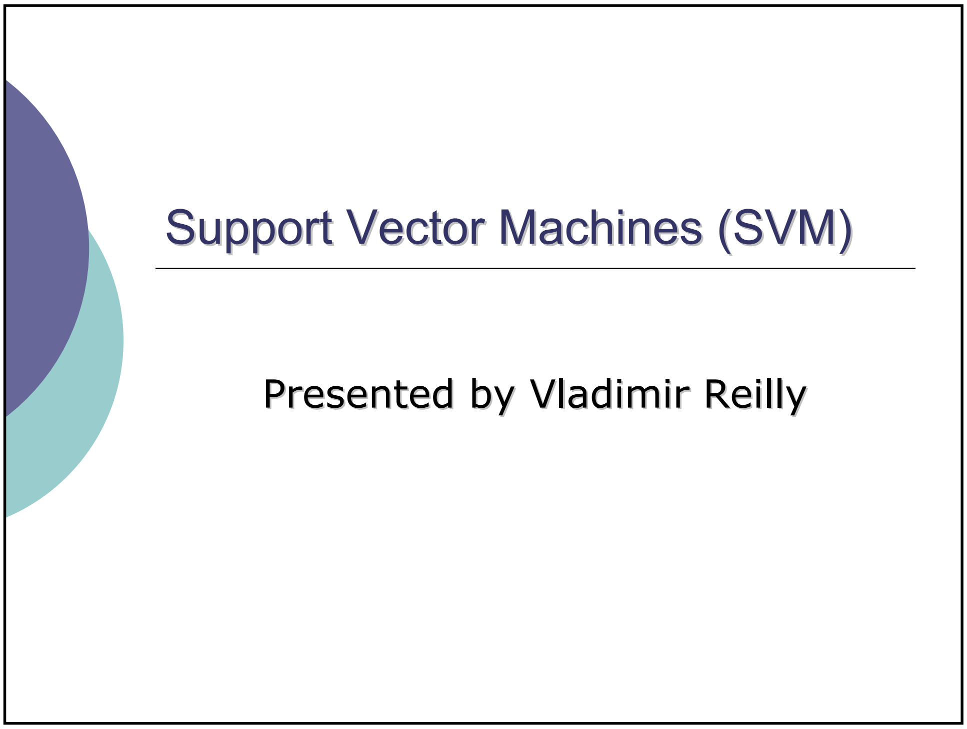
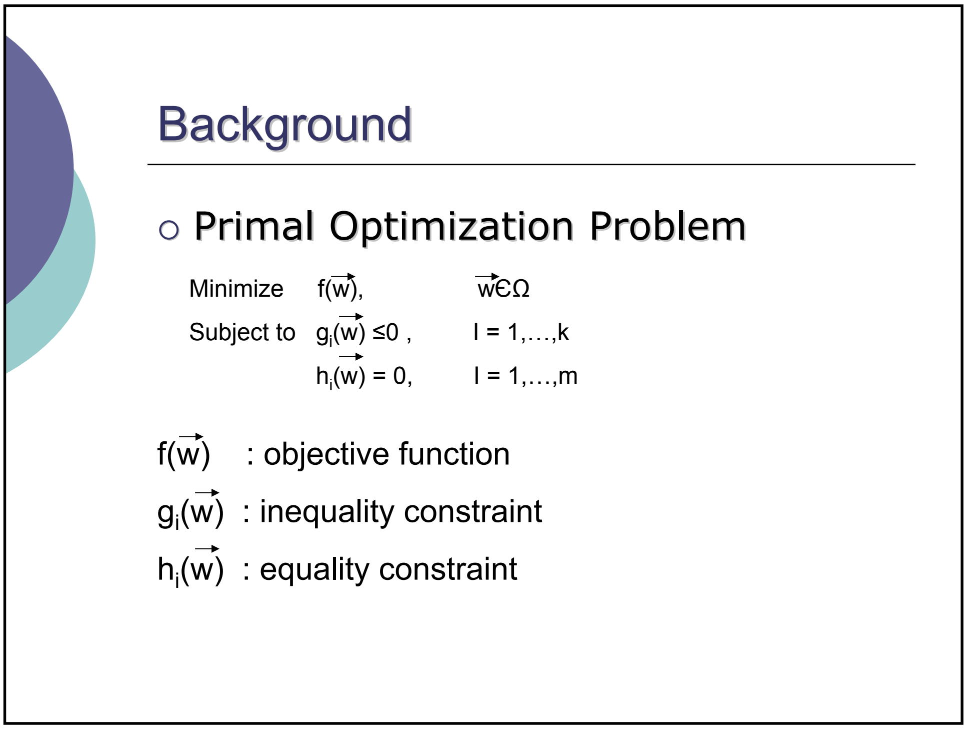


Support Vector Machines (SVM)

Presented by Vladimir Reilly



Background

- Primal Optimization Problem

Minimize	$f(\vec{w})$,	$\vec{w} \in \Omega$
Subject to	$g_i(\vec{w}) \leq 0$,	I = 1,…,k
	$h_i(\vec{w}) = 0$,	I = 1,…,m

$f(\vec{w})$: objective function

$g_i(\vec{w})$: inequality constraint

$h_i(\vec{w})$: equality constraint

Lagrangian

$$
\begin{array}{lll}
\text{Minimize} & f(\vec{w}), & \vec{w} \in \Omega \\
\text{Subject to} & g_i(\vec{w}) \leq 0\,, & I = 1,\ldots,k \\
 & h_i(\vec{w}) = 0, & I = 1,\ldots,m
\end{array}
$$

Generalized Lagrangian function is given by

$$L(\vec{w},\alpha,\beta) = f(\vec{w}) + \sum_{j=1}^{k} \alpha_i g_i(\vec{w}) + \sum_{j=1}^{m} \beta_i h_i(\vec{w}) = f(\vec{w}) + \vec{\alpha}^T \vec{g}(\vec{w}) + \vec{\beta}^T \vec{h}(\vec{w})$$

