MARRAKECH
جامعة القاضي عياض
UNIVERSITÉ CADI AYYAD

كليـــــــة العلوم
السملالية - مراكش
FACULTÉ DES SCIENCES
SEMLALIA - MARRAKECH

scikit learn

Présenté par :
Abderrahman Salhi
Maryame laouina

Supervisé par :
Qaffou Issam

Plan

Introduction

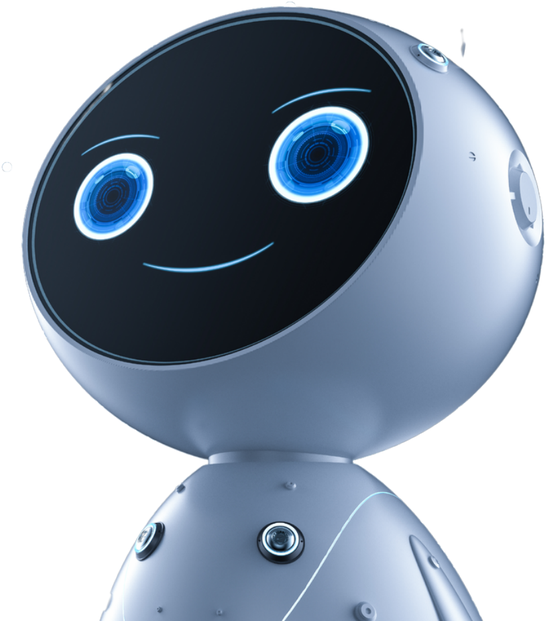
Définition et installation de scikit learn

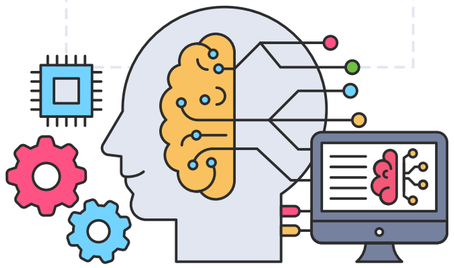
les modules de scikit learn et ses domaines d'utilisation

Comment choisir le meilleur algorithme ?

Démonstration

Conclusion





Machine Learning

Unsupervised Learning

Dimensionality Reduction

Clustering

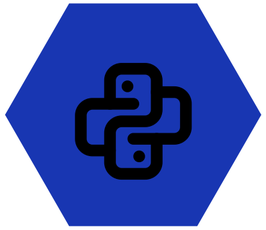
Reinforcement Learning

Supervised Learning

Classification

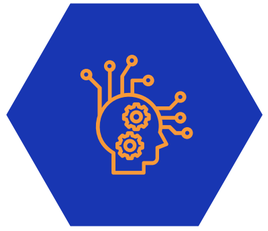
Regression

c'est quoi scikit learn ?



Une bibliothèque
Python open source

Utilisée dans l'apprentissage automatique



Basée sur plusieurs bibliothéques : scipy, numpy, matplotlib

Qui utilise scikit learn ?

