計算機科学実験及演習 4 エージェント 課題 1

1029-28-2473 二見 颯

2018 年 10 月 12 日

1　プログラム概要

データ点が書かれたファイルを読み込み、SVM の識別器を作成してそのパラメータを出力する。作成した SVM によりクラスの予測を行い、与えられたデータ点と予測された識別境界を平面上にプロットする。

2　外部仕様

- main.py - プログラムのエントリポイント。svc.py の SVClassifier および utils.py の関数を参照している
- svc.py - SVM のクラス SVClassifier を実装
- utils.py - ファイル読み込み (load_data) と結果のプロット (plot_decision_regions) を実装

プログラムの実行は
./main.py [入力ファイルへのパス] [カーネルトリックの種類] [カーネルトリックのパラメータ]
カーネルトリックの種類については、n でカーネルトリックなし、p で多項式カーネル、g でガウスカーネル、s でシグモイドカーネルとする。プログラム引数が正しくない形式の場合は、エラーとしてプログラムが終了する。

```
$ ./main.py data/square.dat n
    pcost dcost gap pres dres
0: -1.2457e-01 -3.5986e-01 2e-01 7e-18 1e+00
1: -1.2497e-01 -1.2743e-01 2e-03 5e-17 3e-02
2: -1.2500e-01 -1.2502e-01 2e-05 2e-17 3e-04
3: -1.2500e-01 -1.2500e-01 2e-07 4e-17 3e-06
4: -1.2500e-01 -1.2500e-01 2e-09 3e-17 3e-08
Optimal solution found.
alpha:
0 [0.0625]
1 [0.0625]
2 [0.0625]
3 [0.0625]
theta candidates: [0.9999999680249498, 1.0, 0.9999999680249498, 1.0]
SVClassifier W = [0.49999999 0. ], θ = 0.9999999840124749
```

```
16 [グラフ出力]
```

必要なライブラリ等は以下

```
1 Python 3.6.1 :: Anaconda custom (64-bit)
2 numpy 1.14.2
3 cvxopt 1.2.1
4 tqdm 4.26.0
5 matplotlib 2.0.2
```

3 内部仕様

svc.py

SVClassifier クラス

サポートベクタマシンを定義する. private メンバ

- X, y - 訓練データ点とその正解クラス
- n - 訓練データの個数
- kf - kernel trick として用いる関数 (kernel function)
- p, q - kernel function のパラメータ (optional)
- alpha, theta - SVM の内部パラメータであり、それぞれ setLagrange 関数, setClassifier 関数で決定する

コンストラクタにより、kf, p, q を決定する

fit

X, y, n を決定して、alpha, theta を決定する各関数を呼び出す
@param[in] X, y

_setLagrange

以下の 2 次計画問題を cvxopt.solvers.qp を用いて解くことで、alpha を決定する

_setClassifier

alpha と x, y から theta を決定して、”W = , θ = ”の形で識別器を出力する

_kernelPolynomial

多項式カーネル

_kernelGauss

_kernelSigmoid

predict

SVM により tX のクラスの予測をする
@param[in] tX テストデータ
utils.py

load_data

filepath にある sample_linear.dat 同様の形式のデータを X,y に読み込む
@param[in] filepath
@return X, y
入力ファイルのデータ点は、行の最後に正解クラスが指定される以下のような形式で与えられる。

```
1.0, 0.5, 1
0.5, 0.25, -1
```

1 行ごとに", "で split して処理するためデータ点が n 次元の特徴で表せる場合にも対応できる。

plot_decision_regions

x を 2 次元平面に plot する. plot された点の色は y によって決まり、領域の色は model の予測結果により決まる
@param[in] x, y, model, resolution resolution によりグリッドの幅を指定する
@return なし
x が 2 次元でない場合、この関数は終了する。
x=(x1, x2) に対する正解クラスを y, SVM による予測クラスを z とする。x1 の最小値-1 から最大値 +1, x2 の最小値-1 から最大値 +1 の範囲でグリッド点を作成して、そのグリッド点に対して model.predict することで z の決定領域が得られる。
その後、各 x の点を y に応じた色 (red, blue) と形 (o, x) で plot する。

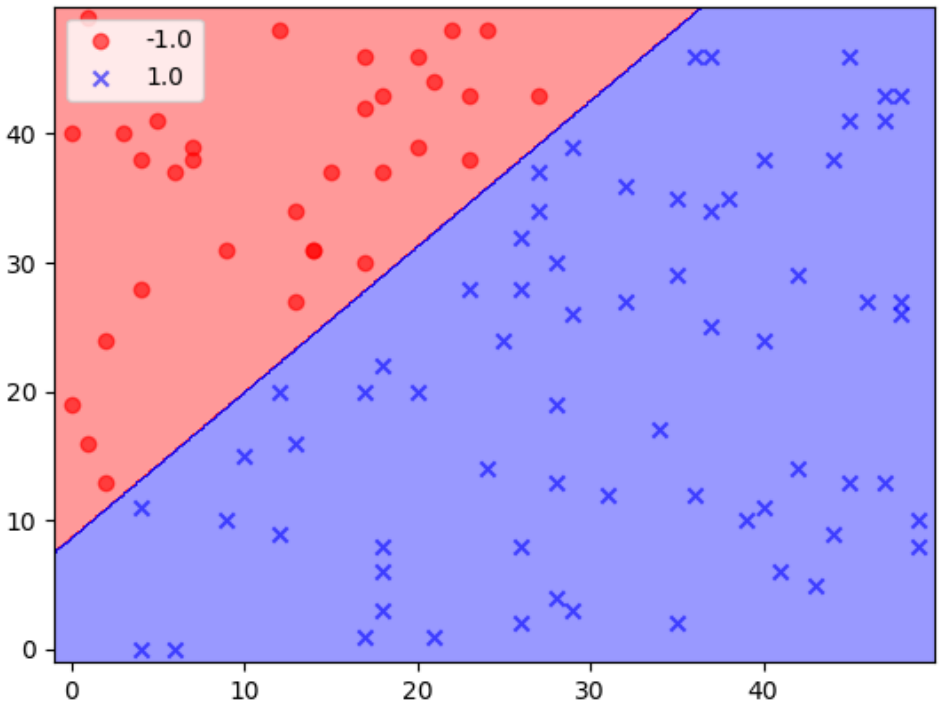

図 1　sample_linear.dat

4　評価結果

4.1　sample_linear.dat

4.2　sample_circle.dat

4.3　poly_five.dat

1 次元データ x=[1, 2, 5, 7, 8], y=[1, 1, -1, -1, 1] を poly_five.dat とする。このとき、α について以下のような結果が得られた。

```
alpha:
0 [2.86101852e-09]
1 [1.01333335]
2 [8.47919771e-08]
3 [4.07999985]
4 [3.06666658]
```

5 考察

SVClassifier.predict の実装について、はじめは

```
def predict(self, tX):
    batch_size = tX.shape[0]
    preds = []
    for i in range(batch_size):
        res = 0
        for j in range(self.__n):
            res += self.__alpha[j] * self.__y[j] * self.__kf(self.__X[j], tX[i])
        res -= self.__theta
        preds.append(np.sign(res))
    return np.array(preds)
```

のように実装していたが、この場合 for 文を 2 重に

6 参考資料

- Python 機械学習プログラミング Sebastian Raschka 著
- はじめてのパターン認識 平井 有三 著