Réseaux de neurones

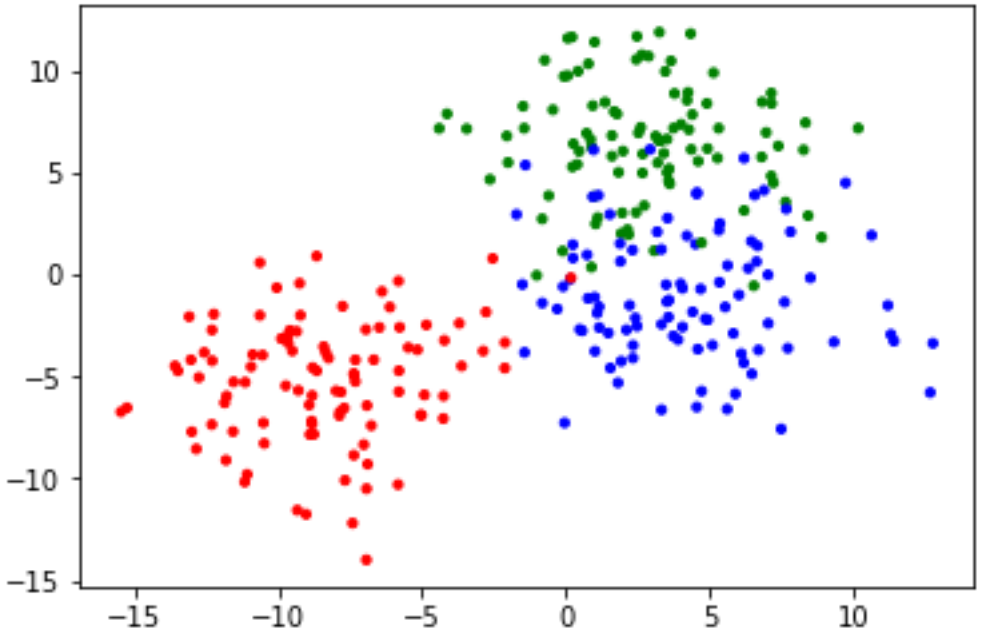
Micro tutoriel Python

Lionel PREVOST, Directeur de Recherche LDR Lab

(lionel.prevost@esiea.fr)

Charger et visualiser des données

```
#charger les données
import numpy as np
data = np.loadtxt('Path/dataset.dat')

#séparer les observations et les labels
X = data[:,0:2]
y = data[:,2]
y = y.astype(int)

#visualiser les données
from matplotlib import pyplot
colors = np.array([x for x in "rgbcmyk"])
pyplot.scatter(X[:, 0], X[:, 1], color=colors[y].tolist(), s=10)
pyplot.show()
```

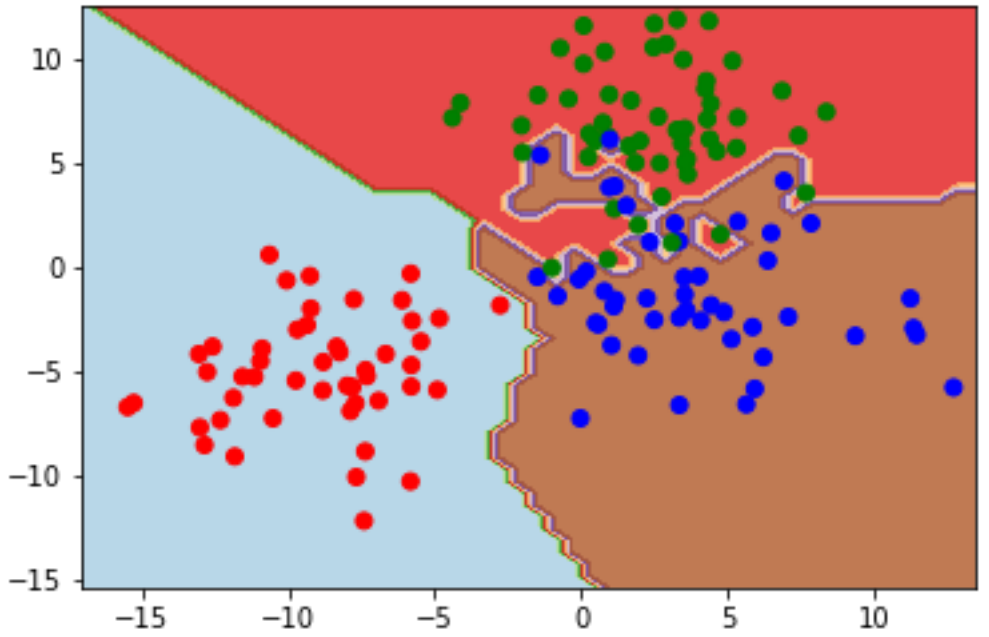


Entraîner et évaluer un modèle

```
#partition des données
from sklearn import model_selection
X_train, X_test, y_train, y_test = model_selection.train_test_split(X, y, train_size=0.7,
test_size=0.3)

#choisir et entraîner un modèle (KNN)
from sklearn.neighbors import KNeighborsClassifier
one_NN = KNeighborsClassifier(n_neighbors=1, algorithm='brute')
one_NN.fit(X_train, y_train)

#score sur la base d'apprentissage
print('accuraccy on training set:', one_NN.score(X_train, y_train))

#matrice de confusion
from sklearn import metrics
y_pred_test = one_NN.predict(X_test)
metrics.confusion_matrix(y_test, y_pred_test)
```

Visualiser les frontières

```
# Créer une grille
x_min, x_max = X[:, 0].min()*1.1, X[:, 0].max()*1.1
y_min, y_max = X[:, 1].min()*1.1, X[:, 1].max()*1.1
x_h = (x_max - x_min)/50
y_h = (y_max - y_min)/50
xx, yy = np.meshgrid(np.arange(x_min, x_max, x_h),
                     np.arange(y_min, y_max, y_h))
Y = one_NN.predict(np.c_[xx.ravel(), yy.ravel()])
Y = Y.reshape(xx.shape)
```




```
#afficher les frontières/données d'apprentissage
pyplot.contourf(xx, yy, Y, cmap=pyplot.cm.Paired, alpha=0.8)
pyplot.scatter(X_train[:, 0], X_train[:, 1], cmap=pyplot.cm.Paired, color=colors[y_train].tolist())
pyplot.xlim(xx.min(), xx.max())
pyplot.ylim(yy.min(), yy.max())
pyplot.show()
```

Faire varier un paramètre

```
#impact de la taille de la base d'apprentissage
max_size = X_train.shape[0]
acc = []
for size in range (1,100):
    X_train1, temp1, y_train1, temp2 = model_selection.train_test_split(#à completer#)
    one_NN.fit(#à completer#)
    acc.append(one_NN.score(#à completer#))
```