Instalation

Packager pip

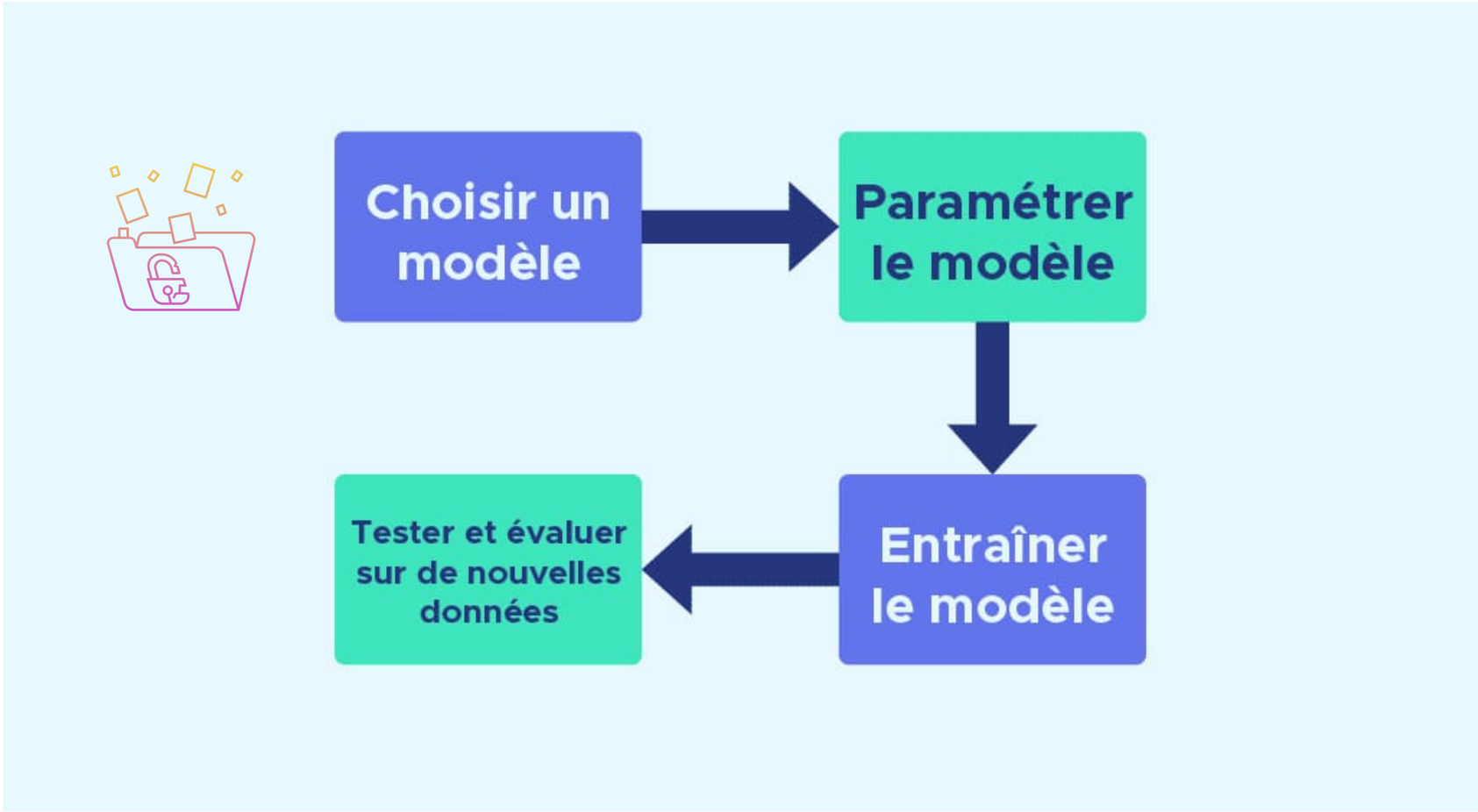
Install the 64bit version of Python 3, for instance from https://www.python.org.
Then run:

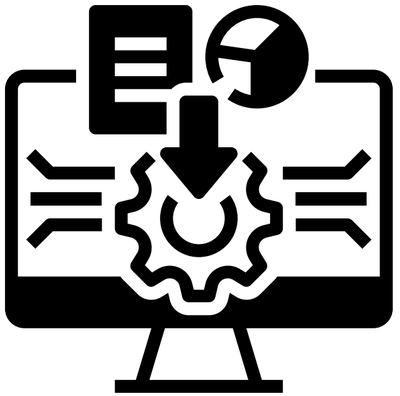
```
$ pip install -U scikit-learn
```

Packager conda

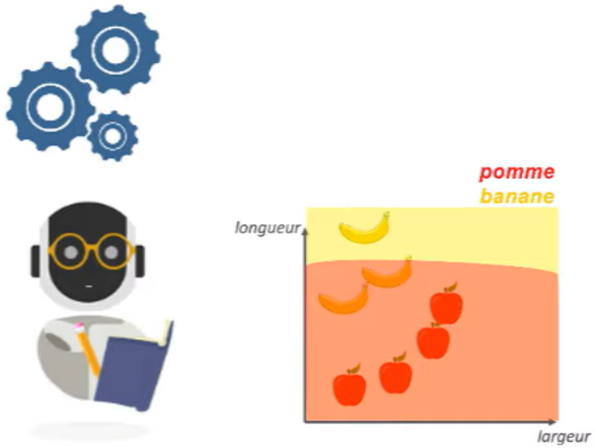
```
$ conda create -n sklearn-env -c conda-forge scikit-learn
$ conda activate sklearn-env
```

