計算機科学実験及演習 4 (エージェント) レポート 2

佐竹誠

2018/10/19

1　プログラム概要

サポートベクターマシンによってデータ点が書かれたファイルを読み込み、識別器を作成。その後交差検証を行い識別器の予測精度を表示します。

2　外部仕様

2.1　プログラム名

estimata.cpp がプログラムファイルです。

2.2　ファイルの説明

2.2.1　estimate.cpp

サポートベクターマシンによって作成された識別器の予測精度を表示するプログラムファイルです。

2.2.2　quadprog++.cc

二次計画問題を解くライブラリです。estimate.cpp で使用しています。

2.2.3　quadprog++.hh

quadprog++.cc のヘッダーファイルです。estimate.cpp でインクルードしています。

2.2.4　sample_linear.dat

線形識別可能なサンプルデータです。

2.2.5　sample_circle.dat

線形識別不可能なサンプルデータです。

2.2.6　Makefile

コンパイルするための make コマンドを使えるようにするファイルです。

2.3　入力方法

プログラムを実行すると、まず「データファイルを指定してください：」と表示されるので、使用するデータファイルを指定してください。ただしデータ数の上限は 100 個になっています。

次に「データの次元数を指定してください：」と表示されるので、使用するデータの次元数を指定してください。

次に「使用するカーネルを指定してください (0:カーネルトリックなし　1:多項式カーネル　2:ガウスカーネル)：」と表示されるので、使用したいカーネルを数字で指定してください。ガウスカーネルを指定した場合は続いて「ガウスカーネルで使用するシグマ定数を指定してください：」と表示されるのでシグマ定数を指

定してください。

最後に「交差検証を行う際のデータ分割数を指定してください：」と表示されるので、データ分割数を正整数で指定してください。

2.4 出力

指定された分割数に従って分割したブロックごとの結果を表示したのち、平均値を出力します。各結果の意味は以下の通りです。

2.4.1 accuracy

正解率のこと。予測結果全体と、答えがどれぐらい一致しているかを判断する指標。

2.4.2 precision

適合率のこと。予測を正と判断した中で、答えも正のもの。

2.4.3 recall

再現率のこと。答えが正の中で、予測が正とされたもの。

2.4.4 F_measure

F 値のこと。Presicion と Recall の調和平均。予測精度の評価指標。

2.5 コンパイル方法

プログラムファイルがあるディレクトリで make コマンドを実行すると quadprog++.cc と estimate.cpp のコンパイルが行われ、svm という実行可能ファイルができます。実行後は画面の指示に従って条件を入力してください。

2.6 実行例

学習データを sample_linear、次元数を 2、カーネルトリックなし、分割数 5 で学習させた結果は次のようになります。

```
$ ./svm
データファイルを指定してください：sample_linear.dat
データの次元数を指定してください：2
使用するカーネルを指定してください (0:カーネルトリックなし　1:多項式カーネル　2:ガウスカーネル)：0
交差検定を行う際のデータ分割数を指定してください：5
estimate_block： 0
accuracy： 0.8
precision： 0.764706
recall： 1
F_measure： 0.866667
.....
estimate_block： 1
accuracy： 0.95
precision： 1
recall： 0.933333
F_measure： 0.965517
.....
estimate_block： 2
accuracy： 0.55
precision： 0.55
recall： 1
F_measure： 0.709677
.....
estimate_block： 3
accuracy： 0.75
precision： 0.75
recall： 1
F_measure： 0.857143
.....
estimate_block： 4
accuracy： 0.7
precision： 0.684211
recall： 1
F_measure： 0.8125
.....
Average
accuracy： 0.75
precision： 0.749783
recall： 0.986667
F_measure： 0.842301
```

2.7　エラー処理

2.7.1　指定されたデータファイルがないとき

全ての入力が終わった時点で”ファイルが見つかりません”と表示し、プログラムを終了します。

2.7.2　指定されたカーネルが正しくないとき

全ての入力が終わった時点で”正しくカーネルを指定してください (0:カーネルトリックなし　1:多項式カーネル　2:ガウスカーネル)”と表示し、プログラムを終了します。

