

Introduction to Scikit-Learn

- Scikit-Learn, one of the best known Python package for machine learning algorithms that provide solid and efficient implementations of a range of machine learning algorithms.
- It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.
- This library is built upon NumPy, SciPy and Matplotlib.
- The names "Scikit-Learn" and "Sklearn" both are same, with the difference that "scikit-learn" is the official name of the package and is mostly used only for installing the package, whereas "sklearn" is the abbreviated name which is used when we have to use it for python programming.

Installing Scikit-learn

- Using pip
 - `pip install scikit-learn`
- Using conda
 - `conda install scikit-learn`

Scikit-learn Version

In [8]:

```
from sklearn import __version__  
  
print(__version__)
```

0.24.2

Dataset

- A collection of data is called dataset. It is having the following two components: features and response.
- **Features:** The variables of data are called its features. They are also known as predictors, inputs or attributes.
 - **Feature matrix:** It is the collection of features. By convention, this features matrix is often stored in a variable named X.
 - **Feature Names:** It is the list of all the names of the features.
- **Response:** It is the output variable that basically depends upon the feature variables. They are also known as **target**, **label** or **output**.
 - **Response/Target Vector:** It is used to represent response column. Generally, we have just one response column.
 - **Target Names:** It represent the possible values taken by a response vector.

Built-in Dataset

- Scikit-learn have few built-in datasets like "iris" and "digits", "boston", or "diabetes" .

In [26]:

```
# Let's load iris dataset  
  
# import the dataset module  
from sklearn import datasets  
  
# load the iris dataset  
iris = datasets.load_iris()  
iris
```

Out[26]:

```
{'data': array([[5.1, 3.5, 1.4, 0.2],  
                [4.9, 3. , 1.4, 0.2],  
                [4.7, 3.2, 1.3, 0.2],  
                [4.6, 3.1, 1.5, 0.2],  
                [5. , 3.6, 1.4, 0.2],  
                [5.4, 3.9, 1.7, 0.4],  
                [4.6, 3.4, 1.4, 0.3],
```

[5. , 3.4, 1.5, 0.2],
[4.4, 2.9, 1.4, 0.2],
[4.9, 3.1, 1.5, 0.1],
[5.4, 3.7, 1.5, 0.2],
[4.8, 3.4, 1.6, 0.2],
[4.8, 3. , 1.4, 0.1],
[4.3, 3. , 1.1, 0.1],
[5.8, 4. , 1.2, 0.2],
[5.7, 4.4, 1.5, 0.4],
[5.4, 3.9, 1.3, 0.4],
[5.1, 3.5, 1.4, 0.3],
[5.7, 3.8, 1.7, 0.3],
[5.1, 3.8, 1.5, 0.3],
[5.4, 3.4, 1.7, 0.2],
[5.1, 3.7, 1.5, 0.4],
[4.6, 3.6, 1. , 0.2],
[5.1, 3.3, 1.7, 0.5],
[4.8, 3.4, 1.9, 0.2],
[5. , 3. , 1.6, 0.2],
[5. , 3.4, 1.6, 0.4],
[5.2, 3.5, 1.5, 0.2],
[5.2, 3.4, 1.4, 0.2],
[4.7, 3.2, 1.6, 0.2],
[4.8, 3.1, 1.6, 0.2],
[5.4, 3.4, 1.5, 0.4],
[5.2, 4.1, 1.5, 0.1],
[5.5, 4.2, 1.4, 0.2],
[4.9, 3.1, 1.5, 0.2],
[5. , 3.2, 1.2, 0.2],
[5.5, 3.5, 1.3, 0.2],
[4.9, 3.6, 1.4, 0.1],
[4.4, 3. , 1.3, 0.2],
[5.1, 3.4, 1.5, 0.2],
[5. , 3.5, 1.3, 0.3],
[4.5, 2.3, 1.3, 0.3],
[4.4, 3.2, 1.3, 0.2],
[5. , 3.5, 1.6, 0.6],
[5.1, 3.8, 1.9, 0.4],
[4.8, 3. , 1.4, 0.3],
[5.1, 3.8, 1.6, 0.2],
[4.6, 3.2, 1.4, 0.2],

