06t4027h 斎藤高行

第8章　サポートベクターマシン

8.7　Rによるサポートベクターマシン

関数svmとは

- 関数svmの実体は“LIBSVM”というフリーウェア

- 利用できるカーネルは4つ
  1. 線形カーネル
  2. 多項式カーネル
  3. RBFカーネル
  4. シグモイドカーネル

  ソフトマージンの表現も可能

関数svmとは

- SVMの枠組みで実行できること
  1. 2クラス分類問題
  2. 多クラス分類問題
  3. 回帰分析

ここでは2クラス分類問題のみを扱う

関数svmとは

- 交差妥当化検証が自動的に行われる
- 関数svmに含まれる引数crossに任意の整数を与えて重交差妥当化検証ができる
- 関数tune.svmを利用すると、グリッドサーチと重交差妥当化検証が同時に行うことができる

偽札データの分析

- パッケージ”e1071”と、データをRに読み込む

```
> library("e1071");
 要求されたパッケージ class をロード中です
> お札<-read.csv("お札.csv",header=T);
```

パッケージ”e1071”には、いくつもの統計手法の関数が含まれる

お札.csvは、「真偽」「下部マージン幅」「対角線長さ」について200個のデータが記述されている

関数svmの実行法

- RBFカーネル, C=1, γ=1に設定

  “radial”はRBFカーネルの指定、チューニング母数はソフトウェアのデフォルト

- 学習を済ませ、真偽の判別を行わせる

```
> お札結果<-svm(真偽~.,data=お札,probability=TRUE,kernel="radial",cross=1)
> お札予測値<-predict(お札結果,newdata=お札,probability=TRUE,decision.values=TRUE)
> table(お札予測値,お札[,1])

お札予測値 偽札 真札
      偽札  100    1
      真札    0   99
```

関数svmの実行法

- Probability=TRUE

  関数predictにおいて、オブザベーションのクラス所属確率を算出させる指定

- Decision.value=TRUE

  各オブザベーションの判別結果と、$f(x_i)$ の予測値を算出させる指定

解の抽出

- 関数svmの実行、”お札結果”オブジェクトを作成

```
> お札結果

Call:
svm(formula = 真偽 ~ ., data = お札, probability = TRUE, kernel = "radial", cross = 1)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.5

Number of Support Vectors:  19
```

学習結果の要約を出力している。 cost:C, gamma:γ,

Number of Support Vectors:サポートベクターの数

解の抽出

- “お札結果”オブジェクトの中身を確認

```
> t(round(head(お札結果$SV,10),3))
        3      4      7     11    13     27     29     61     71     80
x1 -0.497  0.334 -0.081 -0.981 0.196 -0.012  0.472 -1.397 -0.497 -0.843
x2 -0.246 -0.159 -0.246 -1.114 0.101 -0.246 -0.159 -1.808 -2.329 -2.329
> t(round(head(お札結果$coefs,10),3))
     [,1] [,2] [,3]  [,4] [,5] [,6]  [,7]  [,8]  [,9] [,10]
[1,]    1    1    1 0.794    1    1 0.006 0.396 0.187 0.001
> round(お札結果$rho,3)
[1] -0.232
> (ガンマ<-お札結果$gamma)
[1] 0.5
> (C<-お札結果$cost)
[1] 1
```

“お札結果$SV”　　: サポートベクタのIDと実現値を抽出

“お札結果$coefs”　　: 係数の抽出

“お札結果$rho”　　: 閾値bの抽出

解の抽出

サポートベクターと係数にはheadを利用しているので、一部抜粋の指定となる

Roundを利用して、小数点第3位まで求めている

サポートベクターを含め、学習モデルの母数となる

未知のデータに関する予測値の算出には、これらの母数の情報を利用することになる

解の抽出

- 予測値と、所属の予測確率を求める

```
> 予測値<-round(attr(お札予測値,"decision.values"),3)
> 予測確率<-round(attr(お札予測値,"probabilities"),3)
```

attr関数: オブジェクトに含まれる特定の情報を抽出する

Roundを利用して、小数点第3位まで求めている

解の抽出

- 求めた予測値と、予測確率の抜粋を作る

```
> t(head(予測値,8));t(tail(予測値,8))
                1     2     3     4    5     6     7     8
偽札/真札 1.518 1.584 0.157 0.899 1.26 1.566 0.658 1.454
              193    194    195    196   197    198    199    200
偽札/真札 -1.752 -1.347 -1.621 -1.759 -1.26 -1.583 -1.668 -1.512
```

```
> t(head(予測確率,8));t(tail(予測確率,8))
          1     2     3     4     5     6     7     8
偽札 0.994 0.995 0.632 0.951 0.985 0.995 0.901 0.992
真札 0.006 0.005 0.368 0.049 0.015 0.005 0.099 0.008
        193   194   195   196   197   198   199   200
偽札 0.003 0.012 0.005 0.003 0.015 0.005 0.004 0.007
真札 0.997 0.988 0.995 0.997 0.985 0.995 0.996 0.993
```