α and β are the Lagrange multipliers

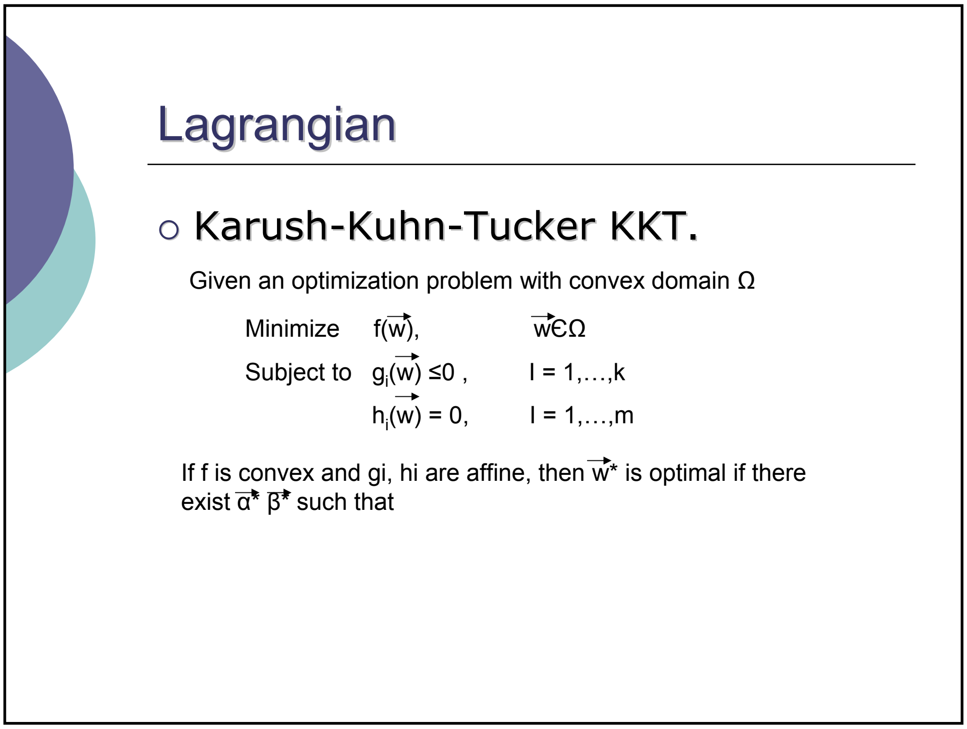
Lagrangian

- Example

Find dimensions of the sides of the box w,u,v, the volume of which is maximal and whose surface is equal to c

$$
\begin{array}{ll}
\text{minimize} & -wuv \\
\text{subject to} & wu + uv + vw = c/2
\end{array}
$$

$$L(w,u,v) = -wuv + \beta(wu + uv + vw - c/2)$$



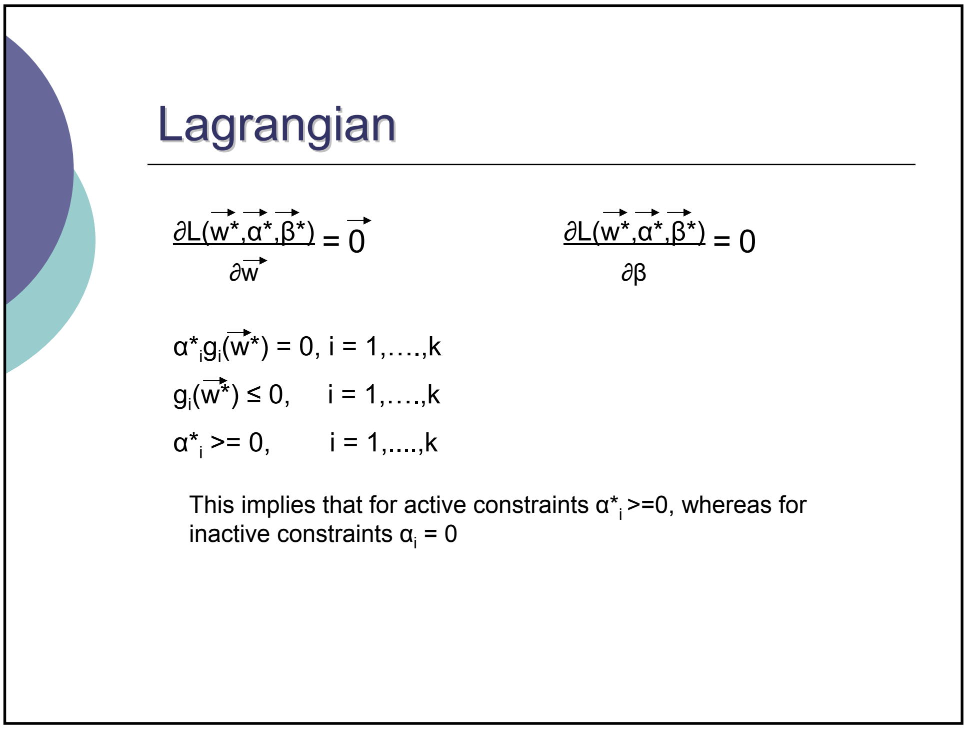
Lagrangian

- Karush-Kuhn-Tucker KKT.

Given an optimization problem with convex domain Ω

Minimize	$f(\vec{w})$,	$\vec{w}$ЄΩ
Subject to	$g_i(\vec{w}) \le 0$,	I = 1,…,k
	$h_i(\vec{w}) = 0$,	I = 1,…,m

If f is convex and gi, hi are affine, then $\vec{w}^*$ is optimal if there exist $\vec{\alpha}^*$ $\vec{\beta}^*$ such that



Lagrangian

$$\frac{\partial L(\vec{w}^*,\vec{\alpha}^*,\vec{\beta}^*)}{\partial \vec{w}} = \vec{0} \qquad \frac{\partial L(\vec{w}^*,\vec{\alpha}^*,\vec{\beta}^*)}{\partial \beta} = 0$$

$\alpha^*_i g_i(\vec{w}^*) = 0$, i = 1,….,k

$g_i(\vec{w}^*) \le 0$, i = 1,….,k

$\alpha^*_i >= 0$, i = 1,....,k

This implies that for active constraints $\alpha^*_i >=0$, whereas for inactive constraints $\alpha_i = 0$

Background

- $g_i(\vec{w}) \leq 0$ is said to be active when $g_i(\vec{w}) = 0$, and is inactive otherwise.