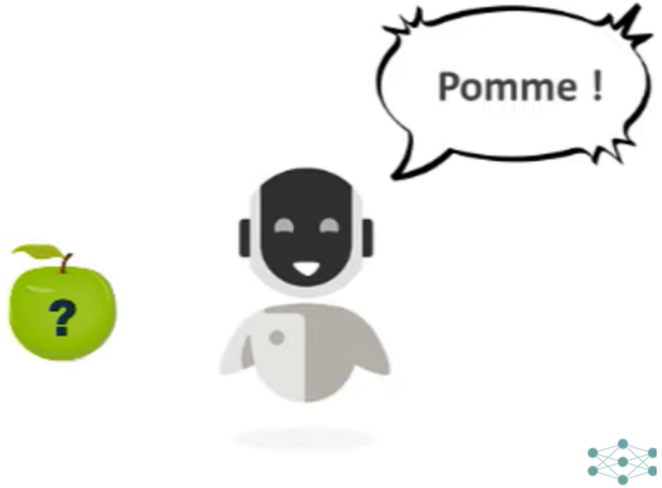
Les bases de scikit learn





Avec trois méthodes présentes dans toutes les classes de scikit learn





Interface de sklearn

Linear Regression

Decision Tree

Random Forest

K-NN

SVM

Neural Network

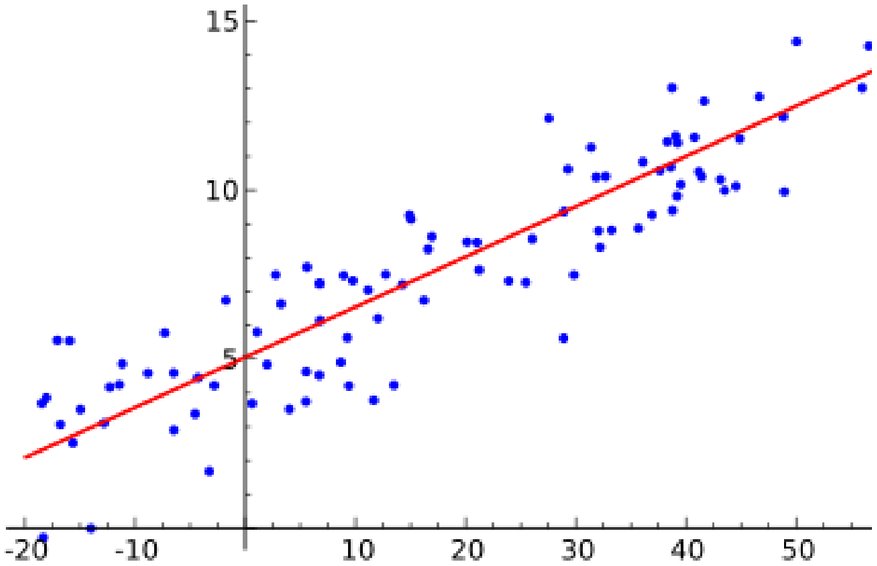
Différents mécanismes

*Mais la **même interface***

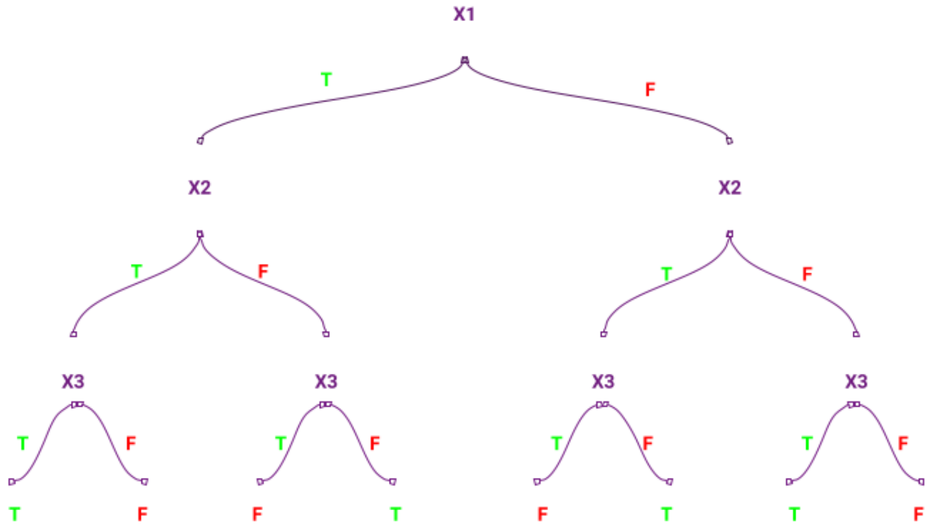
fit

score

predict

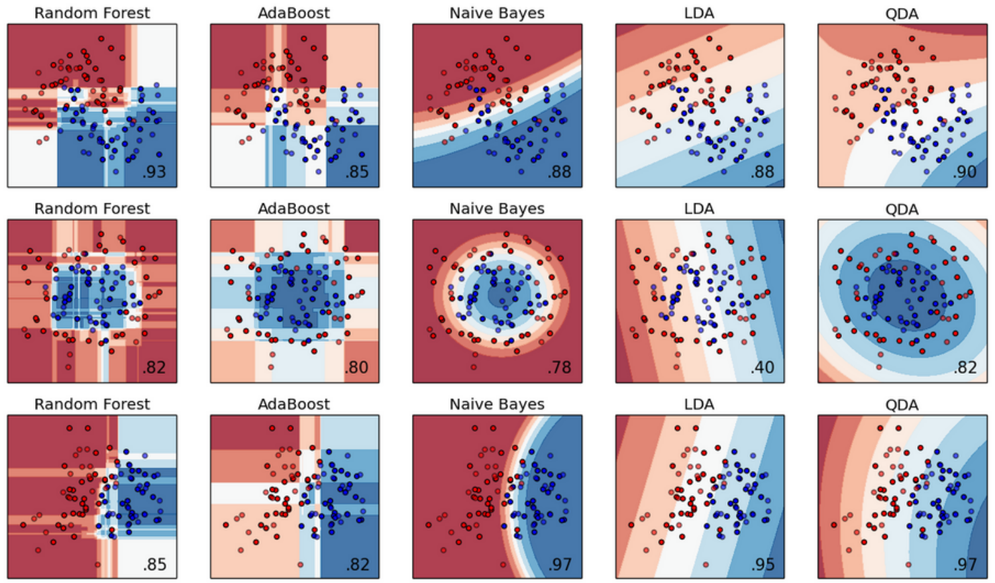



```
model  = LinearRegression()
model.fit(X, y)
