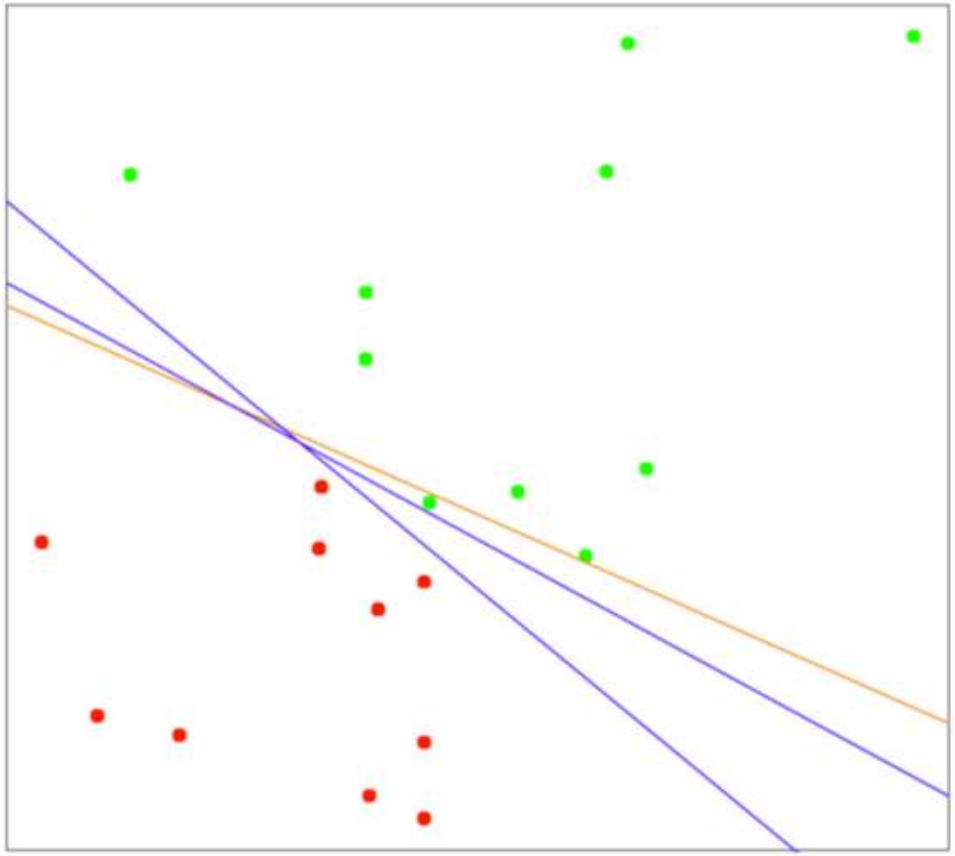
Lecture 11

Support Vector Machine (SVM)
Part 1: Hard Magin

Support Vector Machine

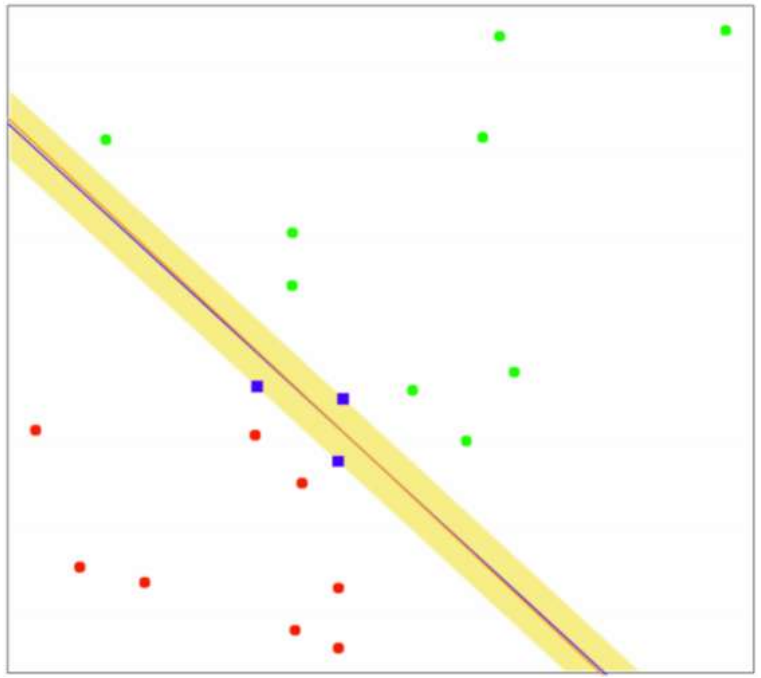
- If the data is linearly separable then there exist infinitely many solutions.
- Which solution is the best?



