

Introduction à Python, scikit-learn et au premier projet de Machine Learning

Adrien Bibal

Université de Namur

adrien.bibal@unamur.be

14 octobre 2019

Contenu de la présentation

Introduction à Python

- Remarques

- Syntaxe : simple script

- Syntaxe : fonctions

- Syntaxe : branchements conditionnels

- Syntaxe : boucles while

- Syntaxe : boucles for

- Syntaxe : import

Introduction à scikit-learn

- Description de scikit-learn

- Site web de scikit-learn

- Installation de scikit-learn

- Recherches sur scikit-learn

Premier projet

- Description du projet

Remarques

Quelques commentaires sur Python :

Remarques

Quelques commentaires sur Python :

- ▶ Deux versions utilisées : Python 2 et Python 3

Remarques

Quelques commentaires sur Python :

- ▶ Deux versions utilisées : Python 2 et Python 3
- ▶ Typage dynamique

Remarques

Quelques commentaires sur Python :

- ▶ Deux versions utilisées : Python 2 et Python 3
- ▶ Typage dynamique
- ▶ Scope défini par l'indentation

Remarques

Quelques commentaires sur Python :

- ▶ Deux versions utilisées : Python 2 et Python 3
- ▶ Typage dynamique
- ▶ Scope défini par l'indentation
- ▶ Possibilité de retourner plusieurs valeurs

Remarques

Quelques commentaires sur Python :

- ▶ Deux versions utilisées : Python 2 et Python 3
- ▶ Typage dynamique
- ▶ Scope défini par l'indentation
- ▶ Possibilité de retourner plusieurs valeurs
- ▶ Possibilité d'utiliser l'OOP

Remarques

Quelques commentaires sur Python :

- ▶ Deux versions utilisées : Python 2 et Python 3
- ▶ Typage dynamique
- ▶ Scope défini par l'indentation
- ▶ Possibilité de retourner plusieurs valeurs
- ▶ Possibilité d'utiliser l'OOP
- ▶ La doc sur <https://docs.python.org>

Syntaxe : simple script

```
a = 3  
b = 4  
c = a + b  
print(c)
```

Syntaxe : fonctions

```
def my_super_function(a, b):  
    c = a + b  
    return c
```

Syntaxe : branchements conditionnels

```
def my_super_function(a, b):  
    c = a + b  
    if c > 10:  
        c = c / 10.0  
    else:  
        c *= 10  
    return c
```

Syntaxe : boucles while

```
def my_super_function(a, b):  
    c = a + b  
    if c > 10:  
        while c mod 3 != 0:  
            c += 1  
    else:  
        c *= 10  
    return c
```

Syntaxe : boucles for

```
def my_super_function(a, b):  
    c = a + b  
    if c > 10:  
        for x in range(3, 7):  
            c += x  
    else:  
        c *= 10  
    return c
```

Syntaxe : import

```
import numpy as np
```

```
x = [1, 2, 3]
```

```
z = np.array(x)
```

Syntaxe : import

```
from numpy import array
```

```
x = [1, 2, 3]
```

```
z = array(x)
```


Description de scikit-learn

"scikit-learn is a Python module for machine learning built on top of SciPy and distributed under the 3-Clause BSD license.

The project was started in 2007 by David Cournapeau as a Google Summer of Code project, and since then many volunteers have contributed. See the AUTHORS.rst file for a complete list of contributors.

It is currently maintained by a team of volunteers."

Source : <https://github.com/scikit-learn/scikit-learn>

Site web de scikit-learn



scikit-learn

Home Installation Documentation Examples

Google Custom Search Search x

scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable -BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition, ...

Algorithms: *SVM, nearest neighbors, random forest, ...* — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: *SVR, ridge regression, Lasso, ...* — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: *k-Means, spectral clustering, mean-shift, ...* — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: *PCA, feature selection, non-negative matrix factorization.* — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: *grid search, cross validation, metrics.* — Examples

Preprocessing


Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: *preprocessing, feature extraction.* — Examples

<http://scikit-learn.org>

Installation de scikit-learn



[Home](#) [Installation](#) [Documentation](#) [Examples](#)

[Previous scikit-learn](#)

[Up scikit-learn](#)

If you use the software, please consider [citing scikit-learn](#).

Installing scikit-learn

Installing the latest release

Third-party Distributions

- Canopy and Anaconda for all supported platforms
- Python(x,y) for Windows

Installing scikit-learn

Note: If you wish to contribute to the project, it's recommended you [install the latest development version](#).

Installing the latest release

Scikit-learn requires:

- Python (≥ 2.6 or ≥ 3.3),
- NumPy ($\geq 1.6.1$),
- SciPy (≥ 0.9).