図の描画

- 3次元プロット用の関数”予測値算出”を用意する

```
> 予測値算出<-function(x,y){
+ サポートベクター<-お札結果$SV;係数<-お札結果$coefs
+ ガンマ<-お札結果$gamma;mad<-as.matrix(cbind(x,y))
+ mat<-scale(mad); zero<-numeric(length(x)*length(サポートベクター[,1]))
+ カーネル<-matrix(zero,nrow=length(x),ncol=length(サポートベクター[,1]),byrow=F)
+ 重み付け<-matrix(zero,nrow=length(y),ncol=length(サポートベクター[,1]),byrow=F)
+ for(i in 1:length(x)){
+ for(s in 1:length(サポートベクター[,1])){
+ カーネル[i,s]<-exp(ガンマ*sqrt(t(mat[i,1:2]-サポートベクター[s,1:2])%*%
+ (mat[i,1:2]-サポートベクター[s,1:2]))^{2})
+ 重み付け[i,s]<-係数[s]*カーネル[i,s]
+ }
+ }
+ 和<-apply(重み付け,1,sum); 予測値<-和-お札結果$rho
+ }
```

図の描画

P.255-図8.8のような、入力ユニットが2つのケース(2次元空間での分離問題)は稀少

そのため、パッケージ”e1071”にも3次元プロット用の関数は含まれていない

“予測値算出”は、任意の2組のベクトルに関して、学習マシンを通じた予測値を算出する関数である

図の描画

- 予測値の曲面を描画する

```
> #予測曲線のプロット
> library(scatterplot3d);N<-30
> x<-seq(-3,3,length=N);y<-seq(-3,3,length=N)
> persp(x,y,outer(x,y,予測値算出),zlab="f(x1,x2)",theta=133,phi=20,
+ ticktype="detailed",xlab="x1",ylab="x2")
```

- 等高線とサポートベクターを描画する

```
> #等高線とサポートベクターのプロット
> contour(x,y,outer(x,y,予測値算出),lwd=1,nlevels=10,labcex=1,
+ xlim=c(-3,3),method="edge")
> par(new=T)
> ID<-as.numeric(labels(お札結果$SV)[[1]])
> ラベル<-as.numeric(お札[,1][ID])+1
> plot(お札結果$SV,xlim=c(-3,3),ylim=c(-3,3),pch=ラベル)
```

