	Make two interleaving half circles.
datasets.make_multilabel_classification([...])	Generate a random multilabel classification problem.
datasets.make_regression([n_samples, ...])	Generate a random regression problem.
datasets.make_s_curve([n_samples, noise, ...])	Generate an S curve dataset.
datasets.make_sparse_coded_signal(n_samples, ...)	Generate a signal as a sparse combination of dictionary elements.
datasets.make_sparse_spd_matrix([dim, ...])	Generate a sparse symmetric definite positive matrix.
datasets.make_sparse_uncorrelated([...])	Generate a random regression problem with sparse uncorrelated design.
datasets.make_spd_matrix(n_dim, *[, ...])	Generate a random symmetric, positive-definite matrix.
datasets.make_swiss_roll([n_samples, noise, ...])	Generate a swiss roll dataset.

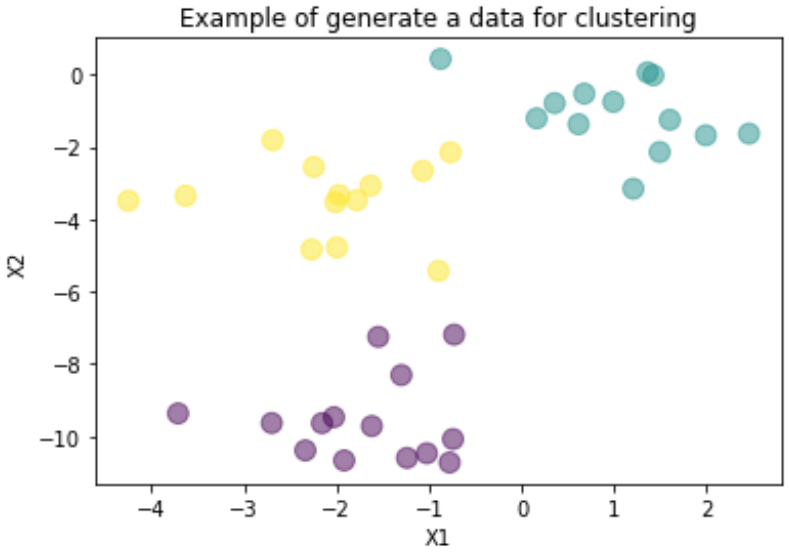
https://scikit-learn.org/stable/modules/classes.html?highlight=datasets#module-sklearn.datasets

Generate data make_blobs

```
from sklearn import datasets
X, y = datasets.make_blobs(n_samples=40,
                           centers=3,
                           n_features=2,
                           cluster_std=1.0,
                           center_box=(-10.0, 10.0),
                           shuffle=True,
                           random_state=None)
```

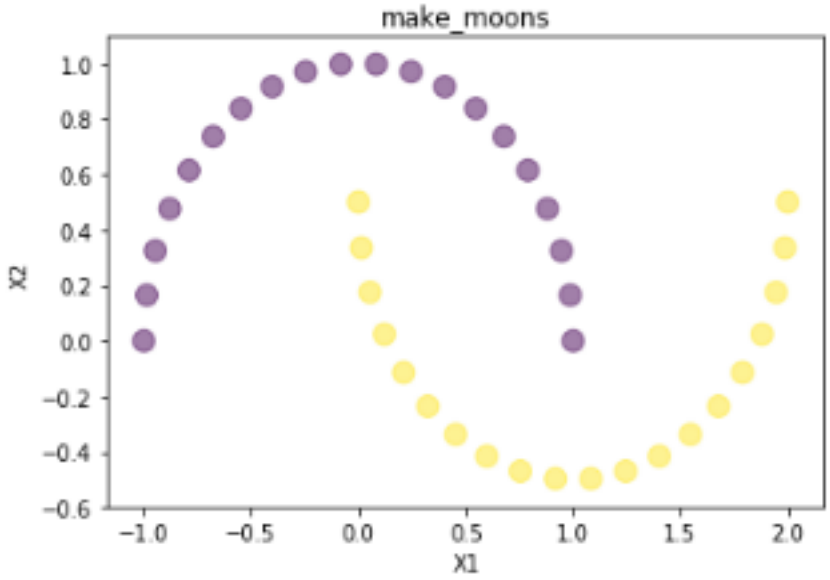
Generate data make_blobs

```
from sklearn import datasets
Xb, yb = datasets.make_blobs(n_samples=40,
                             centers=3,
                             n_features=2,
                             cluster_std=1.0,
                             center_box=(-10.0, 10.0),
                             shuffle=True,
                             [redacted]
```