2.7.3　指定された分割数が負のとき

全ての入力が終わった時点で”データ分割数は正整数で指定してください”と表示し、プログラムを終了します。

3　内部仕様

3.1　主要な大域変数の説明

3.1.1　double G[][]

二次計画問題を解く際に使用するデータを格納する double 二次元配列です。

3.1.2　double g0[]

二次計画問題を解く際に使用する double 配列です。関数の一階微分を表しており、本プログラムでは全ての要素に-1 が格納されています。

3.1.3　double CE[][]

二次計画問題を解く際に使用する double 二次元配列です。本プログラムではラベルの配列を設定しています。

3.1.4　double ce0[]

二次計画問題を解く際に使用する double 配列です。本プログラムでは全て 0 に設定しています。

3.1.5　double CI[][]

二次計画問題を解く際に使用する double 二次元配列です。本プログラムでは単位行列に設定しています。

3.1.6　double ci0[]

二次計画問題を解く際に使用する double 配列です。本プログラムでは全て 0 に設定しています。

3.1.7　double alpha[]

二次計画問題の計算結果として出力される α ベクトルを格納する double 配列です。

3.1.8 int n

二次計画問題を解く際に使用する int 変数です。本プログラムではデータの数が設定されています。

3.1.9 int m

二次計画問題を解く際に使用する int 変数です。CE と ce0 の配列の要素数を指定しており、本プログラムでは n と同じ数に設定しています。

3.1.10 int p

二次計画問題を解く際に使用する int 変数です。CI と ci0 の要素数を指定しており、本プログラムでは 1 に設定しています。

3.1.11 int N

データの次元数を表す int 型の変数です。入力から受け取った値をそのまま格納します。

3.1.12 int kernel

使用するカーネルを表す int 型の変数です。0,1,2 の三値しか取らず、それぞれカーネルトリックなし、多項式カーネル、ガウスカーネルを表します。入力から受け取った値をそのまま格納します。

3.1.13 std::string file_name

使用するデータファイルの名前を表す string 型の変数です。入力から受け取った値をそのまま格納します。

3.1.14 double sigma

ガウスカーネルを使用する際に使用する double 型の変数です。入力から受け取った値をそのまま格納します。

3.1.15 int d

多項式カーネルを使用する際に使用する int 型の変数です。本プログラムでは 2 に設定しています。

3.1.16 int data_n

交差検証の分割数を表す int 型の変数です。入力から受け取った値をそのまま格納します。

3.1.17 double data[][]

データの生のデータを格納する double 配列です。

3.1.18 double label[]

データのラベルを格納する double 配列です。

3.1.19 double theta

得られた識別器の閾値を格納する double 型の変数です。

3.1.20 int total_data_size

使用するデータの数を表す int 型の変数です。

3.1.21 int estimate_data_size

評価に使用するデータの数を表す int 型の変数です。

3.1.22 int training_data_size

学習に使用するデータの数を表す int 型の変数です。

3.2 各関数の説明

3.2.1 double kernel_result(double* x, double* y, int kernel, int degree, double sigma, int d)

カーネルの計算結果を得るための関数です。計算結果を double 型で返します。各引数の意味は以下の通りです。

- double* x,y
  計算する二つの配列の引数です。
- int kernel
  使用するカーネルを表す引数です。(0:内積 1:多項式カーネル 2:ガウスカーネル)
- int degree
  データの次元数を表す引数です。
- double sigma
  ガウスカーネル計算時に使用するシグマ定数を表す引数です。
- int d
  多項式カーネル計算時に使用する次数を表す引数です。

3.2.2 double get_norm(double* x, double* y, int degree)

二乗ノルムを得るための関数です。計算結果を double 型で返します。各引数の意味は以下の通りです。

- double* x,y
  計算する二つの配列の引数です。
- int degree
  データの次元数を表す引数です。

4 評価結果

各カーネルを sample_linear.dat と sample_circle.dat に対して、分割数 5 で行いました。各評価値の平均値を結果として示します。