model.score(X, y)
model.predict(X)
```




```
model = DecisionTreeClassifier()
model.fit(X, y)
model.score(X, y)
model.predict(X)
```

- Classification
- Dimensionality Reduction
- Regression
- Model selection
- Clustering
- Pre-Processing



Classification

Identifier à quelle catégorie appartient un objet.

Applications :

Spam detection

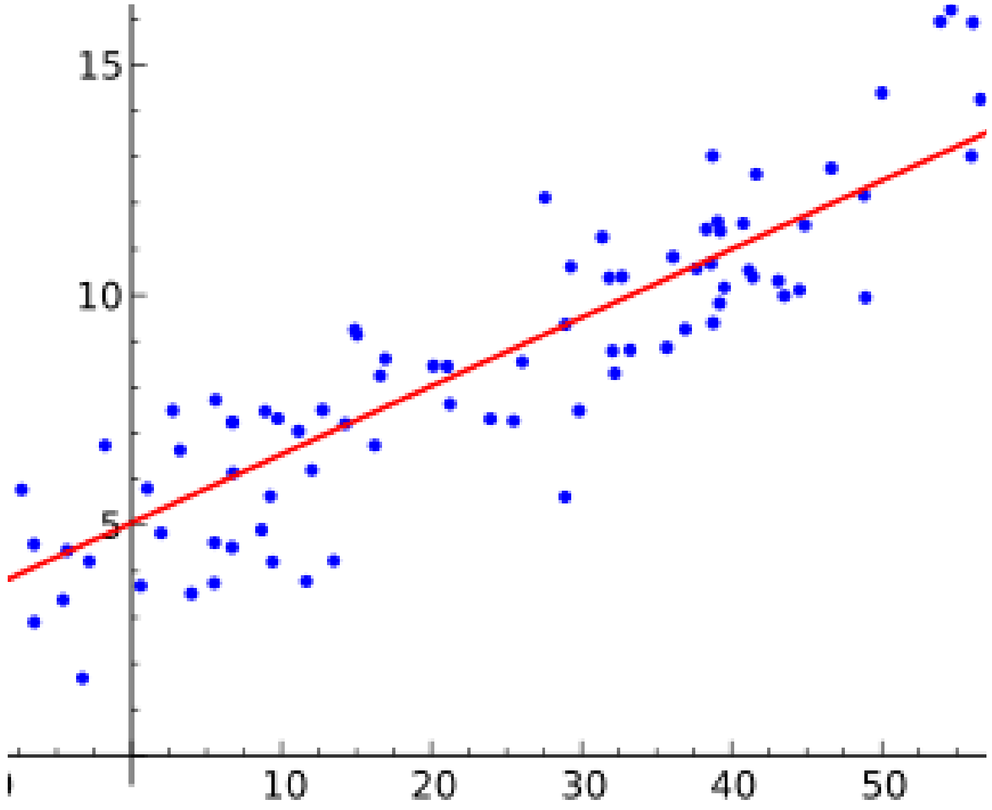
face detection

Algorithmes :

SVM, random forest, Linear Models, Nearest Neighbors,Decision Trees...