[5.3, 3.7, 1.5, 0.2],
[5. , 3.3, 1.4, 0.2],
[7. , 3.2, 4.7, 1.4],
[6.4, 3.2, 4.5, 1.5],
[6.9, 3.1, 4.9, 1.5],
[5.5, 2.3, 4. , 1.3],
[6.5, 2.8, 4.6, 1.5],
[5.7, 2.8, 4.5, 1.3],
[6.3, 3.3, 4.7, 1.6],
[4.9, 2.4, 3.3, 1.],
[6.6, 2.9, 4.6, 1.3],
[5.2, 2.7, 3.9, 1.4],
[5. , 2. , 3.5, 1.],
[5.9, 3. , 4.2, 1.5],
[6. , 2.2, 4. , 1.],
[6.1, 2.9, 4.7, 1.4],
[5.6, 2.9, 3.6, 1.3],
[6.7, 3.1, 4.4, 1.4],
[5.6, 3. , 4.5, 1.5],
[5.8, 2.7, 4.1, 1.],
[6.2, 2.2, 4.5, 1.5],
[5.6, 2.5, 3.9, 1.1],
[5.9, 3.2, 4.8, 1.8],
[6.1, 2.8, 4. , 1.3],
[6.3, 2.5, 4.9, 1.5],
[6.1, 2.8, 4.7, 1.2],
[6.4, 2.9, 4.3, 1.3],
[6.6, 3. , 4.4, 1.4],
[6.8, 2.8, 4.8, 1.4],
[6.7, 3. , 5. , 1.7],
[6. , 2.9, 4.5, 1.5],
[5.7, 2.6, 3.5, 1.],
[5.5, 2.4, 3.8, 1.1],
[5.5, 2.4, 3.7, 1.],
[5.8, 2.7, 3.9, 1.2],
[6. , 2.7, 5.1, 1.6],
[5.4, 3. , 4.5, 1.5],
[6. , 3.4, 4.5, 1.6],
[6.7, 3.1, 4.7, 1.5],
[6.3, 2.3, 4.4, 1.3],
[5.6, 3. , 4.1, 1.3],