If you already have a working installation of numpy and scipy, the easiest way to install scikit-learn is using pip

```
pip install -U scikit-learn
```

or conda:

```
conda install scikit-learn
```

We don't recommend installing scipy or numpy using pip on linux, as this will involve a lengthy build-process with many dependencies. Without careful configuration, building numpy yourself can lead to an installation that is much slower than it should be. If you are using Linux, consider using your package manager to install scikit-learn. It is usually the easiest way, but might not provide the newest version. If you haven't already installed numpy and scipy and can't install them via your operation system, it is recommended to use a third party distribution.

Third-party Distributions

If you don't already have a python installation with numpy and scipy, we recommend to install either via your package manager or via a python bundle. These come with numpy, scipy, scikit-learn, matplotlib and many other helpful scientific and data processing libraries.

Available options are:

Canopy and Anaconda for all supported platforms

<http://scikit-learn.org/stable/install.html>

Recherches sur scikit-learn



[Home](#) [Installation](#) [Documentation](#) [Examples](#)

decision trees

Search ✕

About 109 results (0.20 seconds)



[1.10. Decision Trees — scikit-learn 0.17 documentation](#)

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the ...
[scikit-learn.org](#)



[sklearn.tree.DecisionTreeClassifier — scikit-learn 0.17 documentation](#)

For the default settings of a **decision tree** on large datasets, setting this to true may slow down the training process. When using either a smaller dataset or a ...
[scikit-learn.org](#)



[Decision Tree Regression with AdaBoost — scikit-learn 0.17 ...](#)

A **decision tree** is boosted using the AdaBoost.R2 [1] algorithm on a 1D sinusoidal dataset with a small amount of Gaussian noise. 299 boosts (300 decision ...
[scikit-learn.org](#)



[Decision Tree Regression — scikit-learn 0.17 documentation](#)

We can see that if the maximum depth of the tree (controlled by the `max_depth` parameter) is set too high, the **decision trees** learn too fine details of the training ...
[scikit-learn.org](#)



[Multi-class AdaBoosted Decision Trees — scikit-learn 0.17 ...](#)

Multi-class AdaBoosted **Decision Trees**. This example reproduces Figure 1 of Zhu et al [1] and shows how boosting can improve prediction accuracy on a ...
[scikit-learn.org](#)



[sklearn.tree.DecisionTreeRegressor — scikit-learn 0.17 ...](#)

For the default settings of a **decision tree** on large datasets, setting this to true may slow down the training process. When using either a smaller dataset or a ...
[scikit-learn.org](#)



[sklearn.tree.export_graphviz — scikit-learn 0.17 documentation](#)

This function generates a GraphViz representation of the **decision tree**, which is then written into `out_file`. Once exported, graphical renderings can be generated ...
[scikit-learn.org](#)



[Multi-output Decision Tree Regression — scikit-learn 0.17 ...](#)

We can see that if the maximum depth of the tree (controlled by the `max_depth` parameter) is set too high, the **decision trees** learn too fine details of the training ...
[scikit-learn.org](#)



[1.11. Ensemble methods — scikit-learn 0.17 documentation](#)

These methods are used as a way to reduce the variance of a base estimator (e.g., a **decision tree**), by introducing randomization into its construction procedure ...
[scikit-learn.org](#)



[Plot the decision surfaces of ensembles of trees on the iris dataset ...](#)

This plot compares the decision surfaces learned by a **decision tree** classifier (first column), by a random forest classifier

Description du projet premier projet

Prédire si une personne gagne plus ou moins de 50 000\$/an

Description du projet premier projet

Prédire si une personne gagne plus ou moins de 50 000\$/an

Utilisation d'arbres de décision

Description du projet premier projet

Prédire si une personne gagne plus ou moins de 50 000\$/an

Utilisation d'arbres de décision

Étude des arbres à travers leur précision ET leur complexité

Description du projet premier projet

Prédire si une personne gagne plus ou moins de 50 000\$/an

Utilisation d'arbres de décision

Étude des arbres à travers leur précision ET leur complexité

Utilisation de Python avec scikit-learn (le code doit être rendu !)

Description du projet premier projet

Prédire si une personne gagne plus ou moins de 50 000\$/an

Utilisation d'arbres de décision

Étude des arbres à travers leur précision ET leur complexité

Utilisation de Python avec scikit-learn (le code doit être rendu !)

Répondez aux questions de l'énoncé

Description du projet premier projet

Prédire si une personne gagne plus ou moins de 50 000\$/an

Utilisation d'arbres de décision

Étude des arbres à travers leur précision ET leur complexité

Utilisation de Python avec scikit-learn (le code doit être rendu !)

Répondez aux questions de l'énoncé

/!\ /!\ /!\ Max 4 pages /!\ /!\ /!\

Toute page supplémentaire ne sera pas lue