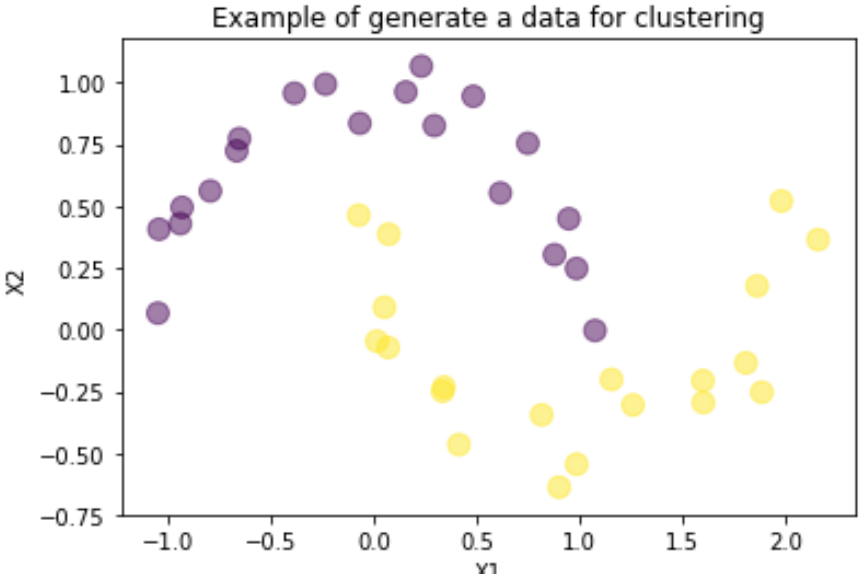
Generate data make_moons

```
Xmn, ymn = datasets.make_moons(n_samples=40,
                               shuffle=True,
                               noise=None,
                               random_state=None)
```



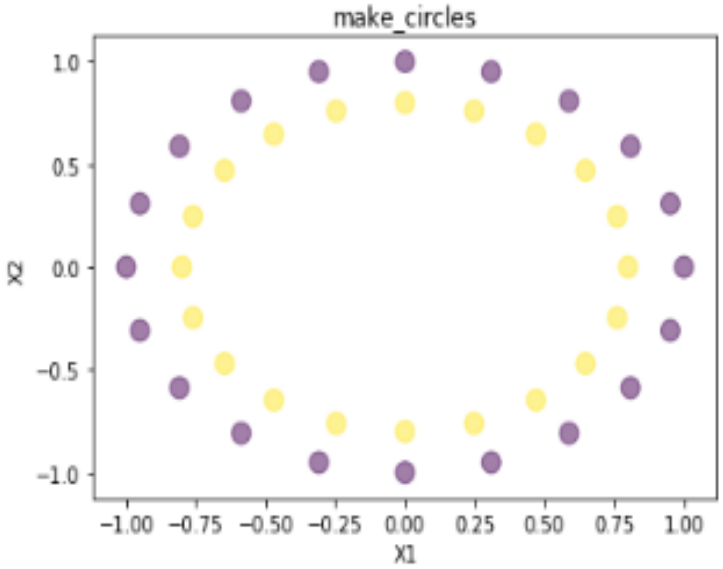
Generate data make_moon

```
Xmn, ymn = datasets.make_moons(n_samples=40,
                               shuffle=True,
                               noise=0.1,
                               [redacted]
```



Generate data make_circles.

```
Xc, yc = datasets.make_circles(n_samples=40,
                               shuffle=True,
                               noise=None,
                               random_state=None,
                               factor=0.8)
```