[5.5, 2.5, 4. , 1.3],
[5.5, 2.6, 4.4, 1.2],
[6.1, 3. , 4.6, 1.4],
[5.8, 2.6, 4. , 1.2],
[5. , 2.3, 3.3, 1.],
[5.6, 2.7, 4.2, 1.3],
[5.7, 3. , 4.2, 1.2],
[5.7, 2.9, 4.2, 1.3],
[6.2, 2.9, 4.3, 1.3],
[5.1, 2.5, 3. , 1.1],
[5.7, 2.8, 4.1, 1.3],
[6.3, 3.3, 6. , 2.5],
[5.8, 2.7, 5.1, 1.9],
[7.1, 3. , 5.9, 2.1],
[6.3, 2.9, 5.6, 1.8],
[6.5, 3. , 5.8, 2.2],
[7.6, 3. , 6.6, 2.1],
[4.9, 2.5, 4.5, 1.7],
[7.3, 2.9, 6.3, 1.8],
[6.7, 2.5, 5.8, 1.8],
[7.2, 3.6, 6.1, 2.5],
[6.5, 3.2, 5.1, 2.],
[6.4, 2.7, 5.3, 1.9],
[6.8, 3. , 5.5, 2.1],
[5.7, 2.5, 5. , 2.],
[5.8, 2.8, 5.1, 2.4],
[6.4, 3.2, 5.3, 2.3],
[6.5, 3. , 5.5, 1.8],
[7.7, 3.8, 6.7, 2.2],
[7.7, 2.6, 6.9, 2.3],
[6. , 2.2, 5. , 1.5],
[6.9, 3.2, 5.7, 2.3],
[5.6, 2.8, 4.9, 2.],
[7.7, 2.8, 6.7, 2.],
[6.3, 2.7, 4.9, 1.8],
[6.7, 3.3, 5.7, 2.1],
[7.2, 3.2, 6. , 1.8],
[6.2, 2.8, 4.8, 1.8],
[6.1, 3. , 4.9, 1.8],
[6.4, 2.8, 5.6, 2.1],
[7.2, 3. , 5.8, 1.6],

```
[7.4, 2.8, 6.1, 1.9],  
[7.9, 3.8, 6.4, 2. ],  
[6.4, 2.8, 5.6, 2.2],  
[6.3, 2.8, 5.1, 1.5],  
[6.1, 2.6, 5.6, 1.4],  
[7.7, 3. , 6.1, 2.3],  
[6.3, 3.4, 5.6, 2.4],  
[6.4, 3.1, 5.5, 1.8],  
[6. , 3. , 4.8, 1.8],  
[6.9, 3.1, 5.4, 2.1],  
[6.7, 3.1, 5.6, 2.4],  
[6.9, 3.1, 5.1, 2.3],  
[5.8, 2.7, 5.1, 1.9],  
[6.8, 3.2, 5.9, 2.3],  
[6.7, 3.3, 5.7, 2.5],  
[6.7, 3. , 5.2, 2.3],  
[6.3, 2.5, 5. , 1.9],  
[6.5, 3. , 5.2, 2. ],  
[6.2, 3.4, 5.4, 2.3],  
[5.9, 3. , 5.1, 1.8]]),  
  
'target': array([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, 2, 2, 2, 2, 2, 2, 2, 2, 2,  
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,  
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]),  
  
'frame': None,  
  
'target_names': array(['setosa', 'versicolor', 'virginica'], dtype='<U10'),  
  
'DESCR': '.. _iris_dataset:\n\nIris plants dataset\n-----\n\n**Data Set Characteristics:**\n\nNumber of Instances: 150 (50 in each of three classes)\n    :Number of Attributes: 4 numeric, predictive attribut  
es and the class\n    :Attribute Information:\n        - sepal length in cm\n        - sepal width in cm\n- petal length in cm\n    - petal width in cm\n    - class:\n        - Iris-Setosa\n- Iris-Versicolour\n    - Iris-Virginica\n\nSummary Statistics:\n\n=====  
Min Max Mean SD Class Correlation\nsepal length: 4.3 7.9 5.84 0.83 0.7826\nsepal width: 2.0 4.4 3.05 0.43 -0.4194\npetal length: 1.0 6.9 3.76 1.76 0.9490 (high!)\npetal width: 0.1 2.5 1.20 0.76 0.9565 (high!)  
Missing Attribute Values: None\nClass Distribution: 33.3% for each of 3 classes.  
Creator: R.A. Fisher\nDonor: Michael Marshall (MARSHALL@PLU@io.arc.nasa.gov)\nDate: July, 1988\nThe famous Iris database, first used by Sir R.A. Fisher. The dataset is taken from Fisher's paper. Note that it's th
```

e same as in R, but not as in the UCI Machine Learning Repository, which has two wrong data points. This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. (See Duda & Hart, for example.) The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

.. topic:: References

- Fisher, R. A. "The use of multiple measurements in taxonomic problems" Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to Mathematical Statistics" (John Wiley, NY, 1950).
- Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Analysis. (Q327.D83) John Wiley & Sons. ISBN 0-471-22361-1. See page 218.
- Dasarthy, B.V. (1980) "Nosing Around the Neighborhood: A New System Structure and Classification Rule for Recognition in Partially Exposed Environments". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 1, 67-71.
- Gates, G.W. (1972) "The Reduced Nearest Neighbor Rule". IEEE Transactions on Information Theory, May 1972, 431-433.
- See also: 1988 MLC Proceedings, 54-64. Cheeseman et al's AUTOCLASS II conceptual clustering system finds 3 classes in the data.
- Many, many more ...

```
'feature_names': ['sepal length (cm)',
'sepal width (cm)',
'petal length (cm)',
'petal width (cm)'],
'filename': '/Users/akinulislamjony/.local/share/virtualenvs/p4ds-notebooks--5YdjQw8/lib/python3.8/site-packages/sklearn/datasets/data/iris.csv'}
```

```
In [14]: print("iris dataset description: \n{}".format(iris.DESCR))
```

```
iris dataset description:
.. _iris_dataset:
```

```
Iris plants dataset
-----
```

```
**Data Set Characteristics:**
```

```
:Number of Instances: 150 (50 in each of three classes)
:Number of Attributes: 4 numeric, predictive attributes and the class
:Attribute Information:
  - sepal length in cm
  - sepal width in cm
  - petal length in cm
  - petal width in cm
  - class:
    - Iris-Setosa
    - Iris-Versicolour
    - Iris-Virginica
```