If the objective function and the constraints are linear, the problem is said to be linear.

If the objective function is quadratic, and the constraints are linear, the problem is said to be quadratic.

Lagrangian

- The purpose of the Lagrangian is to convert the optimization problem from the primal form into the dual form.
- Dual form should result in simpler optimization conditions. In the dual form the Lagrangian is expressed as a function of the dual variables (the Lagrange multipliers)

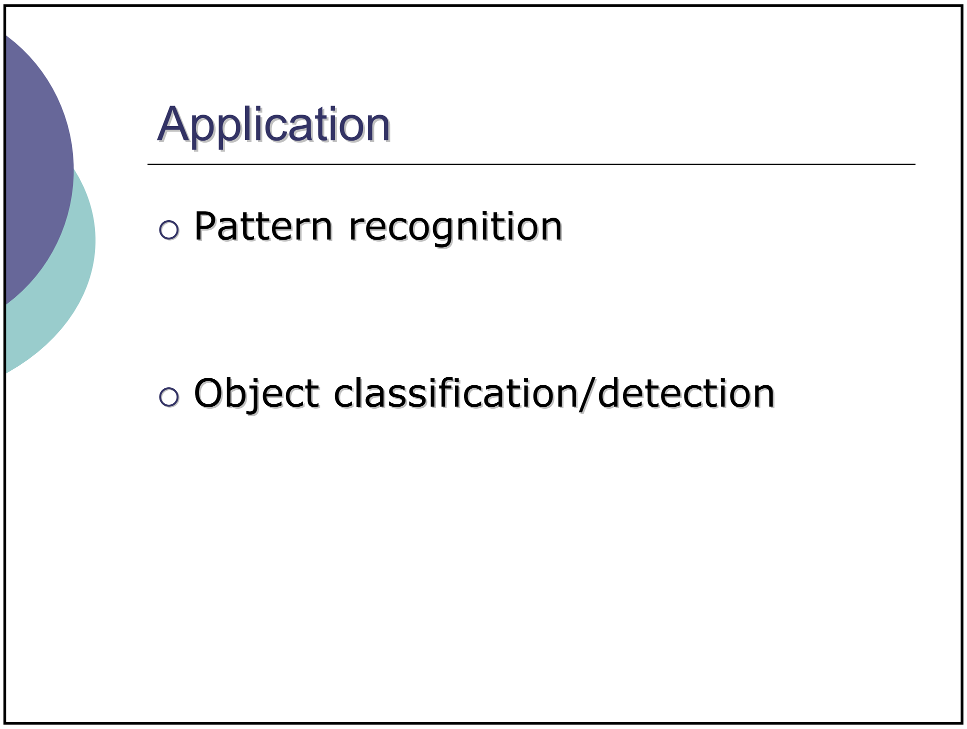
Lagrangian

- Deriving the dual form

1. Construct the Lagrangian of the objective function
2. Take the derivative of L with respect to the primal variables, and set them equal to zero
3. Plug in the resulting relationships back into the Lagrangian and maximize.

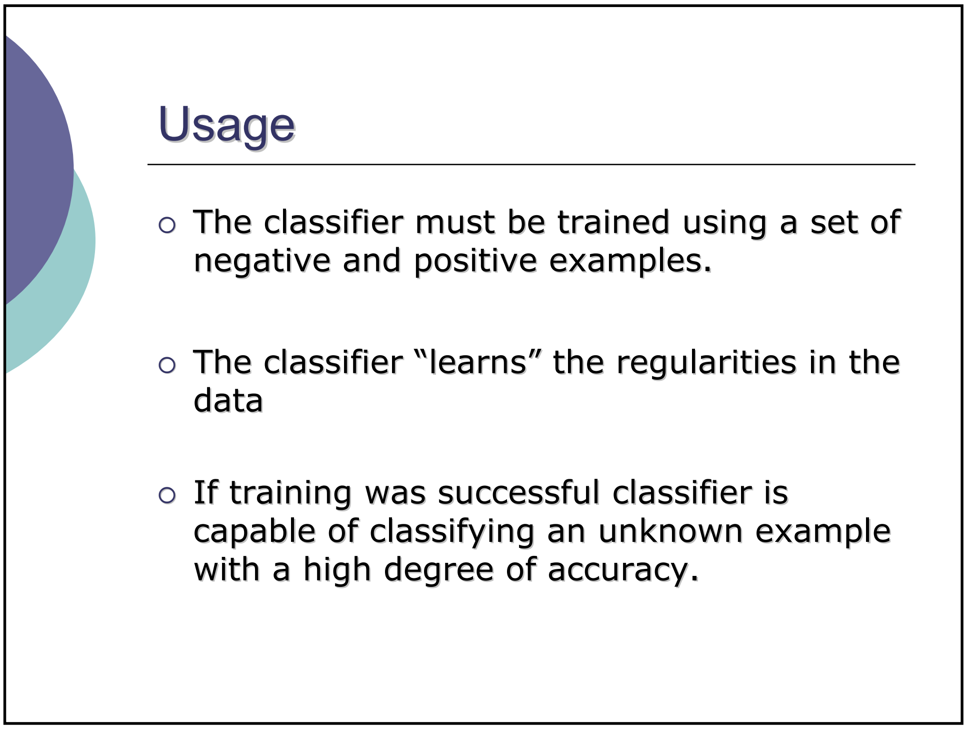
History

- Introduced in 1992 by Vapnik et al.
- Based on Vapnik's structural risk minimization principle (statistical learning theory
- A system for efficiently training linear learning machines.