EXAMPLE

```
>>> from sklearn import svm
>>> X = [[0, 0], [1, 1]]
>>> y = [0, 1]
>>> clf = svm.SVC()
>>> clf.fit(X, y)
SVC()
```



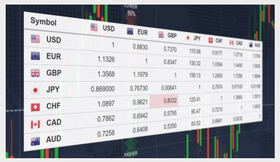
Regression

Prédire un attribut à valeur continue associé à un objet.

Applications :

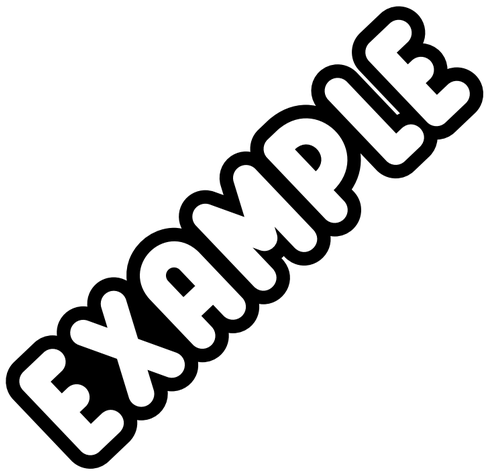
Stock prices

Drug response



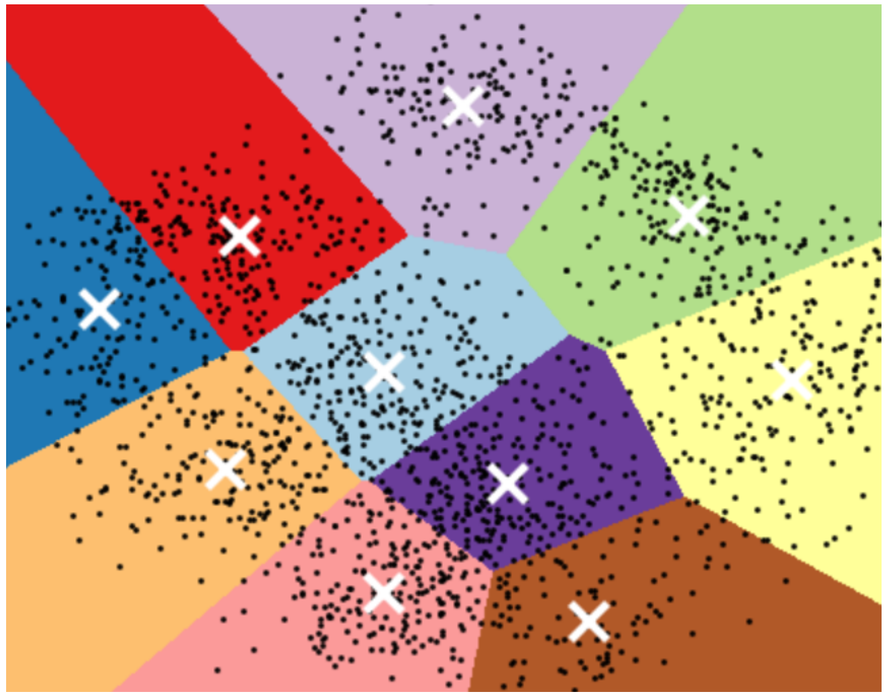
Algorithmes :

SVR, nearest neighbors, random forest, Linear Models




```
>>> from sklearn import linear_model
>>> reg = linear_model.LinearRegression()
>>> reg.fit([[0, 0], [1, 1], [2, 2]], [0, 1, 2])
LinearRegression()
>>> reg.coef_
array([0.5, 0.5])
```

Clustering



Regroupement automatique d'objets similaires en ensembles.