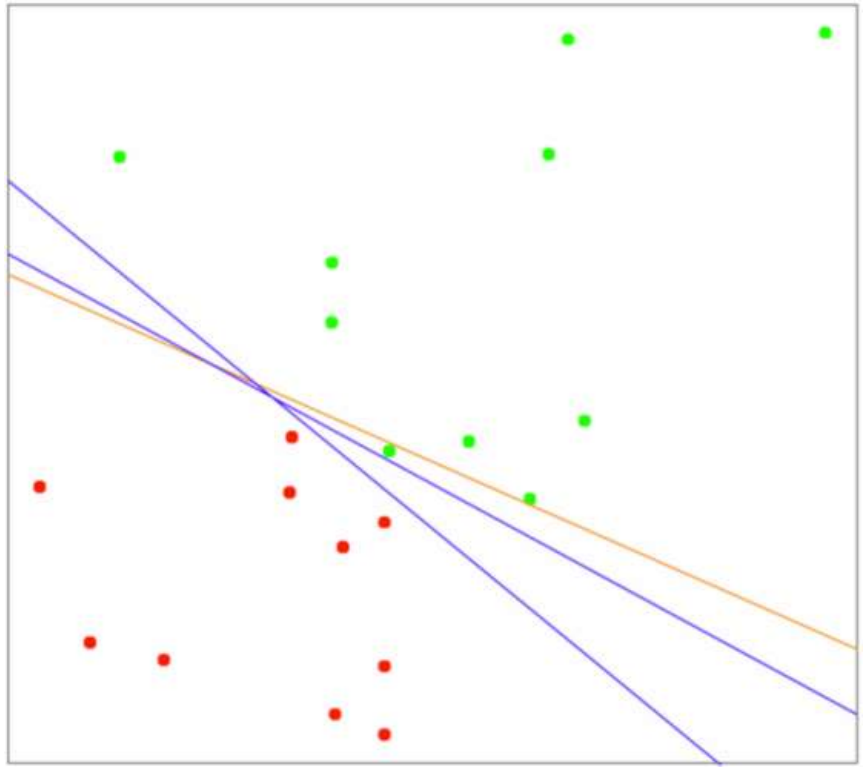
Support Vector Machine

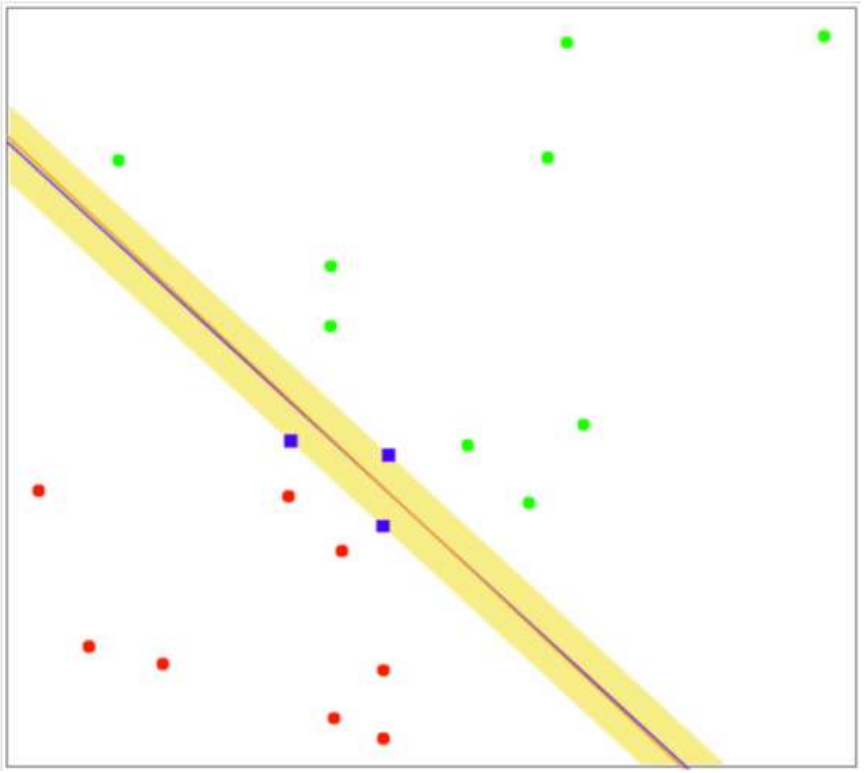
Choose the line farthest from both classes.

i.e. Choose the line which has the maximum distance from the closest point.



Support Vector Machine



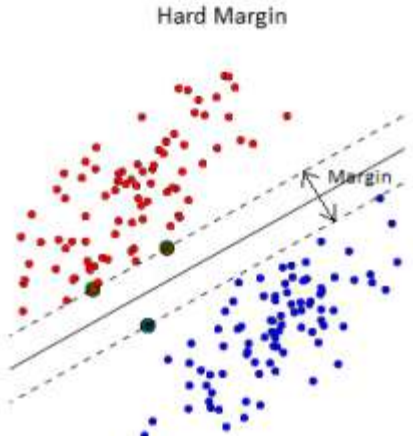


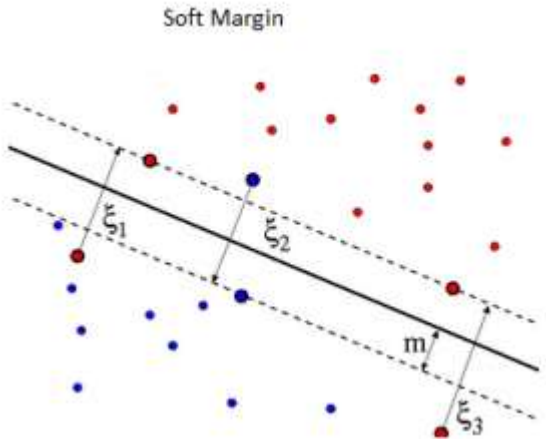
Support Vector Machine (SVM)

The goal is to learn a hyperplane that discriminates between classes; we find a decision boundary which finds the maximum separation between two classes. We consider two cases:

Hard Margin: Classes are linearly separable. We find a hyperplane with margin that maximally separates the two classes.

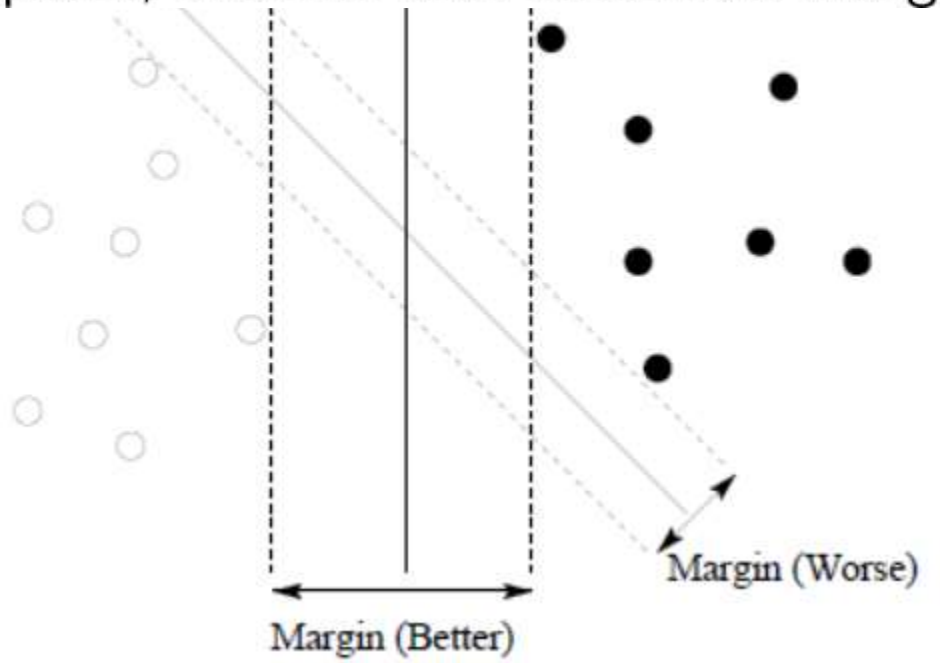
Soft Margin: Classes are not linearly separable; we allow error (slack) in the model fitting and find a decision boundary with the minimum slack required to discriminate between the two classes.