:Summary Statistics:

	Min	Max	Mean	SD	Class Correlation
sepal length:	4.3	7.9	5.84	0.83	0.7826
sepal width:	2.0	4.4	3.05	0.43	-0.4194
petal length:	1.0	6.9	3.76	1.76	0.9490 (high!)
petal width:	0.1	2.5	1.20	0.76	0.9565 (high!)

:Missing Attribute Values: None

:Class Distribution: 33.3% for each of 3 classes.

:Creator: R.A. Fisher

:Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)

:Date: July, 1988

The famous Iris database, first used by Sir R.A. Fisher. The dataset is taken from Fisher's paper. Note that it's the same as in R, but not as in the UCI Machine Learning Repository, which has two wrong data points.

This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. (See Duda & Hart, for example.) The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

.. topic:: References

- Fisher, R.A. "The use of multiple measurements in taxonomic problems" Annual Eugenics, 7, Part II, 179-188 (1936); also in "Contributions to Mathematical Statistics" (John Wiley, NY, 1950).
- Duda, R.O., & Hart, P.E. (1973) Pattern Classification and Scene Analysis. (Q327.D83) John Wiley & Sons. ISBN 0-471-22361-1. See page 218.
- Dasarathy, B.V. (1980) "Nosing Around the Neighborhood: A New System Structure and Classification Rule for Recognition in Partially Exposed Environments". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No. 1, 67-71.
- Gates, G.W. (1972) "The Reduced Nearest Neighbor Rule". IEEE Transactions

- on Information Theory, May 1972, 431-433.
- See also: 1988 MLC Proceedings, 54-64. Cheeseman et al's AUTOCLASS II conceptual clustering system finds 3 classes in the data.
- Many, many more ...

In [35]:

```
print("iris data size is {}".format(iris.data.shape))
print("iris target size is {}".format(iris.target.shape))
print("iris data has {} features".format(iris.data.shape[1]))
print("the feature names: \n\t{}".format(iris.feature_names))
print("iris data has {} samples".format(iris.data.shape[0]))
print("the target label names: \n\t{}".format(iris.target_names))
```

```
iris data size is (150, 4)
iris target size is (150,)
iris data has 4 features
the feature names:
    ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
iris data has 150 samples
the target label names:
    ['setosa' 'versicolor' 'virginica']
```

In [42]:

```
X = iris.data    # Feature matrix
y = iris.target  # Target Vector

print("X datatype: {}".format(X.dtype))
print("X dimensions: {}".format(X.ndim))
print("\nFirst 10 rows of X:\n", X[:10])

print("\ny datatype: {}".format(y.dtype))
print("y dimensions: {}".format(y.ndim))
print("\nFirst 10 rows of Y:\n", y[:10])
```

```
X datatype: float64
X dimensions: 2
```

First 10 rows of X:

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]]
```

```
y datatype: int64
y dimensions: 1
```

First 10 rows of Y:

```
[0 0 0 0 0 0 0 0 0 0]
```

Splitting Dataset

- To check the accuracy of our model
- Split the dataset into two pieces: training set and testing set.
- Training set to train the model
- Testing set to test the model

In [48]:

```
# For example, iris dataset splitted into 70:30 ratio

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 1)
# test_size = 0.3 means 150 * 0.3 = 45 rows
# random_state = 1 guarantte hat the split will always be the same (i.e. reproducible results)

print("X_train shape: ", X_train.shape)
print("X_test shape: ", X_test.shape)

print("y_train shape: ", y_train.shape)
print("y_test shape: ", y_test.shape)
```

```
X_train shape: (105, 4)
X_test shape: (45, 4)
y_train shape: (105,)
y_test shape: (45,)
```