4.1 カーネルトリックなし

4.1.1 sample_linear.dat

- accuracy : 0.75
- precision : 0.749783
- recall : 0.986667
- F_measure : 0.842301

4.1.2 sample_circle.dat

- accuracy : 0.33
- precision : 0.33
- recall : 1
- F_measure : 0.488794

4.2 多項式カーネル

4.2.1 sample_linear.dat

- accuracy : 0.98
- precision : 0.971429
- recall : 1
- F_measure : 0.985185

4.2.2 sample_circle.dat

- accuracy : 0.94
- precision : 0.866667
- recall : 0.925
- F_measure : 0.888485

4.3 ガウスカーネル ($\sigma = 10$)

4.3.1 sample_linear.dat

- accuracy : 0.93
- precision : 0.945833
- recall : 0.96
- F_measure : 0.948392

4.3.2 sample_circle.dat

- accuracy：0.87
- precision：0.730952
- recall：0.975
- F_measure：0.825967

5 考察

5.1 カーネルごとの線形分離可能なデータと線形分離不可能なデータに対する予測精度の比較

5.1.1 カーネルトリックなし

線形分離可能なデータは高い精度で予測できているのに対し、線形分離不可能なデータは半分も正しく予測できていません。これは線形カーネルという特徴を正しく表す結果となりました。

5.1.2 多項式カーネル

どちらも非常に高い精度を示しています。

5.1.3 ガウスカーネル

どちらも高い精度を示しましたが、特に線形分離可能なデータに対して高い精度を示しています。

5.2 カーネル間の予測精度の比較

線形分離可能なデータに対しては多項式カーネルが最も精度が良く、僅差でガウスカーネルが劣り、カーネルトリックなしではさらに低い精度を示しています。このことから、線形分離可能なデータに対してであっても、非線形カーネルを用いた方が良い精度が得られる場合があることがわかります。

また、線形分離不可能なデータに対しても多項式カーネルが最も精度が良く、次にガウスカーネルが良い結果を示しています。

5.3 ガウスカーネルのシグマ定数を変化させたときの線形分離不可なデータに対する予測精度の変化

ガウスカーネルのシグマ定数を変化させたときの予測精度の変化を調べます。線形分離不可能なデータに対して予測を行い、分割数は 5 にしました。結果は表 1 のようになりました。

シグマ定数を大きくするにつれて accuracy,precision,F-measure が高くなっています。代わりに recall は低くなっています。

シグマ定数を小さくすると過学習気味になり、大きくすると境界が直線的になることはレポート 1 で確認しました。参考にシグマ定数を変化させたときの識別器を可視化したものとして、図 1 と図 2 を掲載しておきます。

表 1　シグマ定数を変化させた時の予測精度の変化

σ	accuracy	precision	recall	F-measure
5	0.86	0.715	0.925	0.796068
10	0.87	0.730952	0.975	0.825967
30	0.93	0.872222	0.896429	0.881139
50	0.95	0.916667	0.896429	0.9031
100	0.95	0.916667	0.896429	0.9031