Generate data make_circles.

```
Xc, yc = datasets.make_circles(n_samples=40,
                               shuffle=True,
                               [redacted]
                               random_state=None,
                               [redacted])
```

Load datasets

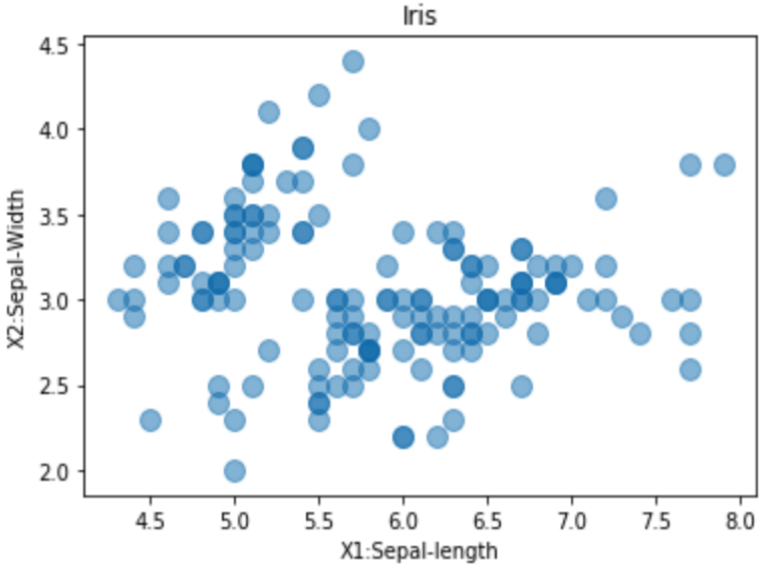
load_boston(*[, return_X_y])	DEPRECATED: load_boston is deprecated in 1.0 and will be removed in 1.2.
load_iris(*[, return_X_y, as_frame])	Load and return the iris dataset (classification).
load_diabetes(*[, return_X_y, as_frame])	Load and return the diabetes dataset (regression).
load_digits(*[, n_class, return_X_y, as_frame])	Load and return the digits dataset (classification).
load_linnerud(*[, return_X_y, as_frame])	Load and return the physical exercise Linnerud dataset.
load_wine(*[, return_X_y, as_frame])	Load and return the wine dataset (classification).
load_breast_cancer(*[, return_X_y, as_frame])	Load and return the breast cancer wisconsin dataset (classification).

