Scikit-learn ゼミ

1.2.8. Implementation details

河野和平

入力データ

- array X
  - 訓練サンプルを格納
  - $size: [n_{sample}, n_{features}]$

- array Y
  - 訓練サンプルのクラスラベルを格納(整数値)
  - $size: [n_{sample}]$
  - 多クラス分類の場合もそのまま利用可能

関数(1)

- svm_problem(Y,X)
  - クラスラベルの格納された配列Y
  - 訓練サンプルの配列X
  - 訓練データの登録
- svm_parameter('option')
  - SVM,カーネルの種類を設定
  - それに付随するパラメータの設定
  - Libsvm:http://www.csie.ntu.edu.tw/~cjlin/libsvm/

関数(2)

- svm_train(problem,parameter)
  - 登録した訓練データとパラメータで学習を行う。

- svm_predict(test_label,test_data,train)
  - test_label: テストデータの期待する分類結果
  - test_data: テストデータ
  - 戻り値:分類結果

サンプルプログラム

```
import sys↓
sys.path.append('C:¥Users¥Hiromu¥Desktop¥libsvm-3.18¥python')↓
from svm import *↓
from svmutil import *↓
↓
Y = [1,-1,1,-1]↓
X = [[0.5, 0.0, 2.0], [-0.5, -1.00,-0.5], [0.35, 0.0, 0.35],[-2.0,-1.5,-0.5]]↓
problem = svm_problem(Y, X)↓
↓
parameter = svm_parameter('-s 0 -t 0')↓
↓
t = svm_train(problem, parameter)↓
↓
test_label = [1, 1, 1, -1]↓
test_data = [[0.7, 0.03, 0.8], [2.2, 3.13, 1.33], [1.1, 1.5, 0.8], [-0.3, -1.85, -0.2]]↓
result = svm_predict(test_label, test_data , t)↓
↓
for r in result:↓
        print r↓
```

実行結果

```
C:¥Users¥Hiromu¥Desktop¥sklearn>python libsvm.py
*
optimization finished, #iter = 1
nu = 0.408998
obj = -0.817996, rho = -0.513292
nSV = 2, nBSV = 0
Total nSV = 2
Accuracy = 100% (4/4) (classification)
[1.0, 1.0, 1.0, -1.0]
(100.0, 0.0, 1.0)
[[1.5807771018715582], [5.528016384920539], [3.061349706227305], [-1.34764827027
95871]]
```