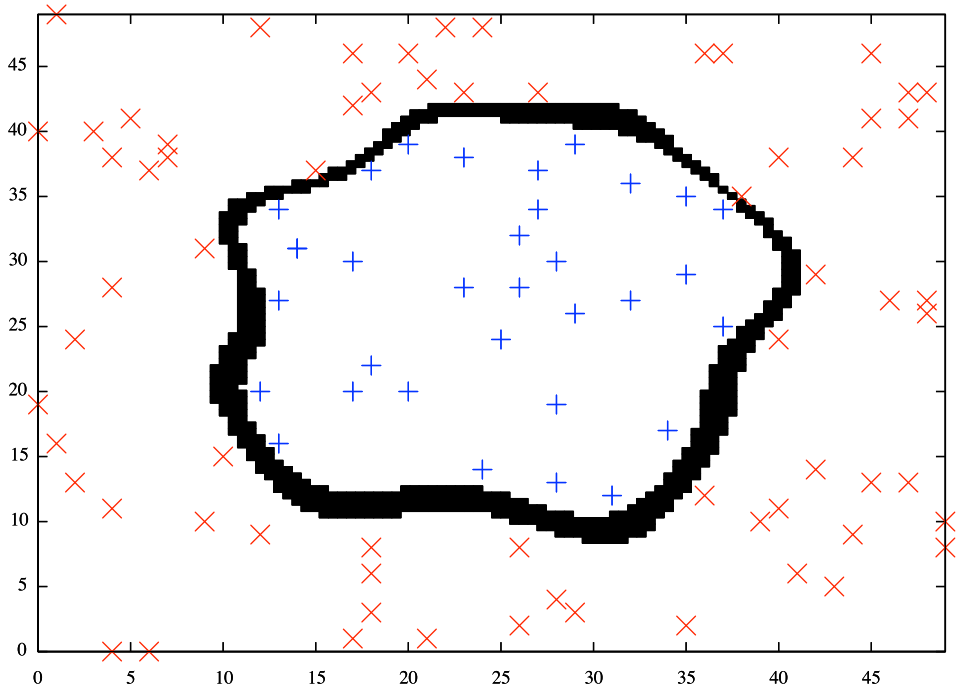

図 1　シグマ定数を 5 にしたときの識別器を可視化したもの

今回のデータセットではデータがきっちりと識別できるようになっていることと識別器が円状に描かれるよう分布になっていることから、マージンが細くても正しく識別できるためこのような結果になったと考えられます。現実のデータには例外データも多く含まれ、綺麗な識別も難しくなると考えられるため、シグマ定数を大きくすれば良くなるといわけでないと考えられます。

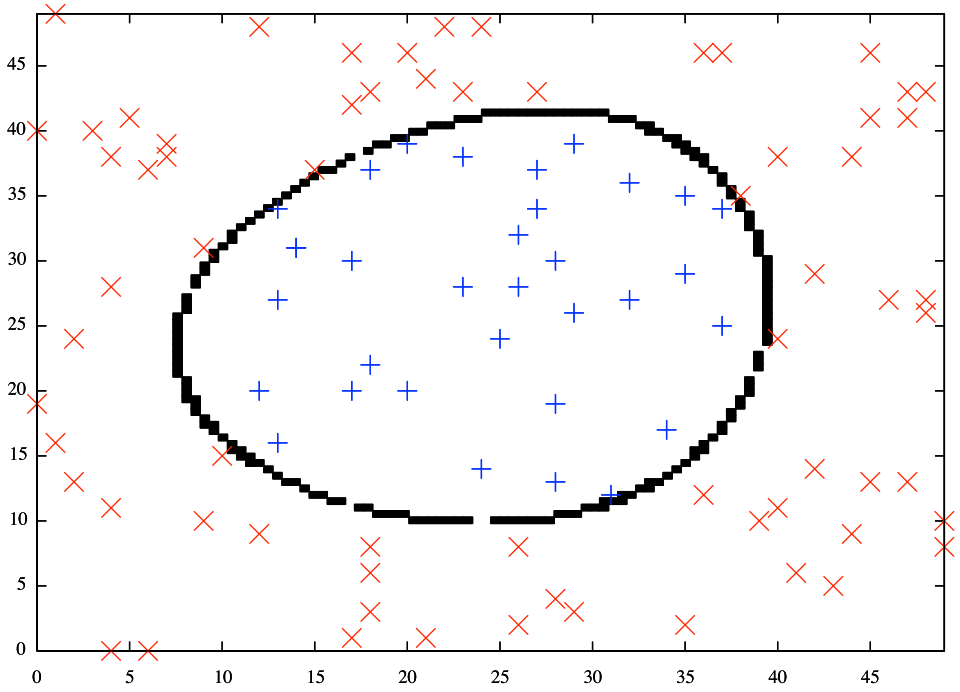

図 2　シグマ定数を 30 にしたときの識別器を可視化したもの