図の描画

prespは、x,y,zの3次元空間に任意の曲面を描画する

zはx, yで構成される関数を定義し、そこで予測値算出を用いている

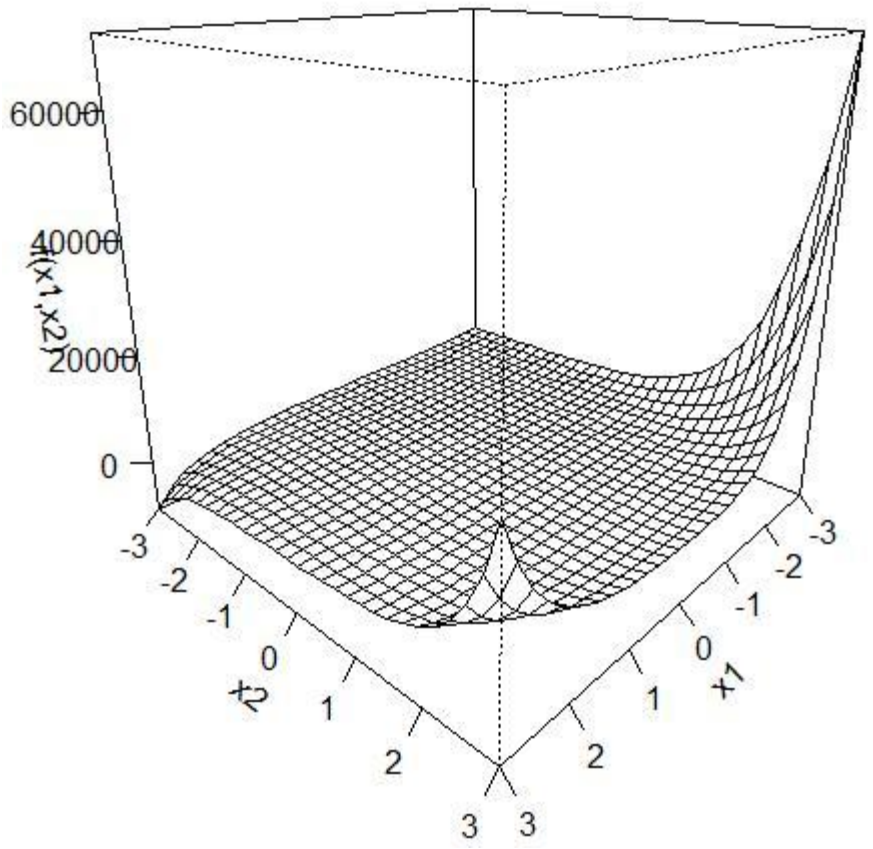
x,yには-3~3の区間を30等分した値を代入し、描画している

よりグリッドの細かい曲面を描画するためには、N>30に設定する

等高線ではcontourを利用し、prespと同様にzはx,yで構成される関数である予測値算出を指定する

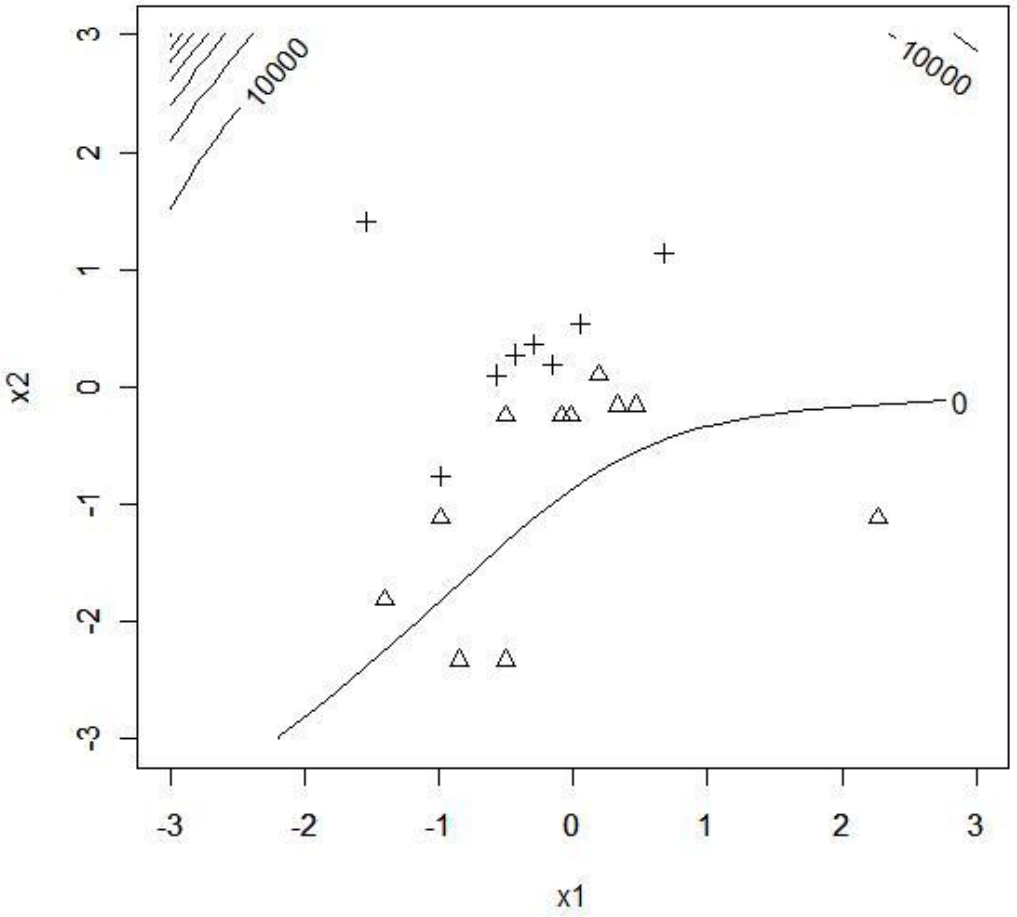
図の描画

- 予測曲線のプロット画面



図の描画

- 等高線とサポートベクターのプロット画面



図の描画

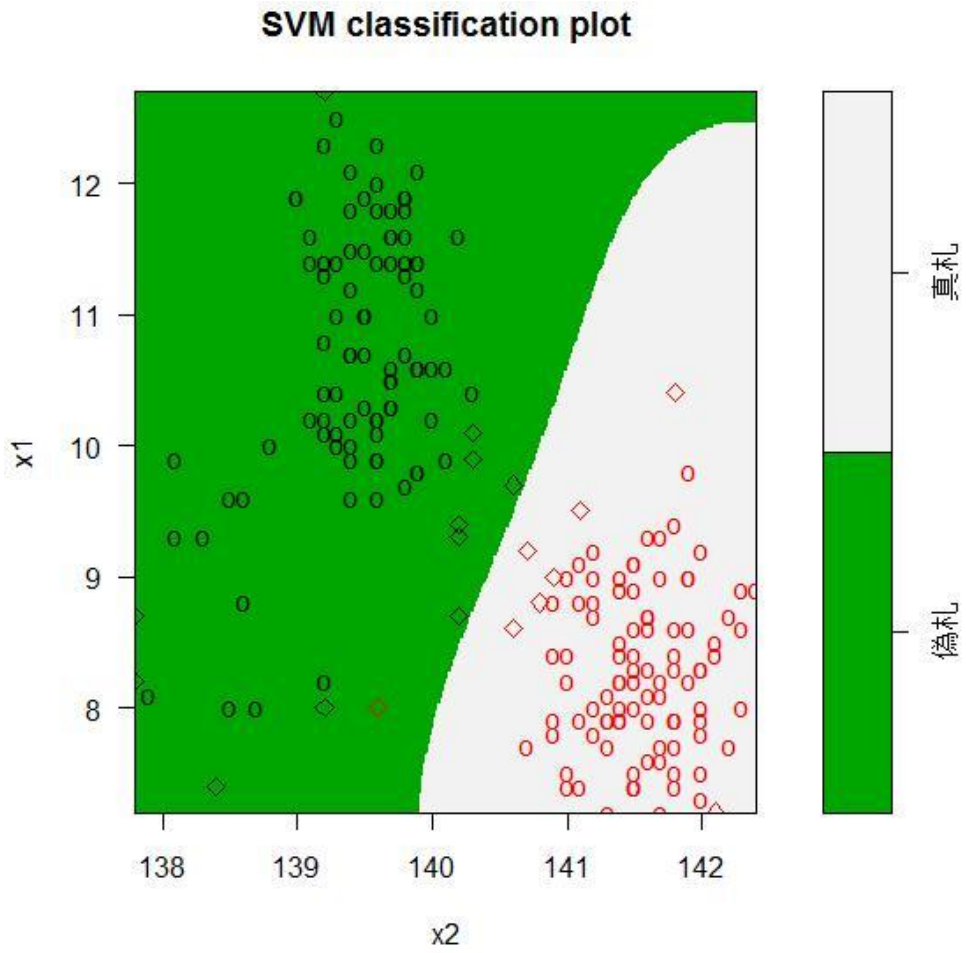
- クラスを分類する超平面の図示を指定

```
> plot(お札結果,お札,grid=300,color.palette=terrain.colors,svSymbol=5)
```

プロット画面は次頁

等高線プロットは、予測値が構成する曲面の標高がプロットされることで、学習マシンの予測性能のより詳細な考察が可能になるため、これと同様に重要である。

図の描画


SVM classification plot
x1
12
11
10
9
8
138
139
140
141
142
x2
真札
偽札

グリッドサーチ

- tune.svmを利用してグリッドサーチを行う

```
> チューニング<-tune.svm(真偽~.,data=お札,gamma=2^c(seq(-2,1,0.25)),
+ cost=2^c(seq(-2,1,0.25)))
```