Support Vector Machine

Illustration. If the data is linearly separable, there exists an infinite number of hyperplanes that can separate the two classes. The algorithm looks for the *best* hyperplane; the one with maximum margin.



Support Vector Machine (Hard Margin)

Hyperplane separating the two classes: $\beta^T \mathbf{x} + \beta_0 = 0$

let labels are $y_i \in \{-1, 1\}$

we can classify points as $sign\{d_i\}$ where $d_i = \beta^T \mathbf{x}_i + \beta_0$

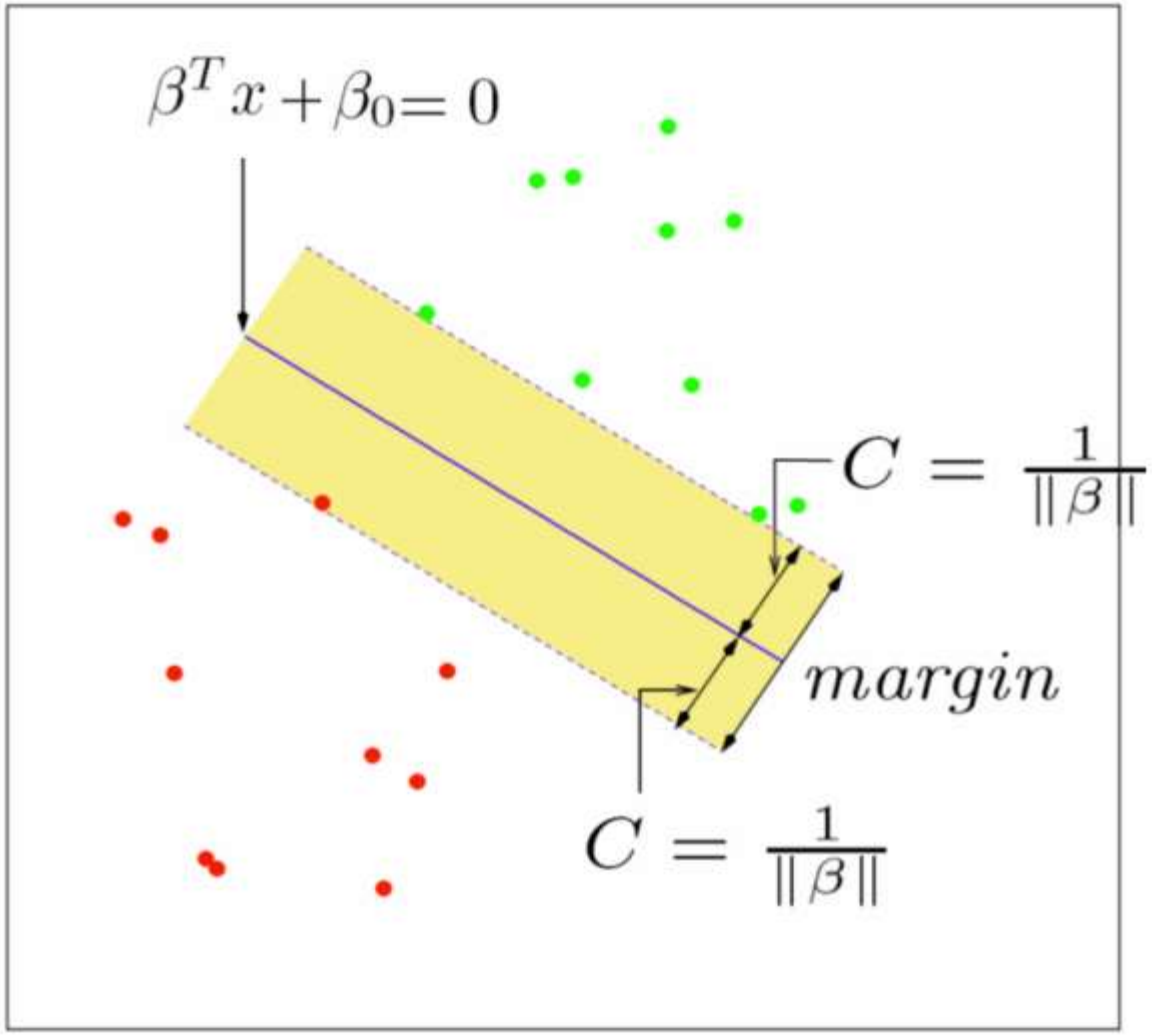
Support Vector Machine

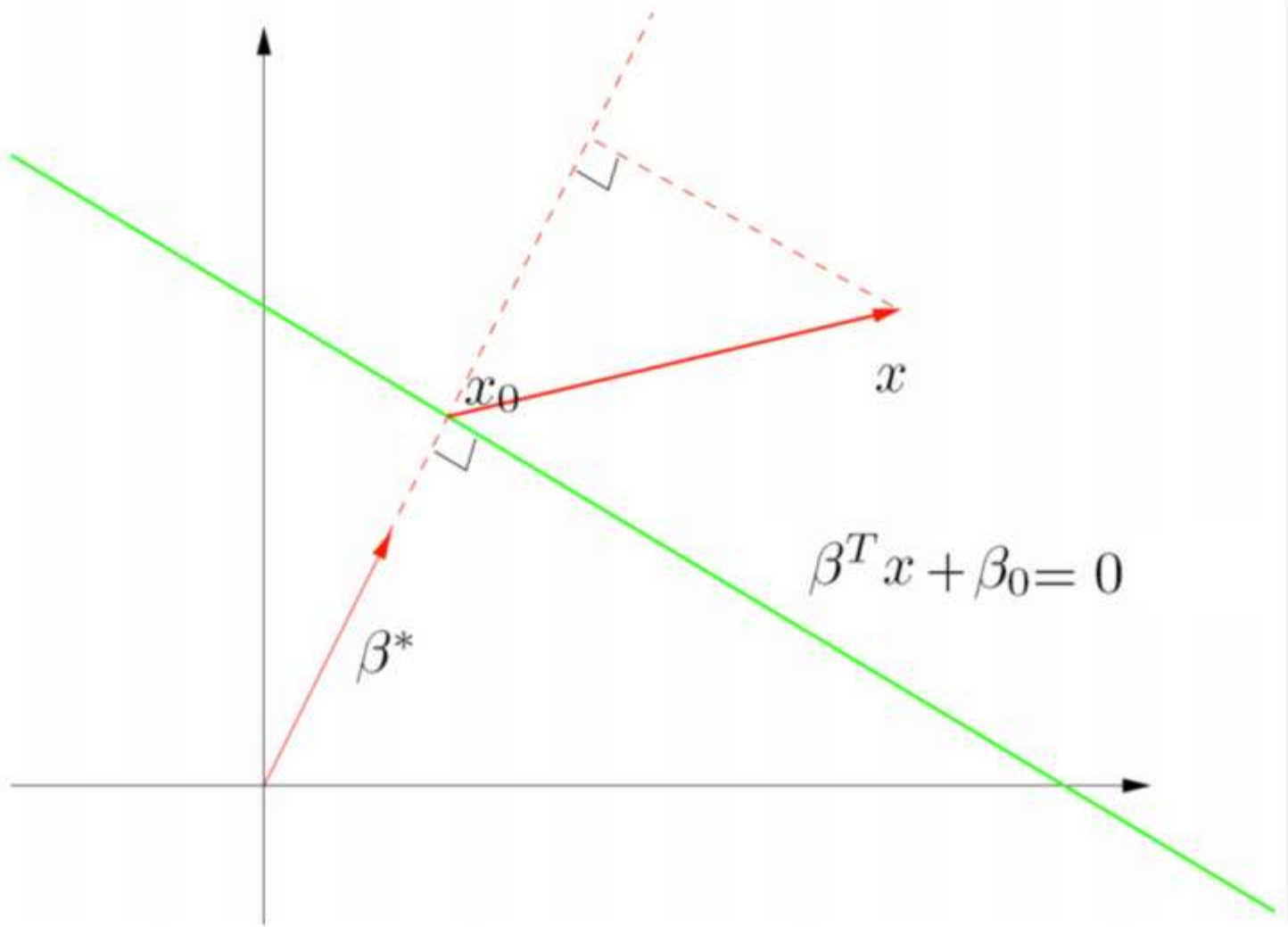
Margin: the distance between the hyperplane and the closest point.

Goal: maximize the margin.

$Margin = min\{y_i d_i\} \qquad i = 1, 2, \ldots, n$