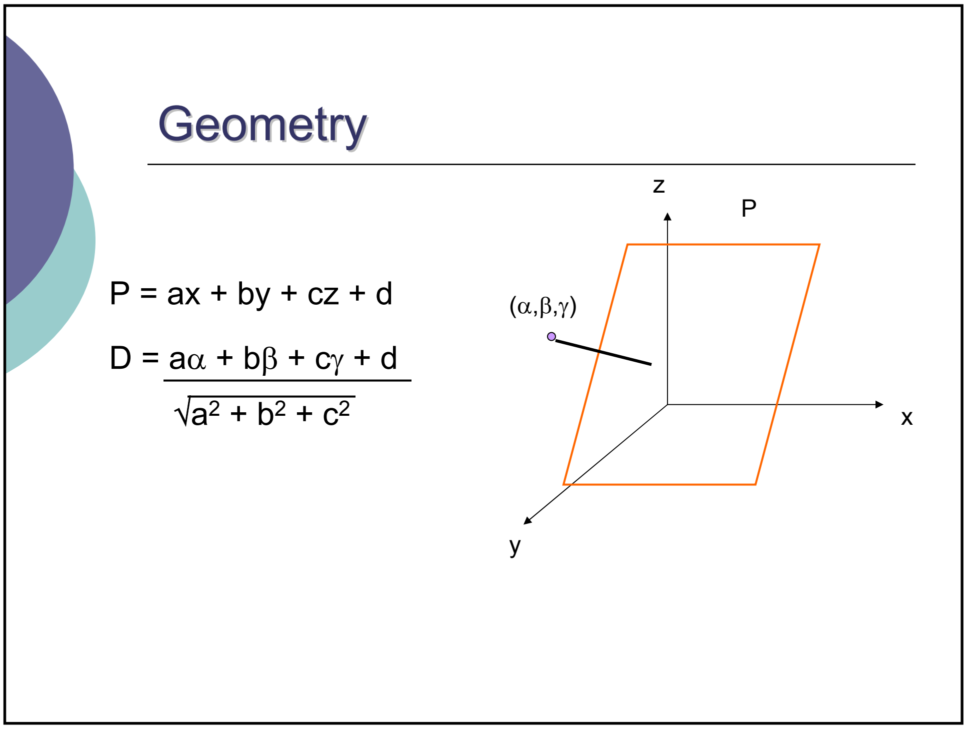
Application

- Pattern recognition
- Object classification/detection



Usage

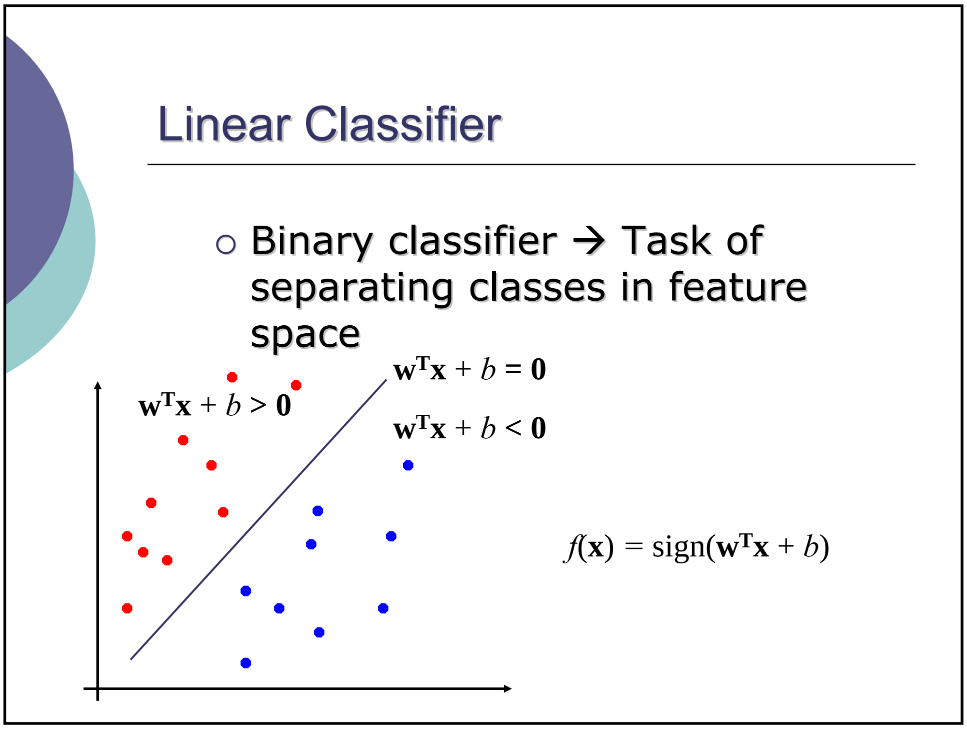
- The classifier must be trained using a set of negative and positive examples.
- The classifier "learns" the regularities in the data
- If training was successful classifier is capable of classifying an unknown example with a high degree of accuracy.