https://scikit-learn.org/stable/datasets/toy_dataset.html

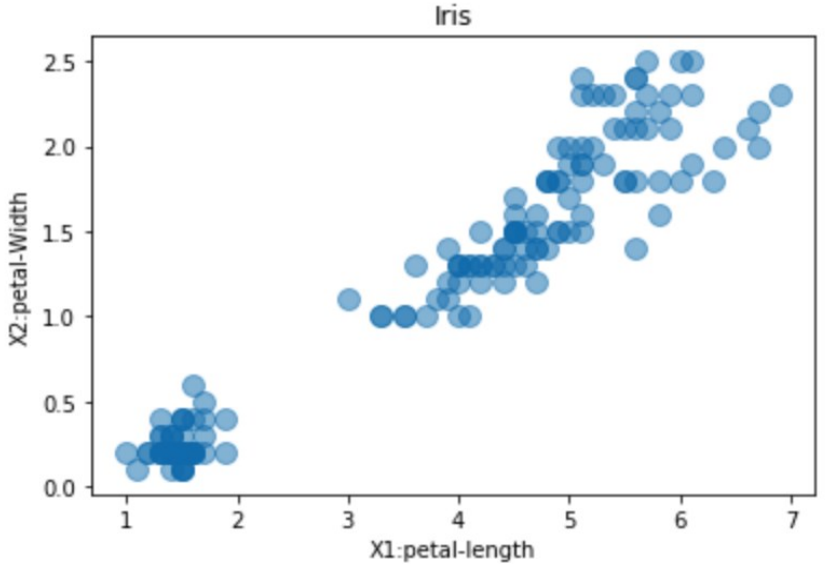
Load_iris

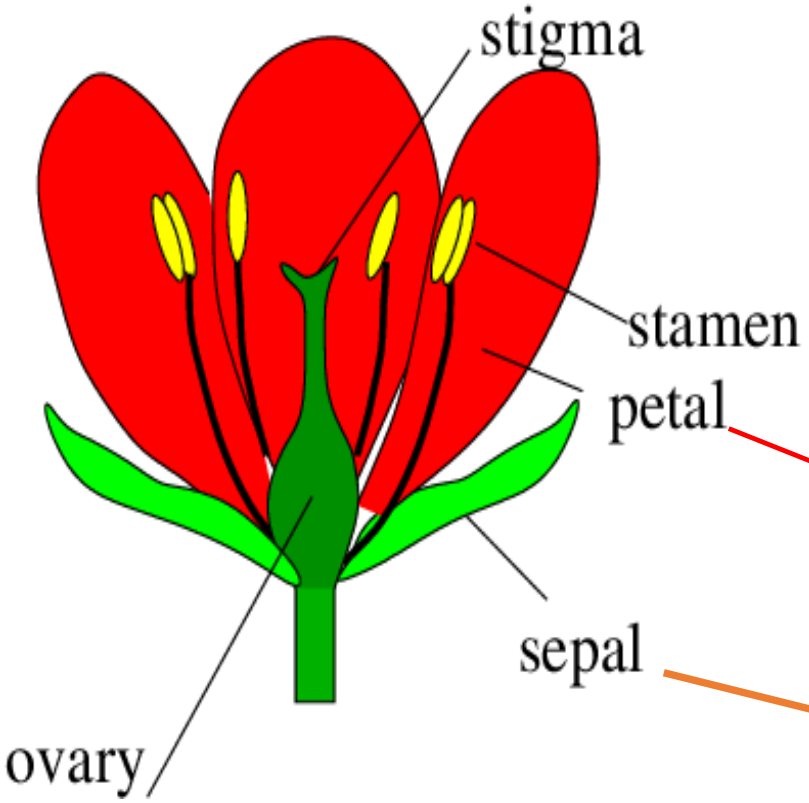
```
# load benchmark datasets
data = datasets.load_iris()
X_data = data.data
y_iris = data.target
```

Sepal width/height



petal width/height





Iris-Setosa

Iris-Versicolour

Iris-Virginica

Load diabetes

```
# load benchmark datasets
diabetesdata = datasets.load_diabetes()
X_diabetes = diabetesdata.data
y_diabetes = diabetesdata.target

print('Features: ', diabetesdata.feature_names)
```

Fetch datasets

- Scikit learn provides methods to access external sources and get data such as

datasets.fetch_20newsgroups(*[, data_home, ...])	Load the filenames and data from the 20 newsgroups dataset (classification).
datasets.fetch_20newsgroups_vectorized(*[, ...])	Load and vectorize the 20 newsgroups dataset (classification).
datasets.fetch_california_housing(*[, ...])	Load the California housing dataset (regression).
datasets.fetch_covtype(*[, data_home, ...])	Load the covertype dataset (classification).
datasets.fetch_kddcup99(*[, subset, ...])	Load the kddcup99 dataset (classification).
datasets.fetch_lfw_pairs(*[, subset, ...])	Load the Labeled Faces in the Wild (LFW) pairs dataset (classification).
datasets.fetch_lfw_people(*[, data_home, ...])	Load the Labeled Faces in the Wild (LFW) people dataset (classification).
datasets.fetch_olivetti_faces(*[, ...])	Load the Olivetti faces data-set from AT&T (classification).
datasets.fetch_openml([name, version, ...])	Fetch dataset from openml by name or dataset id.
datasets.fetch_rcv1(*[, data_home, subset, ...])	Load the RCV1 multilabel dataset (classification).
datasets.fetch_species_distributions(*[, ...])	Loader for species distribution dataset from Phillips et.

- Example

```
1 from sklearn.datasets import fetch_20newsgroups
2
3 data  = fetch_20newsgroups()
4 data.keys()
```

```
dict_keys(['data', 'filenames', 'target_names', 'target', 'DESCR'])
```


< / >
® Creative Commons licenses