Ensembles de données disponibles dans Sklearn

Jeux de données dans Sklearn

`Scikit-learn` met à disposition une multitude de jeux de données pour tester les algorithmes d'apprentissage. Ils sont disponibles en trois versions :

Données emballées : ces petits ensembles de données sont emballés avec l'installation de Scikit-learn et peuvent être téléchargés à l'aide des outils de `sklearn.datasets.load_*` . **Données téléchargeables :** ces ensembles de données plus importants sont disponibles pour le téléchargement, et scikit-learn inclut des outils qui simplifient ce processus. Ces outils se trouvent dans `sklearn.datasets.fetch_*` . **Données générées :** il existe plusieurs jeux de données qui sont générés à partir de modèles basés sur une graine aléatoire. Ils sont disponibles dans `sklearn.datasets.make_` .

Vous pouvez explorer les chargeurs, les récupérateurs et les générateurs de jeux de données disponibles en utilisant la fonctionnalité de complétion de tabulation d'IPython. Après avoir importé le sous-module datasets de sklearn, tapez:

`datasets.load_<TAB>` ou `datasets.fetch_<TAB>` ou `datasets.make_<TAB>`

pour voir la liste des fonctions disponibles.

Structure des données et des étiquettes

Les données dans scikit-learn sont dans la plupart des cas enregistrées sous forme de tableaux `Numpy` bidimensionnels de la forme `(n, m)` . De nombreux algorithmes acceptent également les matrices `scipy.sparse` de la même forme.

- **n :** (n_samples) Le nombre d'échantillons : chaque échantillon est un élément à traiter (par exemple, classifier). Un échantillon peut être un document, une image, un son, une vidéo, un objet astronomique, une ligne dans une base de données ou un fichier CSV, ou tout ce que vous pouvez décrire avec un ensemble fixe de traits quantitatifs.
- **m :** (n_features) Le nombre de caractéristiques ou de traits distincts qui peuvent être utilisés pour décrire chaque élément de manière quantitative. Les caractéristiques sont généralement à valeur réelle, mais peuvent être booléennes ou à valeur discrète dans certains cas.

Entrée [2]:
```
!pip install sklearn
```

```
Requirement already satisfied: sklearn in /Users/francoisalin/opt/
anaconda3/lib/python3.9/site-packages (0.0.post1)
```

Entrée [3]:
```
from sklearn import datasets
```

Attention : beaucoup de ces ensembles de données sont assez volumineux et peuvent prendre beaucoup de temps à télécharger !

Nous allons examiner de plus près l'un de ces ensembles de données. Nous examinons l'ensemble de données sur les chiffres. Nous allons d'abord le charger :

Entrée [4]:
```
from sklearn.datasets import load_digits
digits = load_digits()
```

Là encore, nous pouvons obtenir un aperçu des attributs disponibles en examinant les "clés" :

Entrée [5]:
```
digits.keys()
```

Out[5]:
```
dict_keys(['data', 'target', 'frame', 'feature_names', 'target_nam
es', 'images', 'DESCR'])
```

Jetons un coup d'œil au nombre d'articles et de fonctionnalités :

Entrée [6]:
```
n_samples, n_features = digits.data.shape
print((n_samples, n_features))
```

```
(1797, 64)
```

Entrée [7]:
```
print(digits.data[0])
print(digits.target)
```

```
[ 0.  0.  5. 13.  9.  1.  0.  0.  0.  0. 13. 15. 10. 15.  5.  0.
0.  3.
 15.  2.  0. 11.  8.  0.  0.  4. 12.  0.  0.  8.  8.  0.  0.  5.
8.  0.
  0.  9.  8.  0.  0.  4. 11.  0.  1. 12.  7.  0.  0.  2. 14.  5. 1
0. 12.
  0.  0.  0.  0.  6. 13. 10.  0.  0.  0.]
[0 1 2 ... 8 9 8]
```

Les données sont également disponibles sur `digits.images` . Il s'agit des données brutes des images sous la forme de 8 lignes et 8 colonnes.

Avec `data` une image correspond à un tableau Numpy unidimensionnel avec la longueur 64, et la représentation `images` contient des tableaux `numpy` bidimensionnels avec la forme (8, 8)

Entrée [8]:
```
print("Shape of an item: ", digits.data[0].shape)
print("Data type of an item: ", type(digits.data[0]))
print("Shape of an item: ", digits.images[0].shape)
print("Data tpye of an item: ", type(digits.images[0]))
```