Geometry

$$P = ax + by + cz + d$$

$$D = \frac{a\alpha + b\beta + c\gamma + d}{\sqrt{a^2 + b^2 + c^2}}$$



Linear Classifier

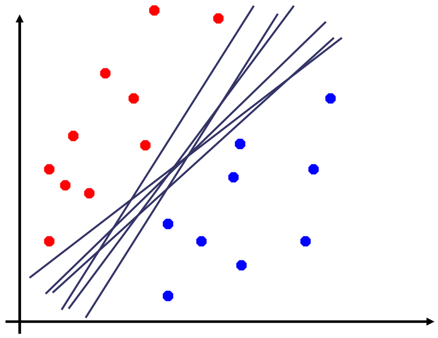
- ○ Binary classifier → Task of separating classes in feature space

$\mathbf{w}^T\mathbf{x} + b = \mathbf{0}$
$\mathbf{w}^T\mathbf{x} + b > \mathbf{0}$
$\mathbf{w}^T\mathbf{x} + b < \mathbf{0}$

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$$

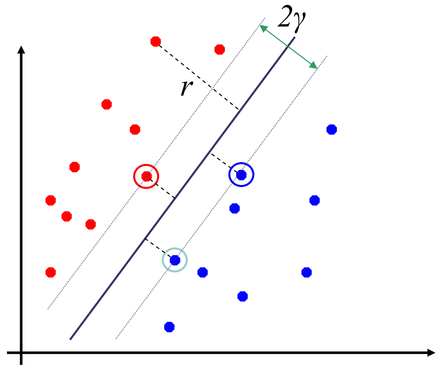
Linear Classifier cont'd

- Which of the linear separators is optimal?



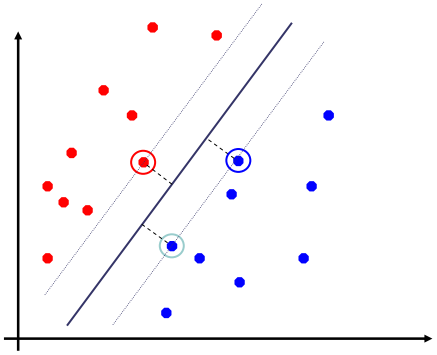
Margin

- Distance from example to the separator is (Point to Plane Distance Equation) $r = \frac{\mathbf{w}^T\mathbf{x}+b}{\|\mathbf{w}\|}$
- Examples closest to the hyperplane are ***support vectors***.
- ***Margin*** 2γ of the separator is the width of separation between classes.



Maximum Margin Classification

- Maximizing the margin is good according to intuition.
- Implies that only support vectors are important; other training examples are ignorable.



Linear SVM

- Fix the output of the decision function to 1, then for a training set $\{(\vec{\mathbf{x}}_{\mathbf{i}}, y_i)\}$

$$\vec{\mathbf{w}}^{\mathbf{T}}\vec{\mathbf{x}}_{\mathbf{i}} + b = 1 \quad \text{if } y_i = 1$$

$$\vec{\mathbf{w}}^{\mathbf{T}}\vec{\mathbf{x}}_{\mathbf{i}} + b = \text{-}1 \quad \text{if } y_i = \text{-}1$$

Linear SVM

- Compute the geometric margin of the resulting classifier given as the distance of each example from the NORMALIZED weight vector.

$$\gamma = \frac{1}{2}\left(<\frac{\vec{\mathbf{w}}}{\|\vec{\mathbf{w}}\|_2} * \vec{\mathbf{x}}^{+}> - <\frac{\vec{\mathbf{w}}}{\|\vec{\mathbf{w}}\|_2} * \vec{\mathbf{x}}^{-}>\right) = \frac{1}{2\|\vec{\mathbf{w}}\|_2}\left(<\vec{\mathbf{w}} * \vec{\mathbf{x}}^{+}> - <\vec{\mathbf{w}} * \vec{\mathbf{x}}^{-}>\right) =$$

$$\gamma = \frac{1}{2\|\vec{\mathbf{w}}\|_2}(1 - b) - (-1 - b) = \frac{1}{2\|\vec{\mathbf{w}}\|_2}(2-b+b) = \frac{1}{\|\vec{\mathbf{w}}\|_2}$$

Linear SVM

- Now we can formulate the quadratic optimization problem as

Given a linearly separable training sample $S = ((\vec{\mathbf{x}}1,y1),\ldots(\vec{\mathbf{x}}l,yl))$,

the hyperplane $(\vec{\mathbf{w}},b)$ that solves the optimization problem

minimizes $<\vec{\mathbf{w}} * \vec{\mathbf{w}}>$

subject to $y_i(<\vec{\mathbf{w}} * \vec{\mathbf{x}}_i> + b) \geq 1 \quad i = 1,\ldots.l$

Linear SVM

minimizes $<\vec{\mathbf{w}} * \vec{\mathbf{w}}>$
subject to $yi(<\vec{\mathbf{w}} * \vec{\mathbf{x}}_i> + b) \geq 1 \quad i = 1,....l$

Convert the problem from the primal form into the dual form

$$L(\vec{\mathbf{w}},b,\vec{\boldsymbol{\alpha}}) = \frac{1}{2}<\vec{\mathbf{w}} * \vec{\mathbf{w}}> - \sum_{i=1}^{l} \alpha_i\, [y_i\, (<\vec{\mathbf{w}} * \vec{\mathbf{x}}_i> + b) - 1]$$

$$\frac{\partial L(\vec{w}^*,\vec{\alpha}^*,\vec{\beta}^*)}{\partial \vec{w}} = \vec{\mathbf{w}} - \sum_{i=1}^{l} y_i\alpha_i\vec{\mathbf{x}}_i = 0 \qquad \frac{\partial L(\vec{\mathbf{w}}^*,\boldsymbol{\alpha}^*,\vec{\boldsymbol{\beta}}^*)}{\partial b} = \sum_{i=1}^{l} y_i\alpha_i = 0$$

$$\vec{\mathbf{w}} = \sum_{i=1}^{l} y_i\alpha_i\mathbf{x}_i \qquad 0 = \sum_{i=1}^{l} y_i\alpha_i$$

