TP 6

Méthodes à noyaux

Machines à vecteurs de support (SVM)

Les machines à vecteurs de support (SVM : *Support Vector Machines*) sont une classe des méthodes d'apprentissage statistique basées sur le principe de la maximisation de la marge (séparation des classes). Il existe plusieurs formulations (linéaires, versions à noyaux) qui peuvent s'appliquer sur des données séparables (linéairement) mais aussi sur des données non séparables.
Les avantages des SVM :

- Très efficaces en dimension élevée.
- Ils sont aussi efficaces dans le cas où la dimension de l'espace est plus grande que le nombre d'échantillons d'apprentissage.
- N'utilisent pas tous les échantillons d'apprentissage, mais seulement une partie (les vecteurs de support). En conséquence, ces algorithmes demandent moins de mémoire.

Désavantages :

- Si le nombre d'attributs est beaucoup plus grand que le nombre d'échantillons, les performances seront moins bonnes.
- Comme il s'agit de méthodes de discrimination entre les classes, elles ne fournissent pas directement des estimations de probabilités.

Scikit-learn

Dans Scikit-learn, les SVM sont implémentées dans le module `sklearn.svm`. Dans cette partie nous allons nous intéresser à la version à noyaux (Scikit-learn utilise la bibliothèque LibSVM déjà discutée).
Relisez le TP SVM linéaire (partie Scikit-learn) pour vous remettre dans le contexte (quels sont les classes Python utilisées et leurs paramètres). La documentation de scikit-learn sur les SVM vous sera utile.
Nous allons reprendre la classification sur les données Iris.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm, datasets
from sklearn.model_selection import train_test_split

# Chargement des données
iris = datasets.load_iris()
X, y = iris.data, iris.target
```

```
# On conserve 50% du jeu de données pour l'évaluation
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5)
```

Question

Refaites la classification de la base de données `iris` mais avec un noyau gaussien. Testez l'effet du paramètre d'échelle du noyau (`gamma`) et du paramètre de régularisation `C`.

Comme dans le TP précédent, nous pouvons afficher la frontière de décision en ne conservant que deux variables explicatives :

```
X, y = iris.data[:,:2], iris.target
# On conserve 50% du jeu de données pour l'évaluation
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5)
clf = svm.SVC(C=0.1, kernel='rbf', gamma=0.25)
clf.fit(X_train, y_train)

# Pour afficher la surface de décision on va discrétiser l'espace avec un
pas h
h = .02
# Créer la surface de décision discretisée
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))

# Surface de décision
Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.8)
# Afficher aussi les points d'apprentissage
plt.scatter(X_train[:, 0], X_train[:, 1], label="train", edgecolors='k',
c=y_train, cmap=plt.cm.coolwarm)
plt.scatter(X_test[:, 0], X_test[:, 1], label="test", marker='*', c=y_test,
cmap=plt.cm.coolwarm)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.title("SVM RBF")
```

Question

Que constatez-vous par rapport au TP précédent ?

Jeu de données Digits

Reprenons notre base de données Digits de chiffres manuscrits.

```
from sklearn.datasets import load_digits
digits = load_digits()
X, y = digits.data, digits.target
print(X.shape)
print(y.shape)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

Question

Réalisez une classification par une SVM linéaire et une SVM à noyau gaussien du jeu de données Digits. Comment est choisi le paramètre `gamma` dans scikit-learn ? Testez différentes valeurs de ce paramètre pour évaluer son influence. En particulier, testez les paramètres `gamma='auto'` et `gamma='scale'`. À quoi correspondent-ils ?

Question :

Réalisez une analyse en composante principale (ACP) et gardez les 2 premières composantes principales (voir la documentation Scikit-learn). Ensuite faites une classification avec un noyau gaussien et affichez les points de test ainsi que la surface de décision (reprendre le code du TP SVM linéaire). Comparez avec une SVM linéaire.

Question

Réalisez une recherche par grille afin de déterminer sur le jeu de données Digits complet (sans l'ACP) :

- le meilleur noyau à utiliser,
- la meilleure valeur de `C`,
- la meilleure valeur de `gamma` (ou le degré du polynôme pour un noyau polynomial).
- la meilleure valeur de `n_components` de l'ACP

Question

(optionnel) Combien de composantes faut-il garder au minimum dans l'ACP pour classer correctement au moins 97% des images ? À quel facteur de réduction de dimension cela correspond-il ?