Applications :

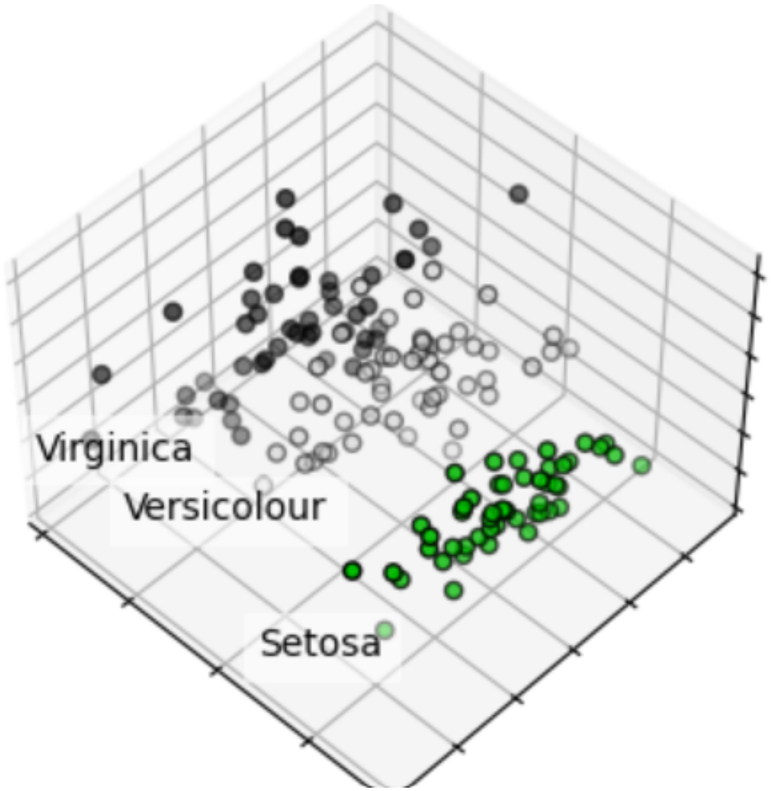
Customer segmentation

Grouping experiment outcomes

Algorithmes :

k-Means, spectral clustering, mean-shift...

Dimensionality reduction

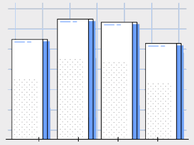


Réduire le nombre de variables aléatoires à considérer

Applications :

Visualization

Increased efficiency



Algorithmes :

PCA, feature selection...

EXAMPLE

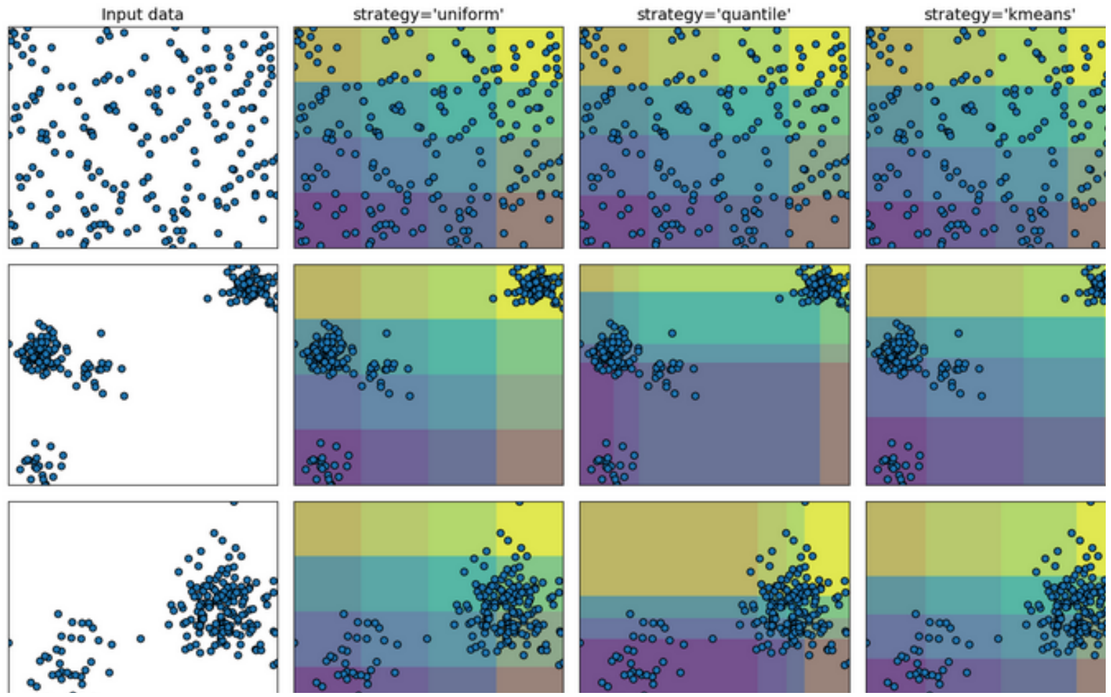
```
[1]  import numpy as np
     from sklearn.decomposition import PCA

     X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
     X.shape
```

```
(6, 2)
```

```
[2]  pca = PCA(n_components=1)
     pca.fit(X)
     X = pca.transform(X)
     X.shape
```

```
(6, 1)
```

