HW2 - Support Vector Machine

Anna Frigge
Csongor Horváth

UPPSALA
UNIVERSITET

Support Vector Machine

Background

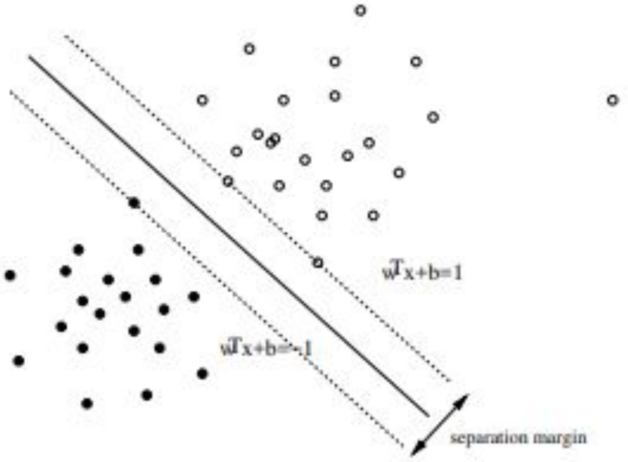
- Given data in n dimension, classified to 2 categories (xi, yi)
- Goal: pattern classification
- SVM: try to find hyperplane, which splits the 2 categories the "best"

Separable case

- The equation of hyperplane $\boldsymbol{w}^T\boldsymbol{x} + b = 0$
- Suppose it is parameterized in a way: $y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1$
- Then the distance between the two categories: 2/|w|
- So it is equivalent if our objective function is $\text{minimise } f(\boldsymbol{w}, b) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}$

Non-separable case

- Here we allow points to be on the other side of the hyperplane
- Assign variables to this and we add penalties to these variables



UPPSALA UNIVERSITET

Optimization problem in matrix form

Non-separable optimisation problem

$$\text{minimise} \quad f(\boldsymbol{\omega}, b) = \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C\sum_{j=1}^{m}\xi_i$$

$$\text{subject to} \quad y_i(\boldsymbol{\omega}^T x_i + b) \geq 1 - \xi_i,\ i = 1...m$$

$$\xi_i \geq 0$$

Optimization variables

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{\omega}^T \\ b \\ \boldsymbol{\xi} \end{bmatrix} (n + 1 + m \times 1)$$

In matrix form

$$\text{minimise} \quad f(\boldsymbol{W}) = \frac{1}{2}\boldsymbol{W}^T\boldsymbol{H}\boldsymbol{W} + \hat{\boldsymbol{f}}^{\boldsymbol{T}}\boldsymbol{W}$$

$$\text{subject to} \quad \boldsymbol{A}\boldsymbol{W} \leq \boldsymbol{R},\ i = 1...m$$

$$\xi_i \geq 0$$

UPPSALA
UNIVERSITET

Optimization problem in matrix form

$$\begin{aligned} \text{minimise} \quad & f(\boldsymbol{W}) = \frac{1}{2}\boldsymbol{W}^T\boldsymbol{H}\boldsymbol{W} + \hat{\boldsymbol{f}}^T\boldsymbol{W} \\ \text{subject to} \quad & \boldsymbol{A}\boldsymbol{W} \leq \boldsymbol{R}, \;\; i = 1...m \\ & \xi_i \geq 0 \end{aligned}$$

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{\omega}^T \\ b \\ \boldsymbol{\xi} \end{bmatrix} (n+1+m \times 1)$$

Objective function:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{I}_{n\times n} & \boldsymbol{Z}_{n\times m+1} \\ \boldsymbol{Z}_{m+1\times n} & \boldsymbol{Z}_{m+1\times m+1} \end{bmatrix} (n+1+m \times n+1+m)$$

$$\hat{\boldsymbol{f}}^T = \begin{bmatrix} 0 & \dots & 0 & 1 & \dots & 1 \end{bmatrix} (1 \times n+1+m)$$

Constraints:

$$\boldsymbol{A} = -\begin{bmatrix} y_1x_{11} & \dots & y_1x_{1n} & y1 & 1 & 0 & \dots & 0 \\ y_2x_{21} & \dots & y_2x_{2n} & y2 & 0 & 1 & \dots & 0 \\ \vdots & \dots & \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ y_mx_{m1} & \dots & y_mx_{mn} & y_m & 0 & 0 & \dots & 1 \end{bmatrix} (n+1+m \times 1)$$

$$\boldsymbol{R} = \begin{bmatrix} -1 \\ \vdots \\ -1 \end{bmatrix} (m \times 1)$$

Code example

Matrices

```
% setting up matrices for A*X <= R
R = -1*ones(m,1);                   % RHS of constraints (m x 1)
X = [xtrain ones(m,1)];             % x data and ones for multiplication
                                    % with b (m x n+1)
Y = ones(n+1,m).*ytrain';           % y_i needs to be multiplied with each
                                    % x_i (n entries) and b (m x n+1)
Y = Y';
A = X.*Y;                           % elementwise multiplication

Cm = eye(m,m);                      % including multiplication with C for
A = -1*[A Cm];                      % epsilon term in A

H = [eye(n),zeros(n,m+1);
     zeros(m+1,n+m+1)];             % constructing H for obj function

f = [zeros(n+1,1); ones(m,1)*C];    % for epsilon term in obj function
lb = [ones(n+1,1)*-inf;zeros(m,1)]; % include lower bound for epsilons
```

Solving

```
[z,fval] = quadprog(H,f,A,R,[],[],lb,[]);

omega = z(1:n);
b = z(n+1);
epsilon = z(n+2:end);
```

UPPSALA
UNIVERSITET

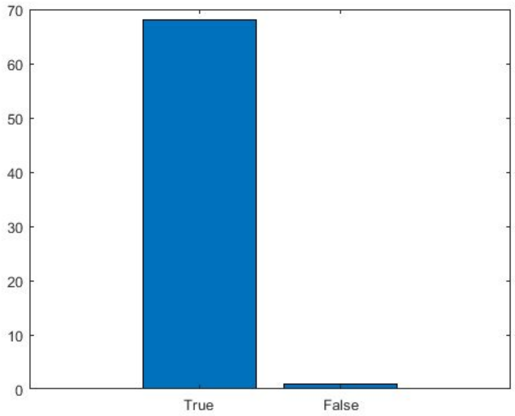
Accuracy, Sensitivity, Specificity

Metrics:

Accuracy = proportion of correct predictions

Sensitivity = proportion of positive diagnoses for patients with disease/malignant cells

Specificity = proportion of negative diagnoses for patients without the disease



For C = 1000, deterministic split into training and test data (500/69)

$$Accuracy = 0.9855$$
$$Sensitivty = 1$$
$$Specificity = 0.9808$$

→ dependent on C and split into training/test data

```
accuracy = true/(true+false);
sensitivity = truePositive/Nmalignant;
specificity = trueNegative/Nbenign;
```

UPPSALA UNIVERSITET

Splitting between test and training data

Deterministic split

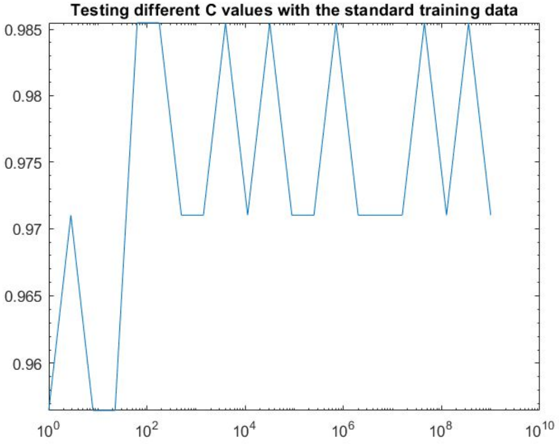
```
xtrain = table2array(data(1:500,3:32));
ytrain = y(1:500);
xtest = table2array(data(501:end,3:32));
ytest = y(501:end);
```

