Tutoriel `scikit-learn` : les classifieurs SVM

Romain Tavenard

Voici l'en-tête à utiliser pour la suite de ce travail :

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.datasets.samples_generator import make_blobs
```

Pour cette partie du tutoriel, nous allons générer des données synthétiques simples. `sklearn` fournit des générateurs automatiques de données synthétiques, nous allons en utiliser un pour plus de facilité :

```
X, y = make_blobs(n_samples=50, centers=2, random_state=0,
                  cluster_std=0.60)
```

Bien sûr, il n'est pas question de manipuler des données sans les visualiser :

```
plt.figure()
plt.scatter(X[:, 0], X[:, 1], c=y, s=40, edgecolors='none')
plt.savefig("svm_lineaire.pdf")
```

Nous allons maintenant entraîner un SVM linéaire sur ces données. Pour cela, nous allons utiliser la classe `SVC` du module `svm` de `sklearn`, qui possède sa page dédiée sur le site du module `scikit-learn`. Pour créer et entraîner un tel modèle, rien de plus simple :

```
clf = SVC(kernel="linear")
clf.fit(X, y)
```

Une fois le modèle entraîné, l'objet `clf` peut vous fournir des informations sur :

- les vecteurs supports retenus, stockés dans l'attribut `support_vectors_` ;
- le vecteur W et le scalaire b utilisés dans la fonction de décision du SVM, stockés respectivement dans les attributs `coef_` et `intercept_`. On rappelle que la classe attribuée par le SVM linéaire dépend du signe de la quantité $WX + b$ (notez que WX signifie ici le produit matriciel entre W (de taille $1 \times d$) et X (de taille d)).

1. Commencez par entourer les points retenus pour être vecteurs supports. Pour cela, vous fournirez les données adéquates à la

fonction `plt.scatter` et lui passerez les arguments suivants : `marker="o", s=80, facecolors='none'`.

2. On souhaite maintenant tracer la séparatrice apprise par le SVM. Pour cela, on va se baser sur les points d'abscisse :

   ```
   x0_left = np.min(X[:, 0])
   x0_right = np.max(X[:, 0])
   ```

   Calculez l'ordonnée correspondante dans l'équation de la séparatrice et tracez le segment résultant entre les points `(x0_left, x1_left)` et `(x0_right, x1_right)`.

3. Chargez maintenant des données plus bruitées :

   ```
   X, y = make_blobs(n_samples=50, centers=2, random_state=0,
                     cluster_std=1.)
   ```

   Répétez l'entraînement du SVM sur ces données en faisant varier le paramètre `C` (entre `0.1` et `10` par exemple) du SVM et observez l'évolution du nombre de vecteurs supports choisis. Expliquez.

4. Chargez maintenant des données clairement non linéairement séparables :

   ```
   X, y = make_circles(100, factor=.1, noise=.1)
   ```

   Visualisez vos données. Comment modifier votre SVM pour qu'il ait des chances d'atteindre des performances raisonnables ? Mettez votre proposition en oeuvre et tracez les vecteurs supports retenus.

5. À l'aide de la méthode `predict` de la classe `SVC`, calculez le nombre de bonnes prédictions effectuées par votre classifieurs sur vos données.