Pre-Processing



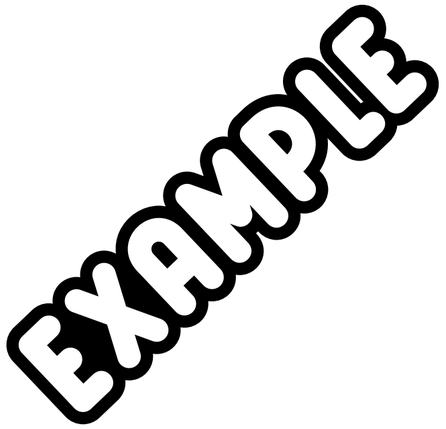
Extraction et normalisation de caractéristiques

Applications :

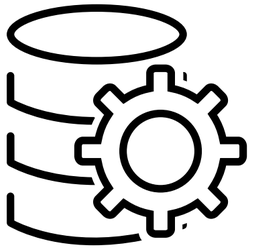
Transformer les données d'entrée telles que le texte pour les utiliser avec des algorithmes d'apprentissage automatique

Algorithmes :

preprocessing, feature extraction,




```
>>> measurements = [
...     {'city': 'Dubai', 'temperature': 33.},
...     {'city': 'London', 'temperature': 12.},
...     {'city': 'San Francisco', 'temperature': 18.},
... ]

>>> from sklearn.feature_extraction import DictVectorizer
>>> vec = DictVectorizer()

>>> vec.fit_transform(measurements).toarray()
array([[ 1.,  0.,  0., 33.],
       [ 0.,  1.,  0., 12.],
       [ 0.,  0.,  1., 18.]])
```



Datasets

scikit-learn est livré avec quelques petits ensembles de données standard qui ne nécessitent pas de télécharger de fichier à partir d'un site Web externe.

Ils peuvent être chargés à l'aide des fonctions suivantes :

`load_boston`(*[, retour_X_y])	DÉCONSEILLÉ : `load_boston` est déprécié dans la version 1.0 et sera supprimé dans la version 1.2.
`load_iris`(*[, return_X_y, as_frame])	Chargez et renvoyez l'ensemble de données d'iris (classification).
`load_diabetes`(*[, return_X_y, as_frame, mis à l'échelle])	Chargez et renvoyez l'ensemble de données sur le diabète (régression).
`load_digits`(*[, n_class, return_X_y, as_frame])	Charger et renvoyer le jeu de données digits (classification).
`load_linnerud`(*[, return_X_y, as_frame])	Chargez et renvoyez l'ensemble de données d'exercice physique Linnerud.
`load_wine`(*[, return_X_y, as_frame])	Chargez et renvoyez le jeu de données de vin (classification).
`load_breast_cancer`(*[, return_X_y, as_frame])	Chargez et renvoyez l'ensemble de données du Wisconsin sur le cancer du sein (classification).

EXAMPLE

```
[3]  # Import Libraries
     from sklearn.datasets import load_iris

     #load iris data
     IrisData = load_iris()
```

```
[10] #X Data
     X = IrisData.data
     print('X shape is ' , X.shape)
     print('X Features are \n' , IrisData.feature_names)
     print('X Data is \n' , X[:3])
```