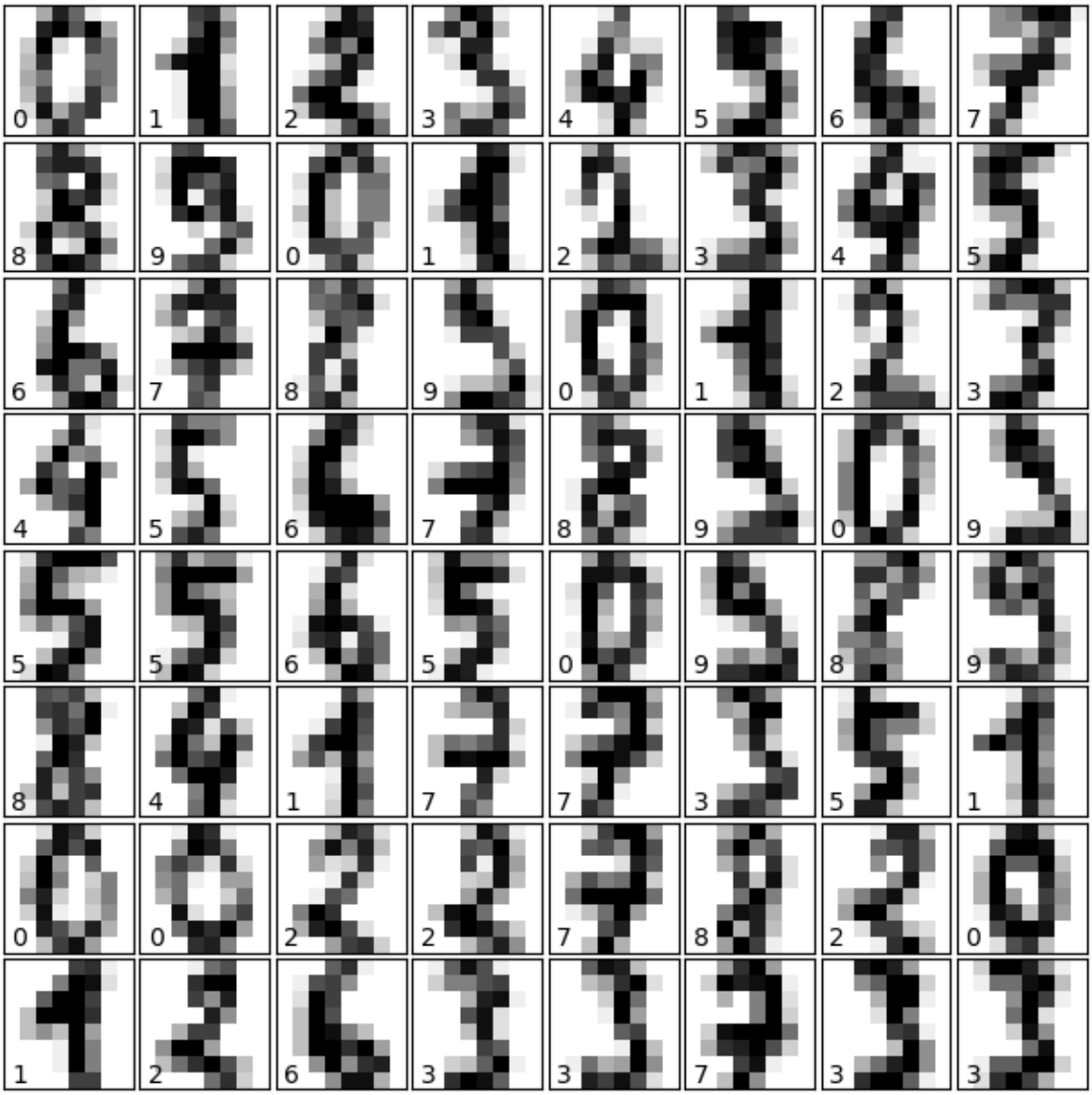
```
Shape of an item:  (64,)
Data type of an item:  <class 'numpy.ndarray'>
Shape of an item:  (8, 8)
Data tpye of an item:  <class 'numpy.ndarray'>
```

Visualisons les données. C'est un peu plus compliqué que le simple nuage de points que nous avons utilisé plus haut, mais nous pouvons le faire assez rapidement.

Entrée [9]:
```
# visualisation of the digits data set
import matplotlib.pyplot as plt
# set up the figure
fig = plt.figure(figsize=(6, 6))  # figure size in inches
fig.subplots_adjust(left=0, right=1, bottom=0, top=1, hspace=0.05, 

# plot the digits: each image is 8x8 pixels
for i in range(64):
    ax = fig.add_subplot(8, 8, i + 1, xticks=[], yticks=[])
    ax.imshow(digits.images[i], cmap=plt.cm.binary, interpolation='

    # label the image with the target value
    ax.text(0, 7, str(digits.target[i]))
```



Exercices

Exercice 1

`sklearn` contient un *ensemble de données sur le vin*.

- Trouvez et chargez cet ensemble de données
- Pouvez-vous trouver une description ?
- Quels sont les noms des classes ?
- Quelles sont les caractéristiques ?
- Où se trouvent les données et les données étiquetées ?

Entrée [31]:
```
from sklearn.datasets import load_wine
data = load_wine()
data.data
```

Out[31]:
```
array([[1.423e+01, 1.710e+00, 2.430e+00, ..., 1.040e+00, 3.920e+00
,
        1.065e+03],
       [1.320e+01, 1.780e+00, 2.140e+00, ..., 1.050e+00, 3.400e+00
,
        1.050e+03],
       [1.316e+01, 2.360e+00, 2.670e+00, ..., 1.030e+00, 3.170e+00
,
        1.185e+03],
       ...,
       [1.327e+01, 4.280e+00, 2.260e+00, ..., 5.900e-01, 1.560e+00
,
        8.350e+02],
       [1.317e+01, 2.590e+00, 2.370e+00, ..., 6.000e-01, 1.620e+00
,
        8.400e+02],
       [1.413e+01, 4.100e+00, 2.740e+00, ..., 6.100e-01, 1.600e+00
,
        5.600e+02]])
```

Exercice 2 :

Créez un diagramme de dispersion des caractéristiques `cendre` et `couleur_intensit` de l'ensemble de données sur le vin.

Exercice 3 :

Créez une matrice de dispersion des caractéristiques de l'ensemble de données sur le vin.

Exercice 4 :

Récupérer le jeu de données des visages Olivetti et visualiser les visages.

Entrée [ ]: