Homework 7: Support Vector Machines

ECE 228, Mark Wagner

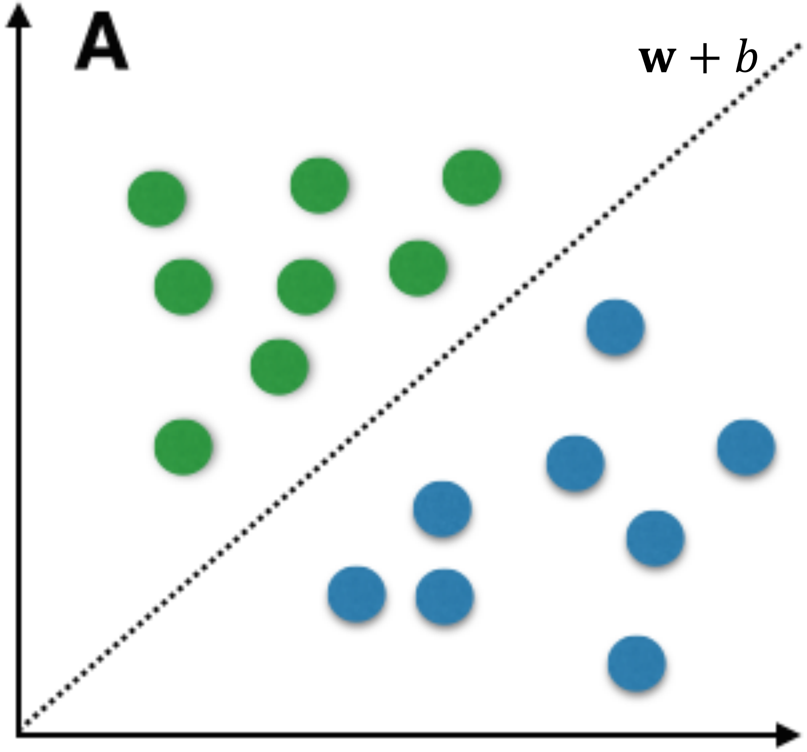
SVM for two Class Classification

Goal: find $\mathbf{w}, b$ such that

$$\mathbf{y} = \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}) + b$$

Where $\mathbf{w}$ is a separating plane
$\phi()$ is an arbitrary transformation
b is a bias term

$\mathbf{y} > 0 \Rightarrow \mathbf{x}$ is class 1
$\mathbf{y} < 0 \Rightarrow \mathbf{x}$ is class 2



$\mathbf{w}$ can be seen as a separating hyperplane

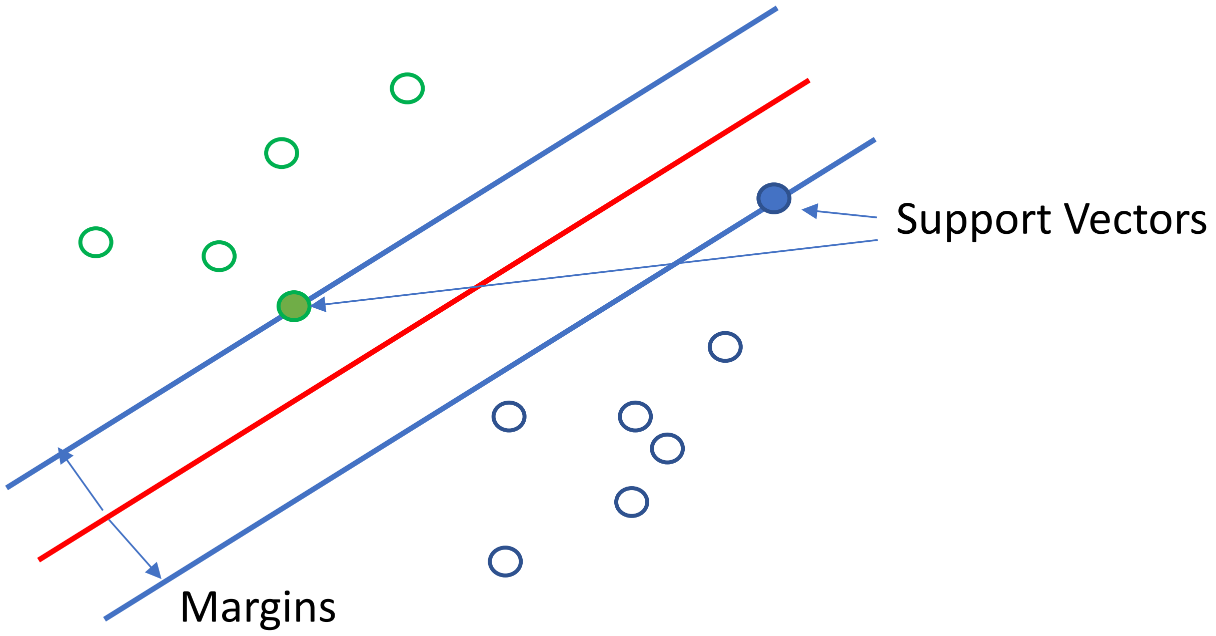
Hard Margin vs. Soft Margin SVM

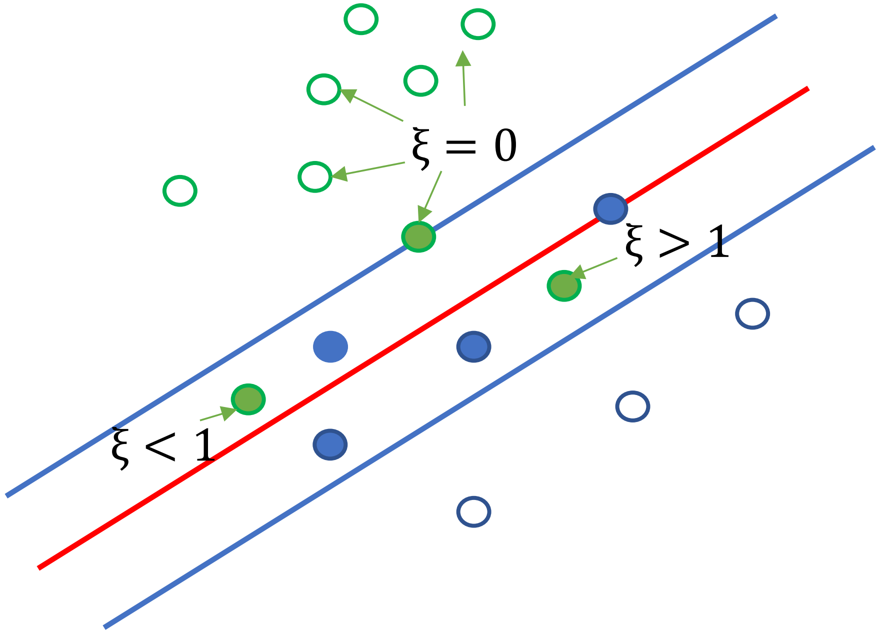
Hard Margin:

- All points correctly classified
- Maximizes margin between separating hyperplane

Soft Margin:

- Allows misclassification
- Minimizes error function based on ξ





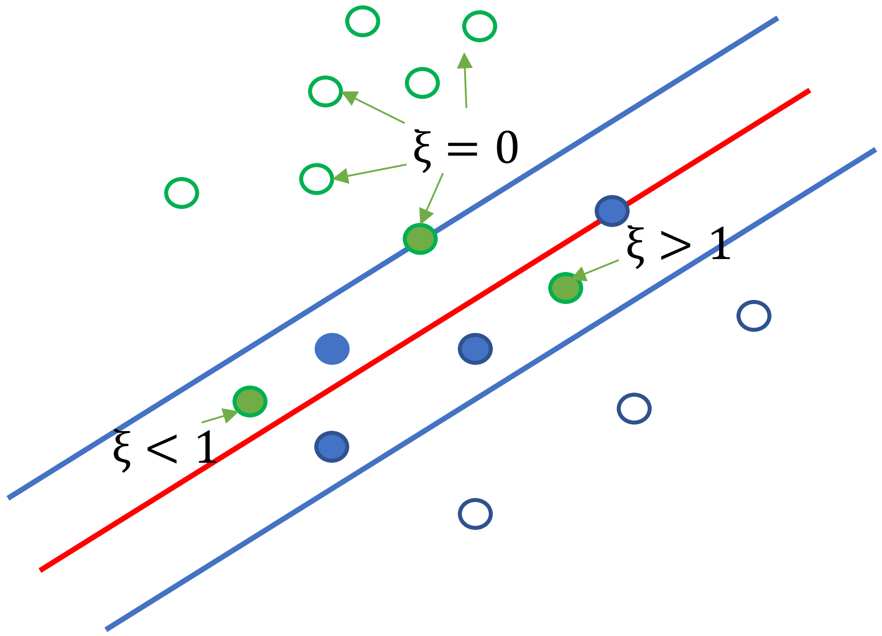
7.1 Soft Margin SVM

minimize

$$C\sum_{n=1}^{N}\xi_n + \frac{1}{2}\|\mathbf{w}\|^2$$

Subject to

$$t_n(\boldsymbol{w}^T\phi(\boldsymbol{x}_n) + b) + \xi \geq 1, \qquad n = 1, \ldots, N$$



Using the Lagrangian dual, equivalent problem is

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}a_n\{t_n y(\mathbf{x}_n) - 1 + \xi_n\} - \sum_{n=1}^{N}\mu_n\xi_n$$

Dual variable **a** introduced

Quadratic programming

Goal: minimize Lagrangian

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N} a_n \{t_n y(\mathbf{x}_n) - 1 + \xi_n\} - \sum_{n=1}^{N}\mu_n\xi_n$$

Setting derivative equal to zero we get

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}(\mathbf{x}_n)$$

$$\frac{\partial L}{\partial b} = 0 \quad \Rightarrow \quad \sum_{n=1}^{N} a_n t_n = 0$$

$$\frac{\partial L}{\partial \xi_n} = 0 \quad \Rightarrow \quad a_n = C - \mu_n.$$

Resulting in dual problem

$$\widetilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

Subject to

$$0 \leqslant a_n \leqslant C$$

$$\sum_{n=1}^{N} a_n t_n = 0$$

Quadratic programming, 7.1

Problem: minimize

$$\widetilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

Can be re-written as:

$$\min_{\mathbf{a}} \mathbf{a}^{\mathrm{T}} \mathbf{K} \mathbf{a} - \mathbf{1}^{\mathrm{T}} \mathbf{a}$$

Subject to

$$0 \leqslant a_n \leqslant C$$

$$\sum_{n=1}^{N} a_n t_n = 0$$

$$\mathbf{K}_{i,j} = t_i t_j \, k(\mathbf{x}_i, \mathbf{x}_j)$$

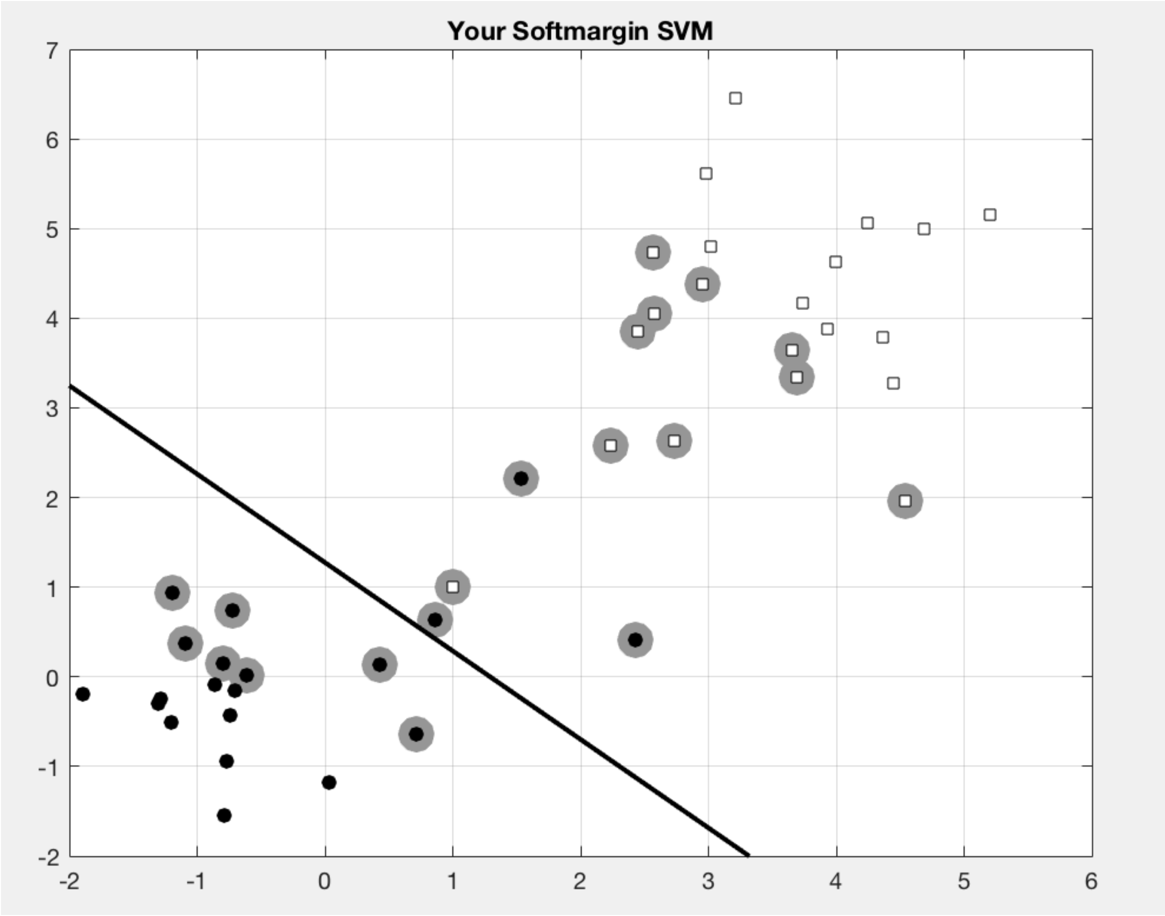
$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle = \mathbf{X}\mathbf{X}^{\mathrm{T}}$$

a represents the margin losses of each point

```
function [a, b] = softsvm(X, t, C)

N = size(X,1);
K = X*X';
H = (t*t').*K;% + 1e-5*eye(N);
f = ones(N,1);
A = [];
b = [];
LB = zeros(N,1);
UB = C*ones(N,1);
Aeq = t';
beq = 0;


alpha = quadprog(H,-f,A,b,Aeq,beq,LB,UB);

    % Compute bias
fout     = K*(alpha.*t);

pos      = find(alpha<C);
bias     = mean(t(pos)-fout(pos));

a = alpha;
b = bias;

end
```

7.1 Solution

```matlab
function [a, b] = softsvm(X, t, C)

N = size(X,1);
K = X*X';
H = (t*t').*K;% + 1e-5*eye(N);
f = ones(N,1);
A = [];
b = [];
LB = zeros(N,1);
UB = C*ones(N,1);
Aeq = t';
beq = 0;


alpha = quadprog(H,-f,A,b,Aeq,beq,LB,UB);

    % Compute bias
fout    = K*(alpha.*t);

pos     = find(alpha<C);
bias    = mean(t(pos)-fout(pos));

a = alpha;
b = bias;

end
```



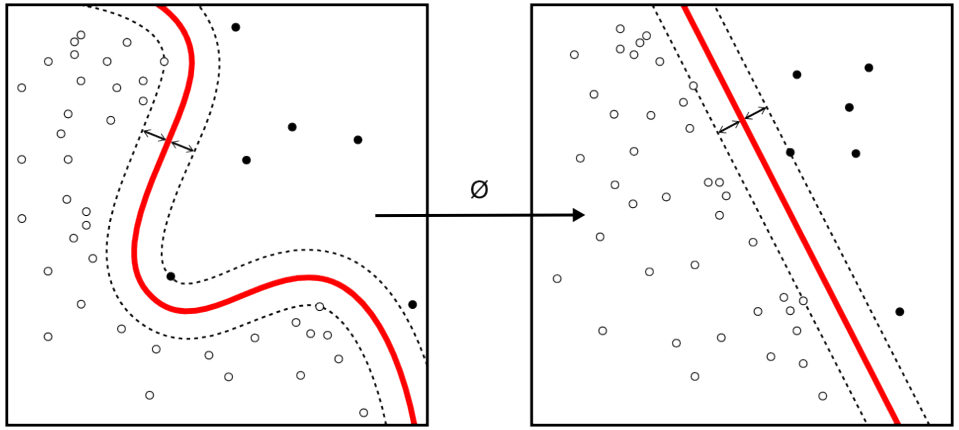
Note: $\mathbf{w} = \frac{1}{N} \sum_n a_n \mathbf{x}_n$

7.2 Kernel Functions

Recall: $$\widetilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

$k(\mathbf{x}_n, \mathbf{x}_m) = \langle \phi(\mathbf{x}_n),\ \phi(\mathbf{x}_m) \rangle$ is known as a **Kernel Function**

$k(\mathbf{x}_n, \mathbf{x}_m)$ can be any function*



Popular choices for $k(\mathbf{x}_n, \mathbf{x}_m)$:

Polynomial (homogeneous) : $k(\mathbf{x}_n, \mathbf{x}_m) = \left(\mathbf{x}_n\ \mathbf{x}_m\right)^d$

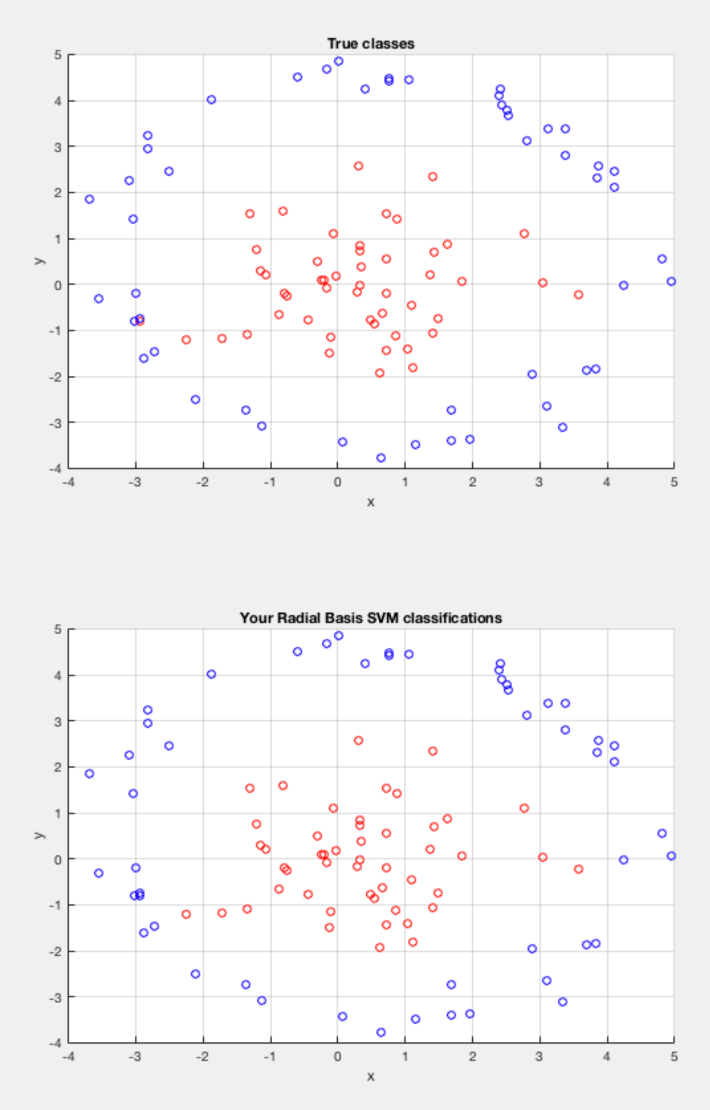
Polynomial (inhomogenious): $k(\mathbf{x}_n, \mathbf{x}_m) = (\mathbf{x}_n\ \mathbf{x}_m + 1)^d$

Gaussian radial : $k(\mathbf{x}_n, \mathbf{x}_m) = \exp(-\gamma||\mathbf{x}_m - \mathbf{x}_n||^2\ )$

Hyperbolic tangent: $k(\mathbf{x}_n, \mathbf{x}_m) = \tanh(\kappa \mathbf{x}_n \mathbf{x}_m + c)$

7.2 Solution

```
function [a, b] = softsvm_RBF(X, t, C)

N = size(X,1);

K   = zeros(N,N);
for j = 1:N
    for i = 1:N
        K(j,i) = exp(-norm(X(j,:)-X(i,:),2).^2);
    end
end

H = (t*t').*K + 1e-5*eye(N);
f = ones(N,1);
A = [];
b = [];
LB = zeros(N,1);
UB = inf(N,1);
Aeq = t';
beq = 0;


UB = C*ones(N,1);
alpha = quadprog(H,-f,A,b,Aeq,beq,LB,UB); % Following

    % Compute bias
fout = sum((alpha.*t)*ones(1,N).*K,1)';
pos = find(alpha<C);
bias = mean(t(pos)-fout(pos));

a = alpha;
b = bias;

end
```



Note: Due to the non-linear transformation on the data from the kernel function, there is no linear hyperplane that can be drawn

Questions