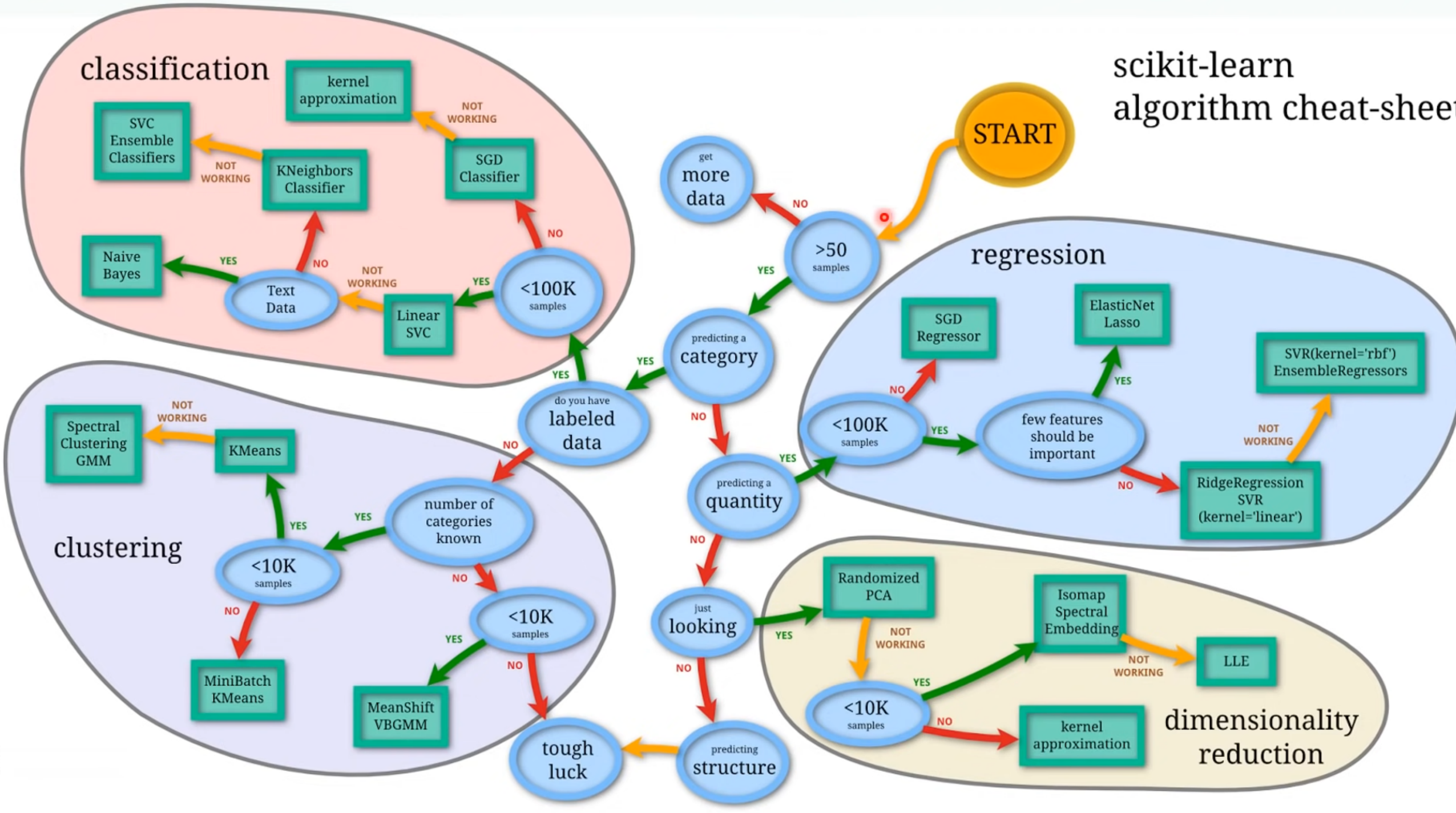
```
X shape is  (150, 4)
X Features are
 ['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
X Data is
 [[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]]
```

```
[16] #y Data
     y = IrisData.target
     print('y shape is ' , y.shape)
     print('y Columns are \n' , IrisData.target_names)
     print('y Data is \n' , y[:122])
```

```
y shape is  (150,)
y Columns are
 ['setosa' 'versicolor' 'virginica']
y Data is
 [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 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2]
```

Ces ensembles de données sont utiles pour illustrer rapidement le comportement des différents algorithmes implémentés dans scikit-learn. Ils sont cependant souvent trop petits pour être représentatifs des tâches d'apprentissage automatique du monde réel.

Comment choisir le meilleur algorithme ?


scikit-learn
algorithm cheat-sheet
START
>50 samples
get more data
NO
YES
predicting a category
predicting a quantity
do you have labeled data
just looking
predicting structure
tough luck
classification
kernel approximation
SVC Ensemble Classifiers
KNeighbors Classifier
SGD Classifier
Naive Bayes
Text Data
Linear SVC
<100K samples
NOT WORKING
regression
SGD Regressor
ElasticNet Lasso
SVR(kernel='rbf') EnsembleRegressors
<100K samples
few features should be important
RidgeRegression SVR (kernel='linear')
clustering
Spectral Clustering GMM
KMeans
number of categories known
<10K samples
MiniBatch KMeans
MeanShift VBGMM
dimensionality reduction
Randomized PCA
Isomap Spectral Embedding
LLE
<10K samples
kernel approximation

Démonstration


Merci pour votre
attention


The
End