where y_i is label and d_i is distance

Support Vector Machine (Hard Margin)




x
x_0
$\beta^T x + \beta_0 = 0$
β^*

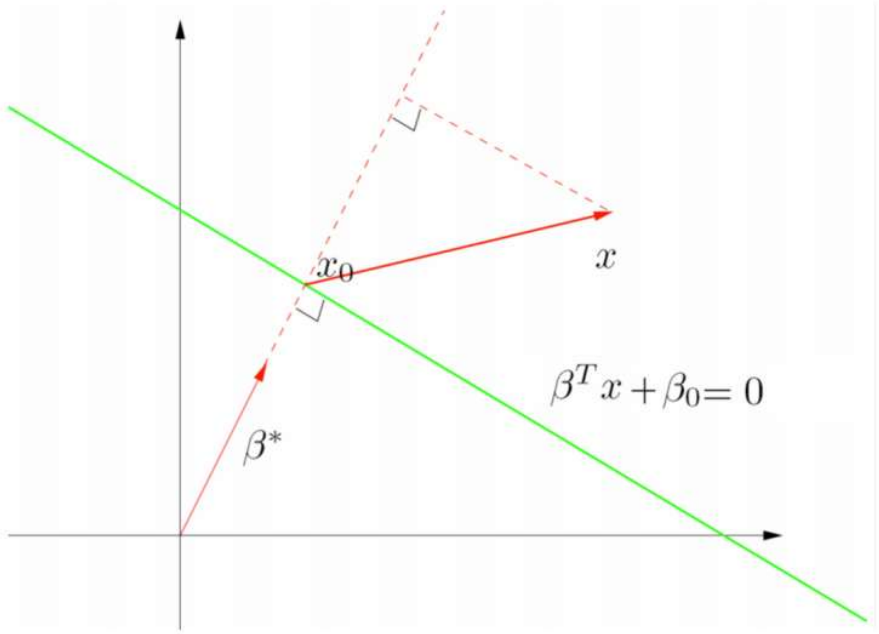
Support Vector Machine (Hard Margin)

$(I)\beta$ is orthogonal to the hyperplane. Consider any x_1 and x_2 not on the hyperplane.

Then

$$\begin{aligned} \beta^T \mathbf{x}_1 + \beta_0 = \beta^T \mathbf{x}_2 + \beta_0 &= 0 \\ \Rightarrow \beta^T (\mathbf{x}_1 - \mathbf{x}_2) &= 0 \end{aligned}$$

$\Rightarrow \beta$ is orthogonal to $(\mathbf{x}_1 - \mathbf{x}_2)$.



Support Vector Machine (Hard Margin)

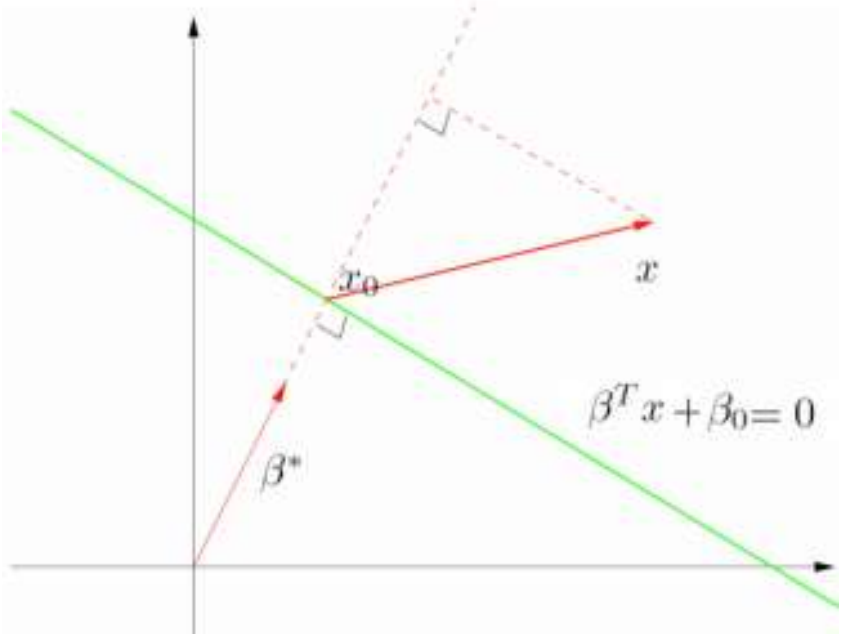
(*II*) for any $\mathbf{x}_0$ on the hyperplane,

$$\beta^T \mathbf{x}_0 + \beta_0 = 0 \Rightarrow \beta^T \mathbf{x}_0 = -\beta_0$$