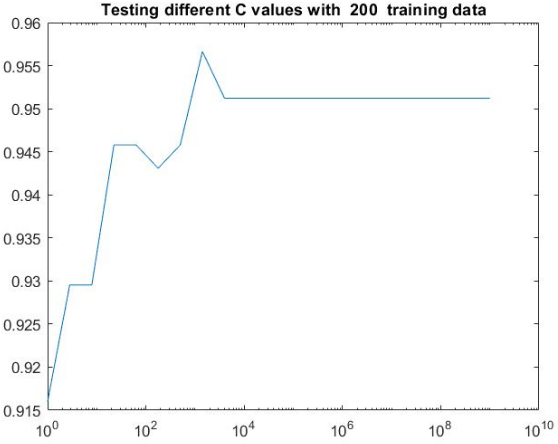
Random split with given ratio

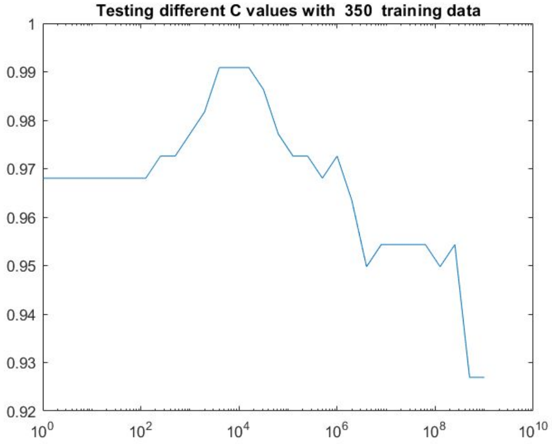
```
m = size(data,1) ;
P = 0.70 ;
idx = randperm(m)  ;
xtrain = table2array(data(idx(1:round(P*m)),3:32));
ytrain = y(idx(1:round(P*m)));
xtest = table2array(data(idx(round(P*m)+1:end),3:32)) ;
ytest = y(idx(round(P*m)+1:end));

m=length(ytrain);
n=30;
```

Testing for different C values - deterministic split

accuracy/C plots





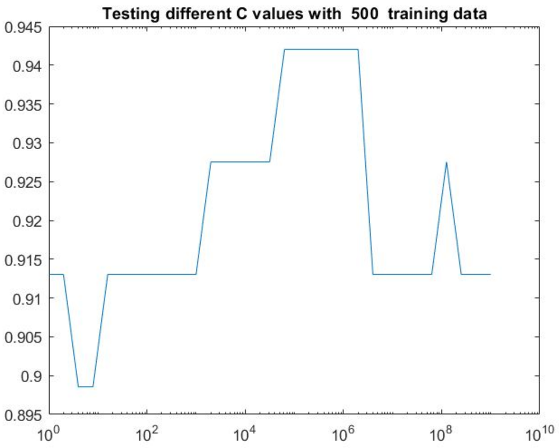


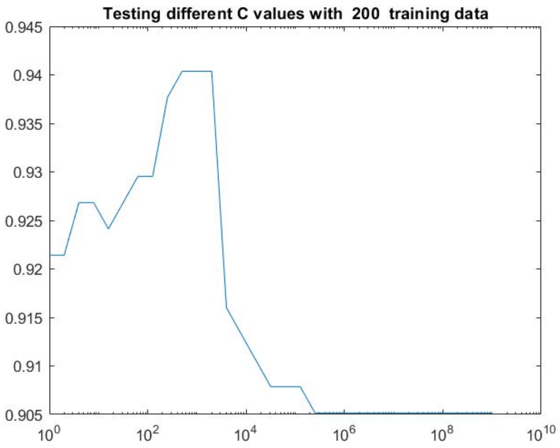
UPPSALA UNIVERSITET

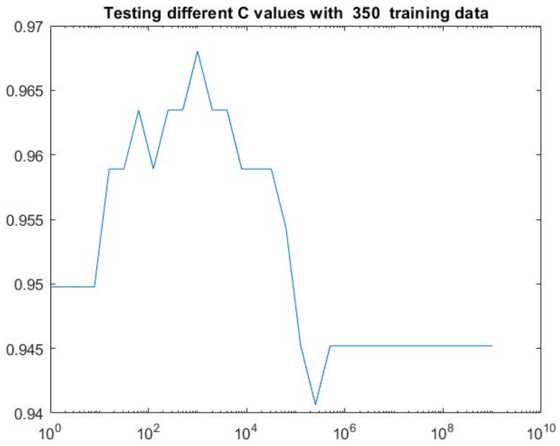
Note: For high C values the optimisation algorithms mostly stops without finding optimal solution

Testing for different C values - random split

accuracy/C plots







UPPSALA UNIVERSITET

Note: For high C values the optimisation algorithms mostly stops without finding optimal solution