Linear SVM

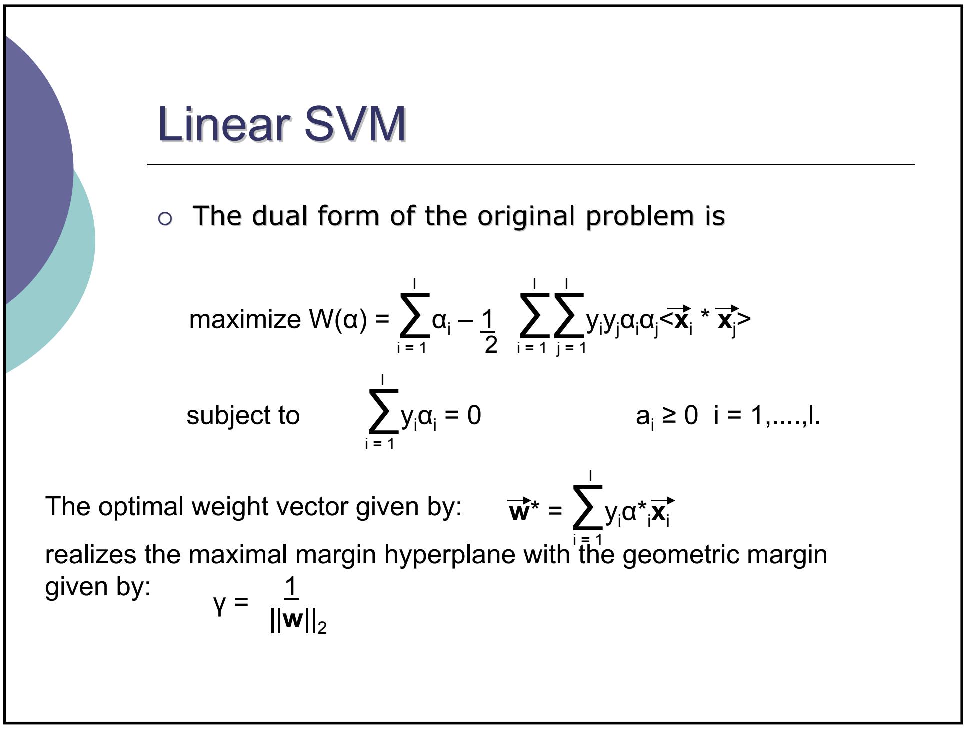
$$\vec{\mathbf{w}} = \sum_{i=1}^{l} y_i\alpha_i\mathbf{x}_i$$

- Now we plug the newly defined $\vec{\mathbf{w}}$ into the $L(\vec{\mathbf{w}},b,\vec{\boldsymbol{\alpha}})$

$$L(\vec{\mathbf{w}},b,\vec{\boldsymbol{\alpha}}) = \frac{1}{2}<\vec{\mathbf{w}} * \vec{\mathbf{w}}> - \sum_{i=1}^{l} \alpha_i\, [y_i\, (<\vec{\mathbf{w}} * \vec{\mathbf{x}}_i> + b) - 1] =$$

$$\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_iy_j\alpha_i\alpha_j<\vec{\mathbf{x}}_i * \vec{\mathbf{x}}_j> - \sum_{i=1}^{l}\sum_{j=1}^{l} y_iy_j\alpha_i\alpha_j<\vec{\mathbf{x}}_i * \vec{\mathbf{x}}_j> + \sum_{j=1}^{l}\alpha_i =$$

$$\sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_iy_j\alpha_i\alpha_j<\vec{\mathbf{x}}_i * \vec{\mathbf{x}}_j>$$



Linear SVM

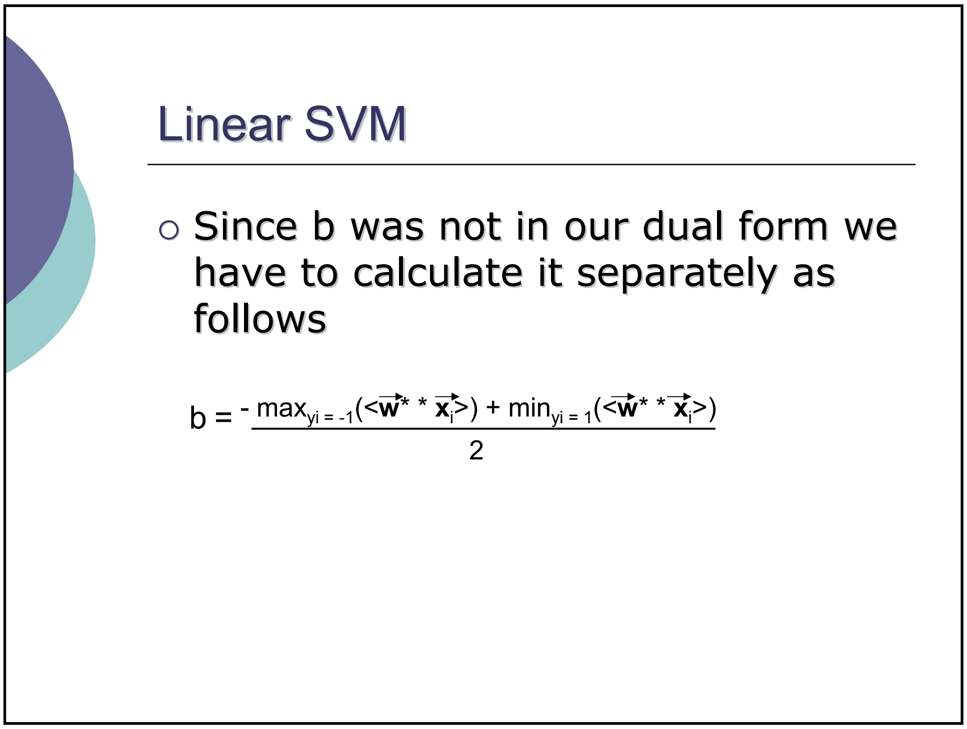
- The dual form of the original problem is

$$\text{maximize } W(\alpha) = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \vec{\mathbf{x}}_i * \vec{\mathbf{x}}_j \rangle$$

$$\text{subject to } \sum_{i=1}^{l} y_i \alpha_i = 0 \qquad a_i \geq 0 \;\; i = 1,....,l.$$

The optimal weight vector given by: $\vec{\mathbf{w}}^* = \sum_{i=1}^{l} y_i \alpha^*_i \vec{\mathbf{x}}_i$