Support Vector Machine (Hard Margin)

(*III*) The distance d_i is the projection of $(\mathbf{x}_i - \mathbf{x}_0)$ to the direction of β.

$$
\begin{aligned}
d_i &= \frac{\beta^T(\mathbf{x}_i - \mathbf{x}_0)}{\|\beta\|} \\
&= \frac{\beta^T\mathbf{x}_i - \beta^T\mathbf{x}_0}{\|\beta\|} \\
&= \frac{\beta^T\mathbf{x}_i + \beta_0}{\|\beta\|}
\end{aligned}
$$

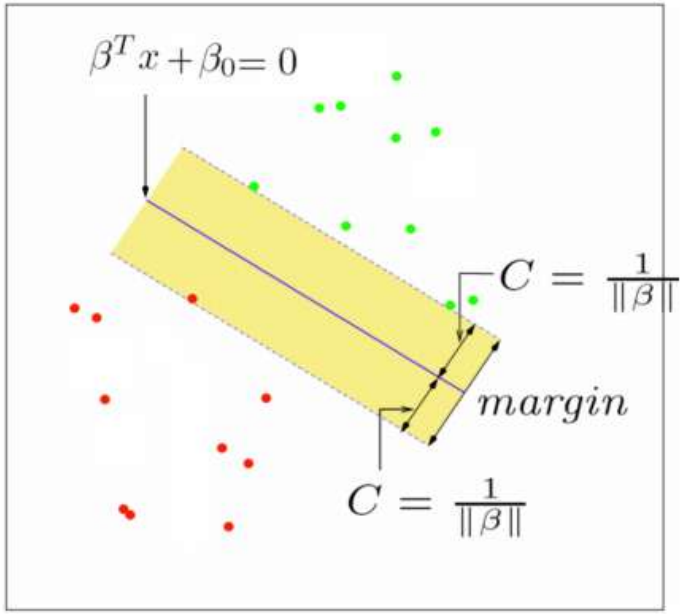


Thus $margin = min\{\frac{y_i(\beta^T\mathbf{x}_i+\beta_0)}{\|\beta\|}\}$

Support Vector Machine (Hard Margin)

(*IV*) If the point is not on the hyperplane then

$$
\begin{aligned}
y_i(\beta^T \mathbf{x}_i + \beta_0) &> 0 \Rightarrow \\
y_i(\beta^T \mathbf{x}_i + \beta_0) &\geq c \Rightarrow \\
y_i(\frac{\beta^T \mathbf{x}_i + \beta_0}{c}) &\geq 1 \Rightarrow
\end{aligned}
$$



There exist β and β_0 such that $y_i(\beta^T \mathbf{x}_i + \beta_0) \geq 1$

Therefore $margin = \frac{1}{\|\beta\|}$

Support Vector Machine (Hard Margin)

The original problem can be reduced now to:

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2$$

subject to

$$y_i(\beta^T \mathbf{x}_i + \beta_0) \geq 1, i = 1, \ldots, n.$$

Support Vector Machine (Hard Margin)

Form the Lagrangian

$$L_p = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{n} \alpha_i [y_i(\beta^T \mathbf{x}_i + \beta_0) - 1]$$

Support Vector Machine (Hard Margin)

Setting the derivative to zero, we obtain:

$$\beta = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

$$0 = \sum_{i=1}^{n} \alpha_i y_i$$

Support Vector Machine (Hard Margin)

Substituting this in the Lagrangian we obtain:

$$L_p = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

subject to:

$$\begin{aligned} \alpha_i &\geq 0 \\ \sum_{i=1}^{n} \alpha_i y_i &= 0 \end{aligned}$$

Support Vector Machine (Hard Margin)

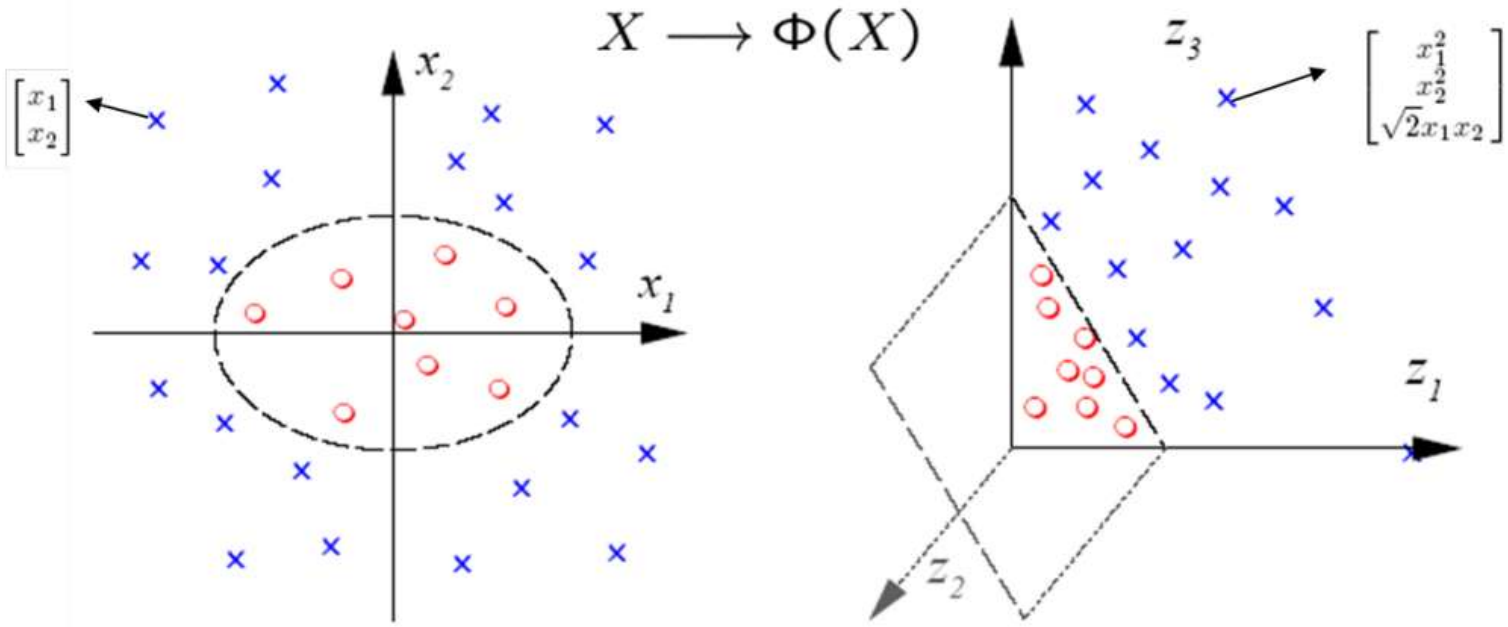
from the complementary slackness (KKT conditions) we have $\alpha_i[y_i(\beta^T \mathbf{x}_i + \beta_0) - 1] = 0$ we can see that