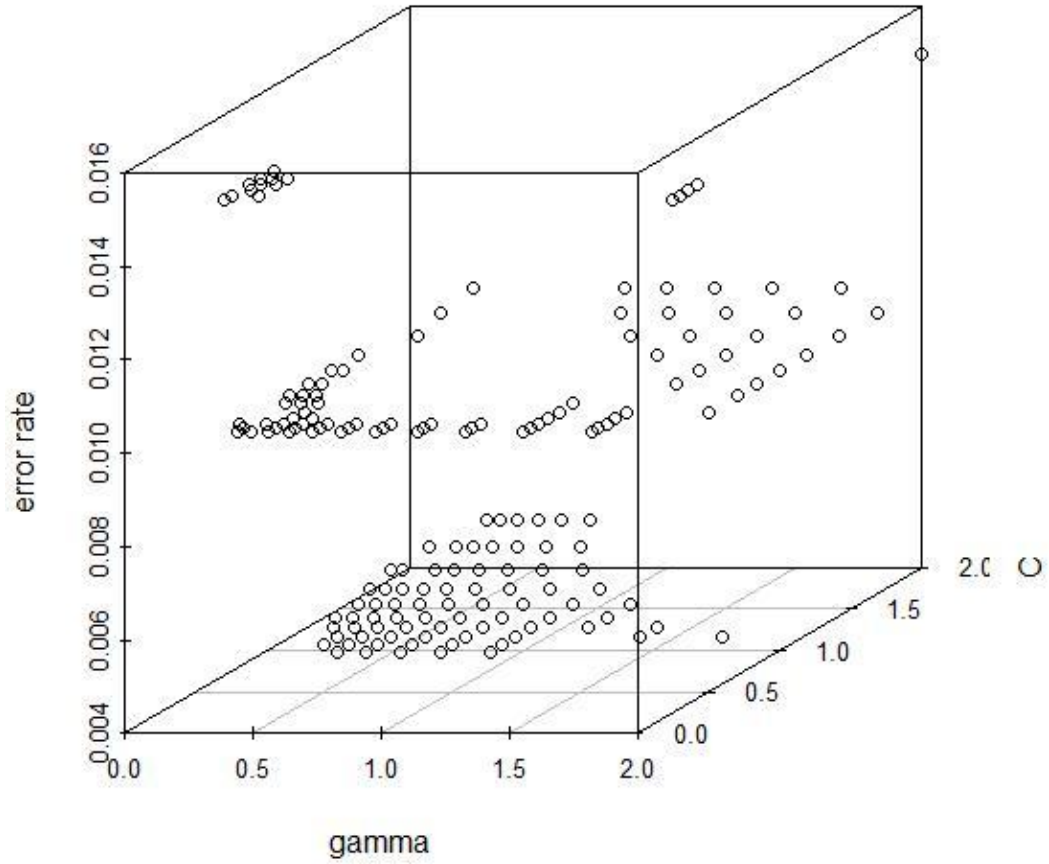
グリッドサーチ

- チューニング出力を指定する

```
> #各モデルでのご判別率の算出
> 結果<-summary(チューニング)
>
> #グリッドサーチ結果誤判別率の3次元プロット
> scatterplot3d(結果[7]$performances[,1],結果[7]$performances[,2],
+ 結果[7]$performances[,3],xlab="gamma",ylab="C",zlab="error rate")
```

グリッドサーチ

- グリッドサーチの結果誤判別率の3次元プロット



スパムデータの分析

- データの読み込みと、データセット折半

```
> #データの読み込み
> library(kernlab)
> data(spam)
>
> #データセットの折半
> index<-sample(1:dim(spam)[1],dim(spam)[1]/2)
> 学習スパム<-spam[-index,]
> テストスパム<-spam[index,
+ ]
```

スパムデータの分析

「スパム」のIDの半数を無作為抽出し、オブジェクトであるindexに保存する

Indexに含まれるIDが、該当しないデータを学習スパム、該当するデータをテストスパムとして折半

Sample(a, b)は、a個のデータの中からb個のデータを無作為に抽出する関数である

スパムデータの分析

- 同一データに対し、4つのモデルを適用する

```
> #学習
> RBFモデル<-svm(type~.,data=学習スパム,kernel="radial")
> 多項式モデル<-svm(type~.,data=学習スパム,kernel="polynomial")
> シグモイドモデル<-svm(type~.,data=学習スパム,kernel="sigmoid")
> 線形モデル<-svm(type~.,data=学習スパム,kernel="linear")
>
> #予測
> RBF予測結果<-predict(RBFモデル,テストスパム[,-58])
> 多項式予測結果<-predict(多項式モデル,テストスパム[,-58])
> シグモイド予測結果<-predict(シグモイドモデル,テストスパム[,-58])
> 線形予測結果<-predict(線形モデル,テストスパム[,-58])
```

スパムデータの分析

- 前頁の続き

```
> #クロス表
> table(RBF予測結果,テストスパム$type)

RBF予測結果 nonspam spam
    nonspam    1316  116
    spam         53  815
> table(多項式予測結果,テストスパム$type)

多項式予測結果 nonspam spam
       nonspam    1339  520
       spam         30  411
> table(シグモイド予測結果,テストスパム$type)

シグモイド予測結果 nonspam spam
          nonspam    1243  163
          spam        126  768
> table(線形予測結果,テストスパム$type)

線形予測結果 nonspam spam
     nonspam    1300  104
     spam         69  827
```

スパムデータの分析

- 偽札データの分析とは異なり、予測値の算出にテストデータを利用している

- この分析では通常の交差妥当性検証をしている

  重交差妥当化を併用したグリッドサーチが望ましい

  しかし、データ数と入力ユニット数が多いため、計算に時間がかかる

- 実務・研究場面等では、重交差妥当化を併用したグリッドサーチを行うべきである