Klasyfikacja i regresja:
Wstęp do biblioteki `scikit-learn`

24 października 2018

Opis pliku z zadaniami

Wszystkie zadania na zajęciach będą przekazywane w postaci plików `.pdf`, sformatowanych podobnie do tego dokumentu. Zadania będą różnego rodzaju. Za każdym razem będą one odpowiednio oznaczone:

- Zadania do wykonania na zajęciach oznaczone są symbolem $\triangle$ – nie są one punktowane, ale należy je wykonać w czasie zajęć.
- Punktowane zadania do wykonania na zajęciach oznaczone są symbolem $\diamond$ – należy je wykonać na zajęciach i zaprezentować prowadzącemu.
- Zadania do wykonania w domu oznaczone są symbolem $\star$ – są one punktowane, należy je dostarczyć w sposób podany przez prowadzącego i w wyznaczonym terminie (zwykle przed kolejnymi zajęciami).

1 Zapoznanie się z biblioteką `sklearn` △

Treść

W tym ćwiczeniu przedstawiony i omówiony jest krótki skrypt w języku `python`. Na podstawie danych uczących zostanie wytrenowany klasyfikator potrafiący rozróżnić cyfry na prostych obrazach o rozmiarze 8×8. Następnie klasyfikator zostanie wykorzystany do klasyfikacji pojedynczego przykładu.

Należy wykonać następujące kroki:

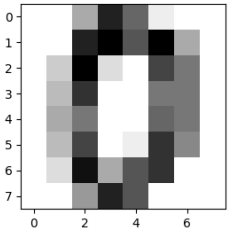
1. Zapoznaj się ze stroną główną projektu `sklearn`:
   http://scikit-learn.org/
   - Przewodnik użytkownika dostępny jest tutaj:
     http://scikit-learn.org/stable/user_guide.html
   - Bieżące ćwiczenie zostało opracowane na podstawie tutorialu:
     http://scikit-learn.org/stable/tutorial/basic/tutorial.html
   - Dodatkowo warto zapoznać się z typami danych obsługiwanymi przez `sklearn` oraz proponowanymi metodami wczytywania danych: http://scikit-learn.org/stable/datasets/index.html#external-datasets

2. Zaimportuj bibliotekę zawierającą przykładowe dane. Wczytaj plik zawierający obrazy cyfr oraz wypisz dane.

```
from sklearn import datasets

#Load the digits dataset
digits = datasets.load_digits()

#Print examples with their features
print(digits.data)

#Array with output (decision) values
digits.target

#The first example
digits.images[0]
```

3. Powyższe dane są zapisane wektorowo. Można jednak łatwo je przedstawić jako bitmapę.

```
import matplotlib.pyplot as plt

#Display the first digit
plt.figure(1, figsize=(3, 3))
plt.imshow(digits.images[0], cmap=plt.cm.gray_r,
    interpolation='nearest')
plt.show()
```



4. Wytrenuj Twój pierwszy klasyfikator używając wszystkich przykładów oprócz ostatniego.

```
from sklearn import svm

#Initialize the SVM classifier
clf = svm.SVC(gamma=0.001, C=100.)

#Train the classifier on training data
clf.fit(digits.data[:-1], digits.target[:-1])
```

5. Zastosuj klasyfikator do ostatniego przykładu i sprawdź czy predykcja jest poprawna (wizualnie oraz sprawdzając prawdziwą etykietę w danych).

```
#Predict the last digit
clf.predict(digits.data[-1:])

#The digit looks like
#Display the last digit
plt.figure(1, figsize=(3, 3))
plt.imshow(digits.images[-1], cmap=plt.cm.gray_r,
    interpolation='nearest')
plt.show()

#The true label of the last image
digits.target[-1]
```

6. Nauczony klasyfikator można zapisać wykorzystując bibliotekę `pickle` lub `joblib`.

```
import pickle

s = pickle.dumps(clf)
clf2 = pickle.loads(s)
clf2.predict(digits.data[-1:])

from sklearn.externals import joblib
joblib.dump(clf, '/home/username/clf.joblib')
clf3 = joblib.load('/home/username/clf.joblib')
clf3.predict(digits.data[-1:])
```