- if $\alpha_i > 0$, then $y_i(\beta^T \mathbf{x}_i + \beta_0) = 1$, i.e. $\mathbf{x}_i$ is on the margin. This point is called support vector.
- if $y_i(\beta^T \mathbf{x}_i + \beta_0) > 1$, $\mathbf{x}_i$ is not on the margin, and $\alpha_i = 0$

Support Vector Machine (Hard Margin)

This is a convex optimization and can be solved by quadratic programming.

Kernel Methods



Two dimensional classification example. A separation in feature space can be found using a linear hyperplane (right). In input space this construction corresponds to a non-linear ellipsoidal decision boundary (left)

(figure from Schölkopf and Smola 2002)

Kernel Methods

Classical Example:

$$\begin{aligned}\langle \phi(X), \phi(X') \rangle &= (x_1^2, x_2^2 \sqrt{2} x_1 x_2)(x_1'^2, x_2'^2 \sqrt{2} x_1' x_2')^T \\ &= \left((x_1, x_2)(x_1', x_2')^T\right)^2 \\ &= \langle x, x' \rangle \\ &= K(X, X')\end{aligned}$$

Common Kernels:

- *Linear* $K_{ij} = \langle X_i, X_j \rangle$
- *Polynomial* $K_{ij} = (1 + \langle X_i, X_j \rangle)^p$
- *Gaussian* $K_{ij} = e^{\frac{-\|X_i - X_j\|^2}{2\sigma^2}}$

Kernel Methods

Classical Example:

$$\begin{aligned}
\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle &= (\mathbf{x}_1^2, \mathbf{x}_2^2 \sqrt{2}\mathbf{x}_1\mathbf{x}_2)(\mathbf{x}_1'^2, \mathbf{x}_2'^2 \sqrt{2}\mathbf{x}_1'\mathbf{x}_2')^T \\
&= \left((\mathbf{x}_1, \mathbf{x}_2)(\mathbf{x}_1', \mathbf{x}_2')^T\right)^2 \\
&= \langle x, x' \rangle \\
&=: K(X, X')
\end{aligned}$$

Common Kernels:

- *Linear* $K_{ij} = \langle X_i, X_j \rangle$
- *Polynomial* $K_{ij} = (1 + \langle X_i, X_j \rangle)^p$
- *Gaussian* $K_{ij} = e^{\frac{-\|X_i - X_j\|^2}{2\sigma^2}}$

Observed and Feature

$$X \rightarrow \phi(X)$$

Observed Space
High-Dimensional
Space

Feature Space

- The coordinates of the points in the feature space are not needed, only their pairwise inner products.
- The pairwise inner products can be computed efficiently directly from the original data items using the kernel function.
- We never compute $\phi(X)$ explicitly.

Example

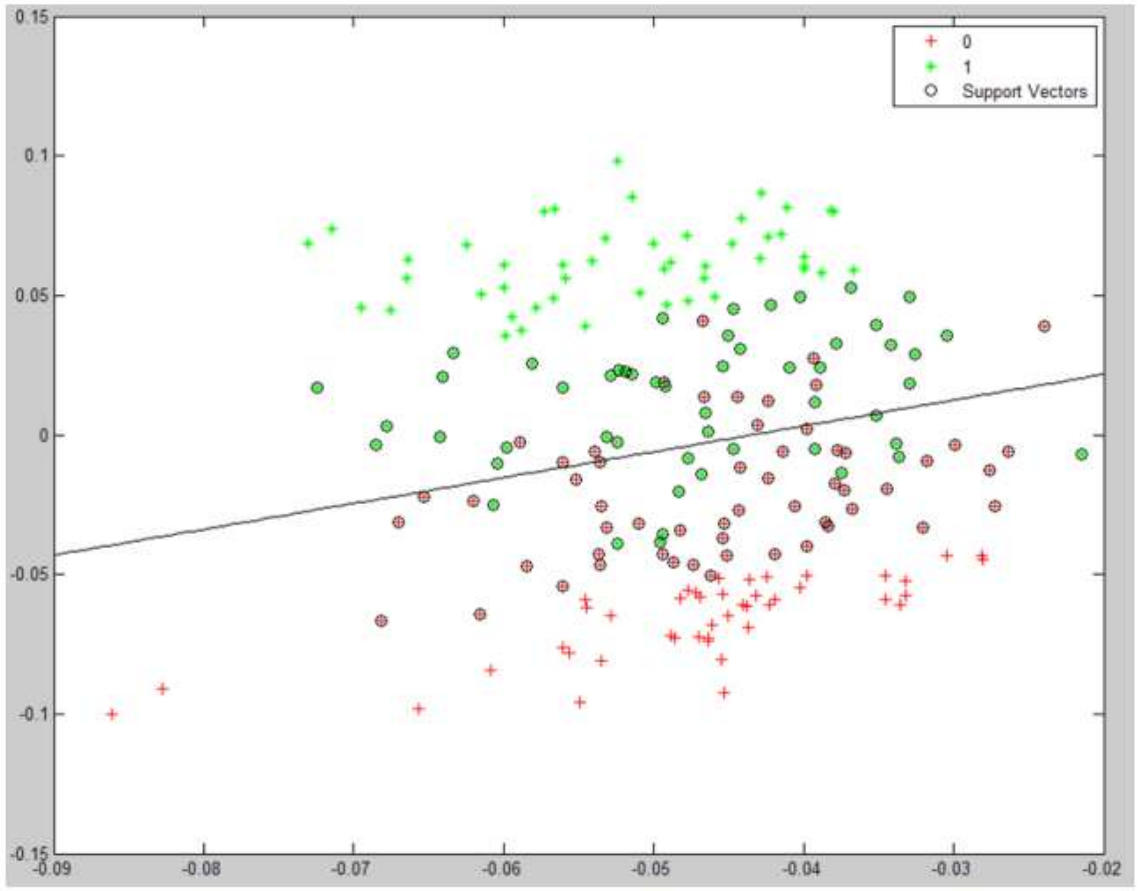
- Load the sample data, which includes digits 2 and 3 of 64 measurements on a sample of 400.
  load2_3.mat
- Use PCA and project data in two-dimensional space.
- Create data
  $data = Y';$
- Create a new vector, groups, to classify data into two groups
  $groups = [zeros(1, 200)ones(1, 200)];$
- Randomly select training and test sets.
  $[train, test] = crossvalind('holdOut', groups);$