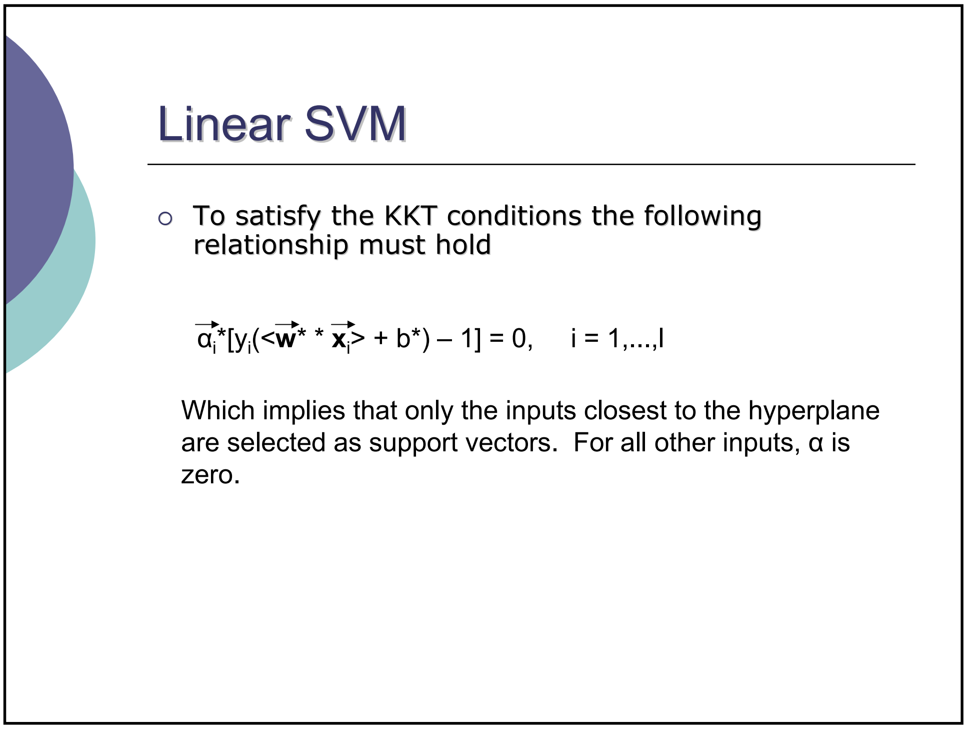
realizes the maximal margin hyperplane with the geometric margin given by: $\gamma = \dfrac{1}{||\mathbf{w}||_2}$



Linear SVM

- Since b was not in our dual form we have to calculate it separately as follows

$$b = \frac{-\max_{y_i = -1}(\langle \vec{\mathbf{w}}^* * \vec{\mathbf{x}}_i \rangle) + \min_{y_i = 1}(\langle \vec{\mathbf{w}}^* * \vec{\mathbf{x}}_i \rangle)}{2}$$

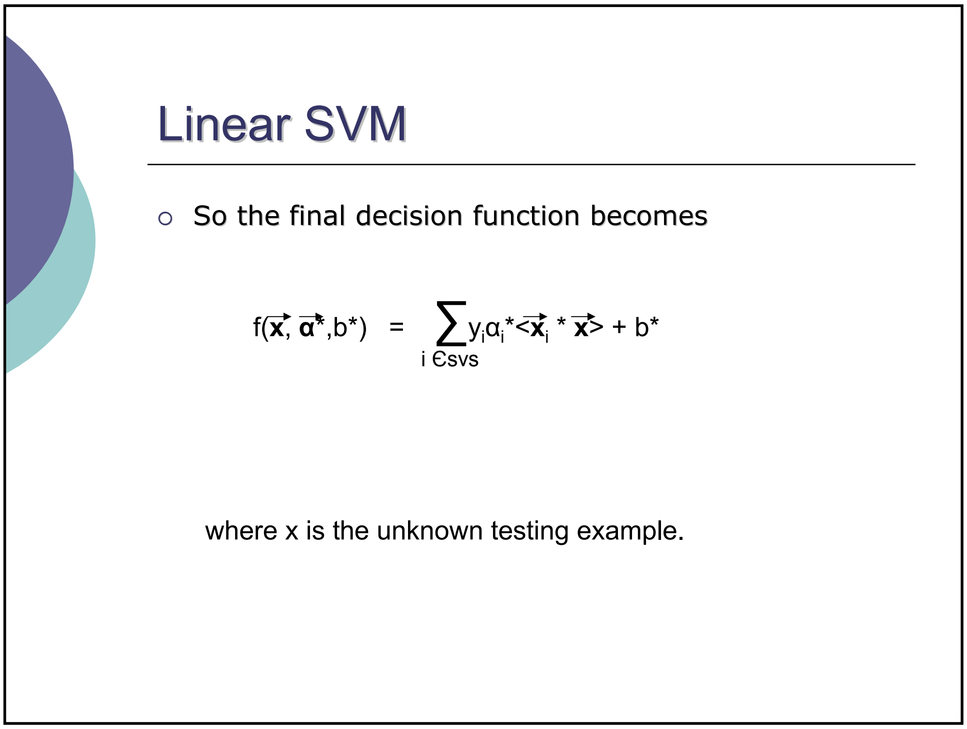


Linear SVM

- To satisfy the KKT conditions the following relationship must hold

$$\vec{\alpha}_i^*[y_i(<\vec{\mathbf{w}}^* * \vec{\mathbf{x}}_i> + b^*) - 1] = 0, \quad i = 1,\dots,l$$

Which implies that only the inputs closest to the hyperplane are selected as support vectors. For all other inputs, α is zero.