Example

- Train an SVM classifier using a linear kernel function and plot the grouped data.

$$svmStruct = svmtrain(data(train,:), groups(train), 'showplot', true);$$

Example

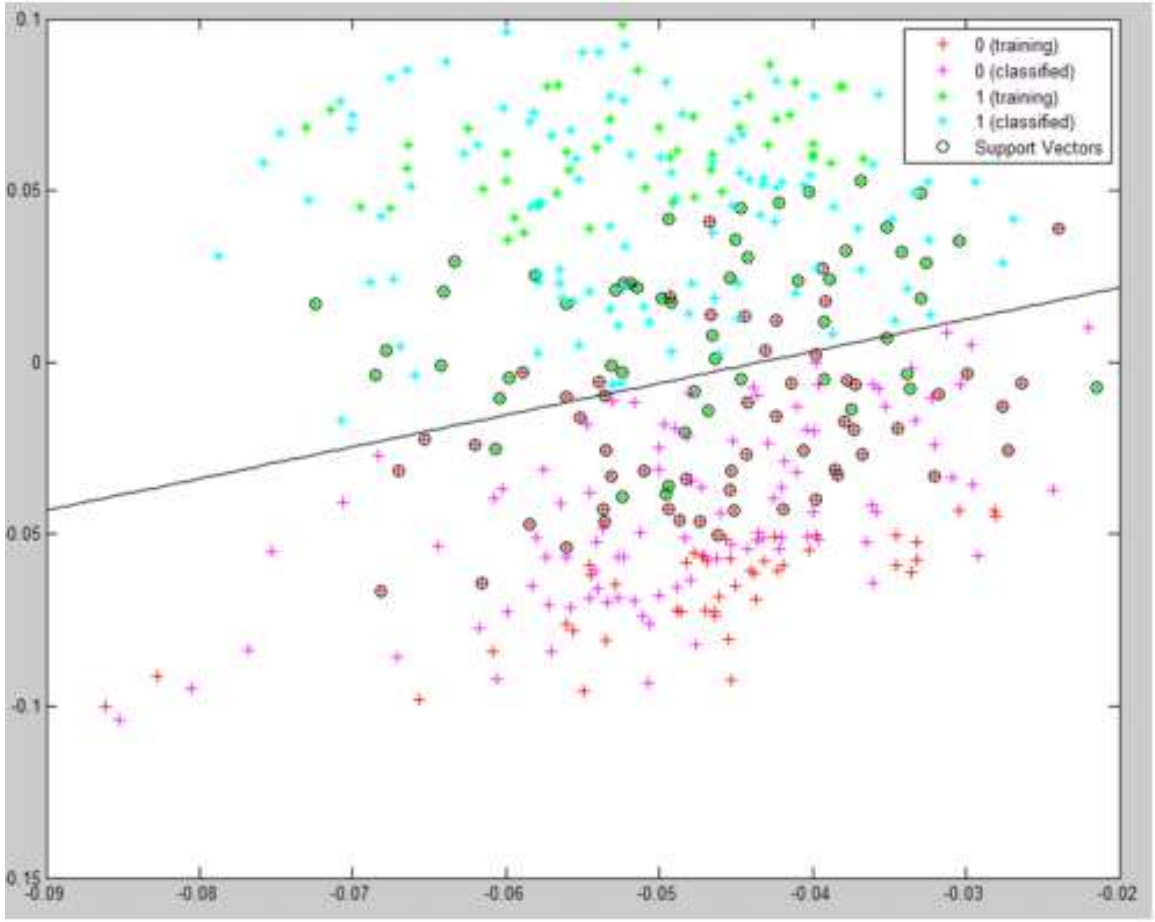

0
1
Support Vectors
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.09
-0.08
-0.07
-0.06
-0.05
-0.04
-0.03
-0.02

Example

- Use the svmclassify function to classify the test set.

$$classes = svmclassify(svmStruct, data(test, :), 'showplot', true);$$

Example

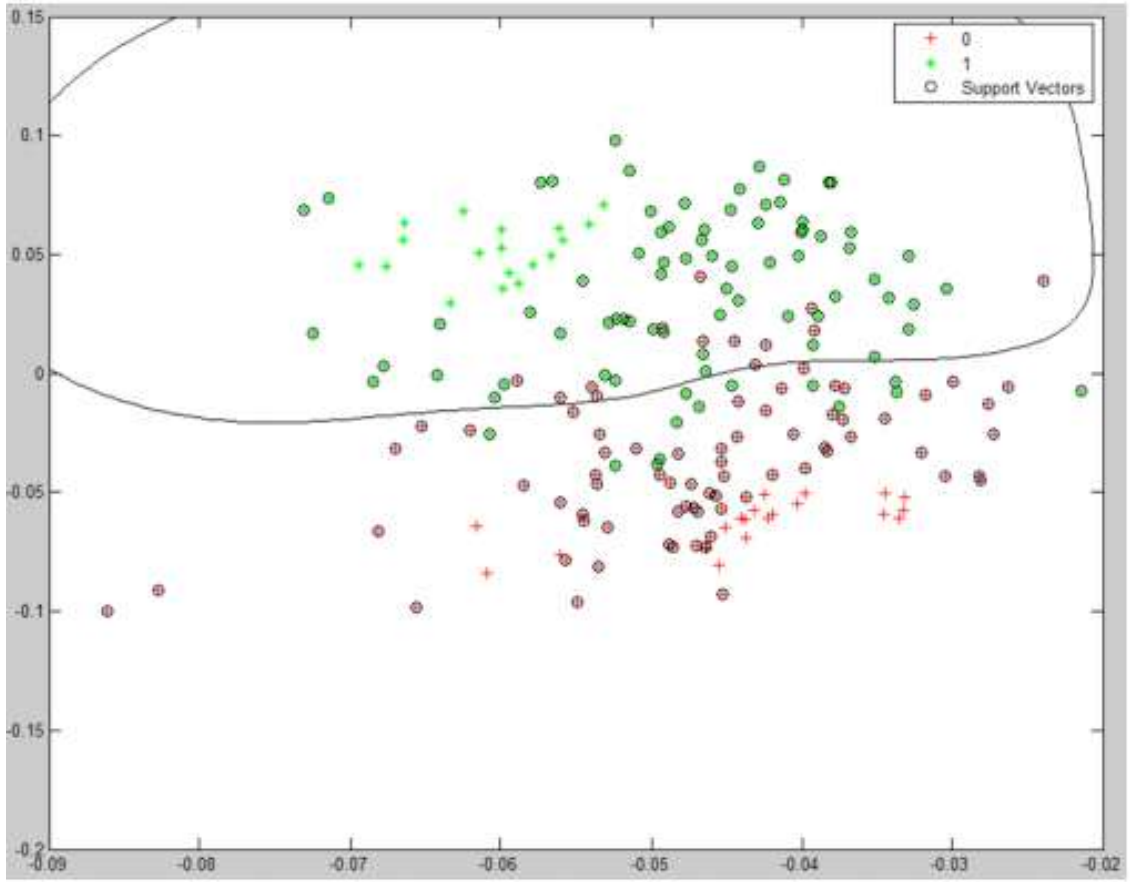

0 (training)
0 (classified)
1 (training)
1 (classified)
Support Vectors
0.1
0.05
0
-0.05
-0.1
-0.15
-0.09
-0.08
-0.07
-0.06
-0.05
-0.04
-0.03
-0.02

Example

- Use Kernel function

$$svmStruct = svmtrain(data(train,:), groups(train),' showplot', true,' Kernel_Function',' rbf')$$

Example

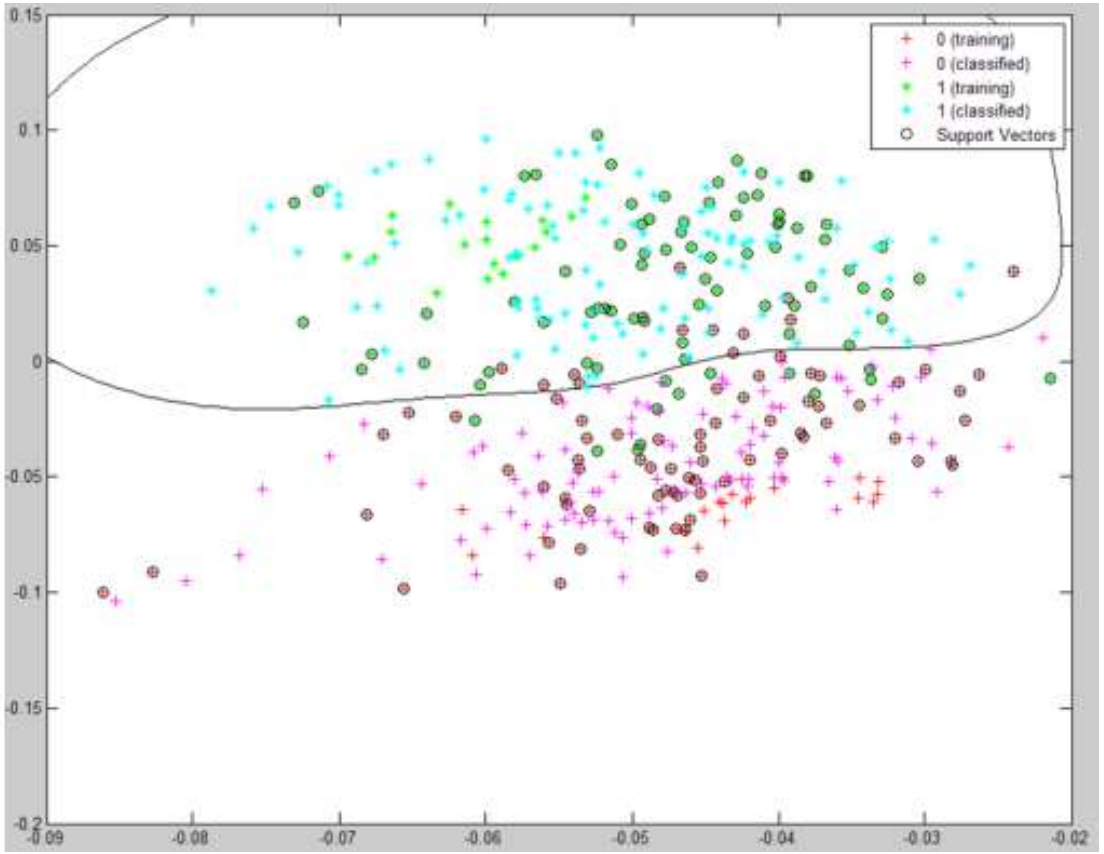

0
1
Support Vectors
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.09
-0.08
-0.07
-0.06
-0.05
-0.04
-0.03
-0.02

Example

- Use the svmclassify function to classify the test set.

$$classes = svmclassify(svmStruct, data(test, :), 'showplot', true);$$

Example


0 (training)
0 (classified)
1 (training)
1 (classified)
Support Vectors
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.09
-0.08
-0.07
-0.06
-0.05
-0.04
-0.03
-0.02