Linear SVM

- So the final decision function becomes

$$f(\vec{\mathbf{x}}, \vec{\boldsymbol{\alpha}}^*, b^*) \;=\; \sum_{i \in svs} y_i\alpha_i^* <\vec{\mathbf{x}}_i * \vec{\mathbf{x}}> + b^*$$

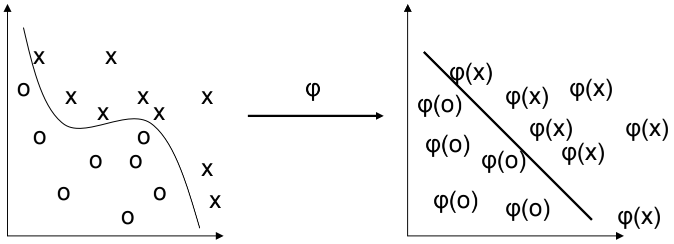
where x is the unknown testing example.

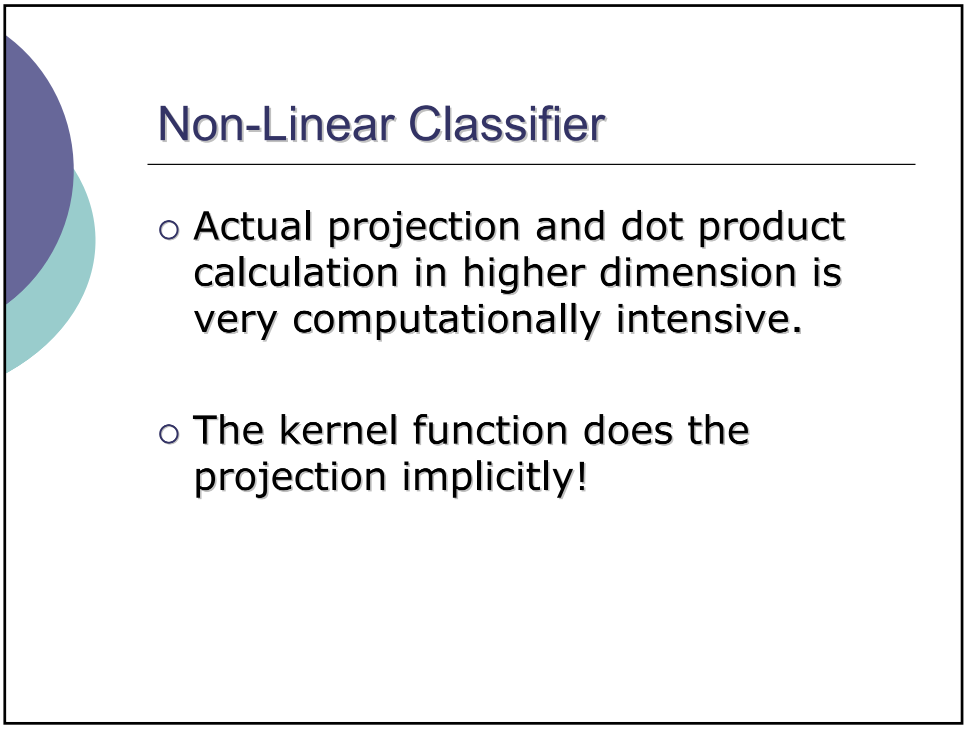
Non-linear classifier

- To allow the SVM to estimate non-linear functions, training examples are projected into higher dimensional space using the Kernel trick.
- In that space the examples will hopefully be linearly separable.

Non-linear Classifier

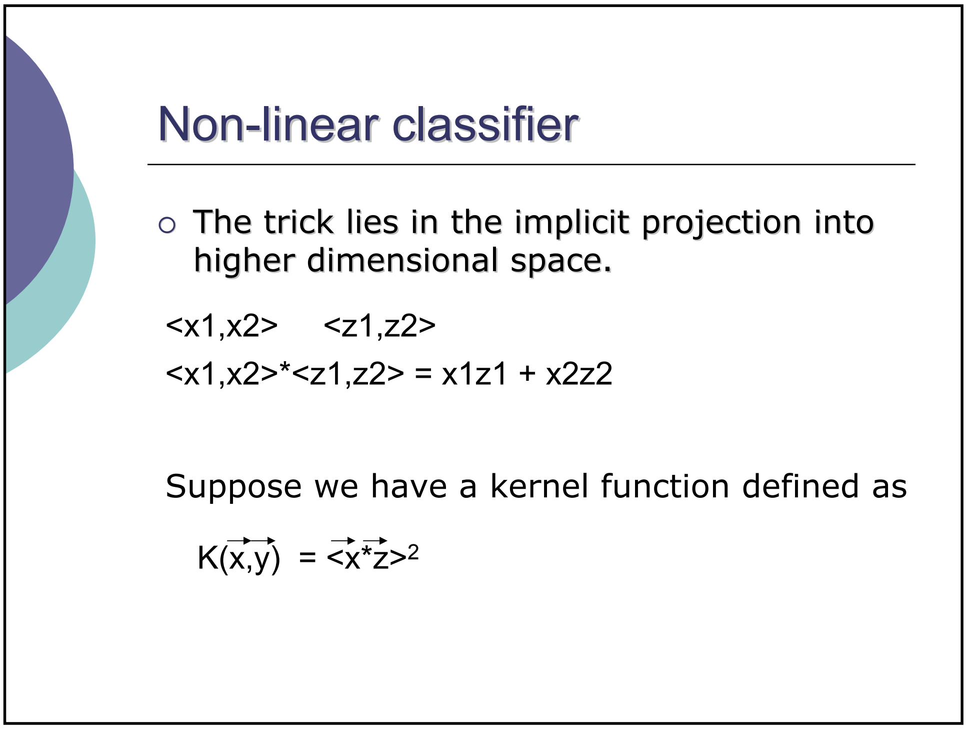
- $K(\vec{x},\vec{z}) = <\varphi(\vec{x})*\varphi(\vec{z})>$ x & z $\in Y$





Non-Linear Classifier

- Actual projection and dot product calculation in higher dimension is very computationally intensive.
- The kernel function does the projection implicitly!



Non-linear classifier

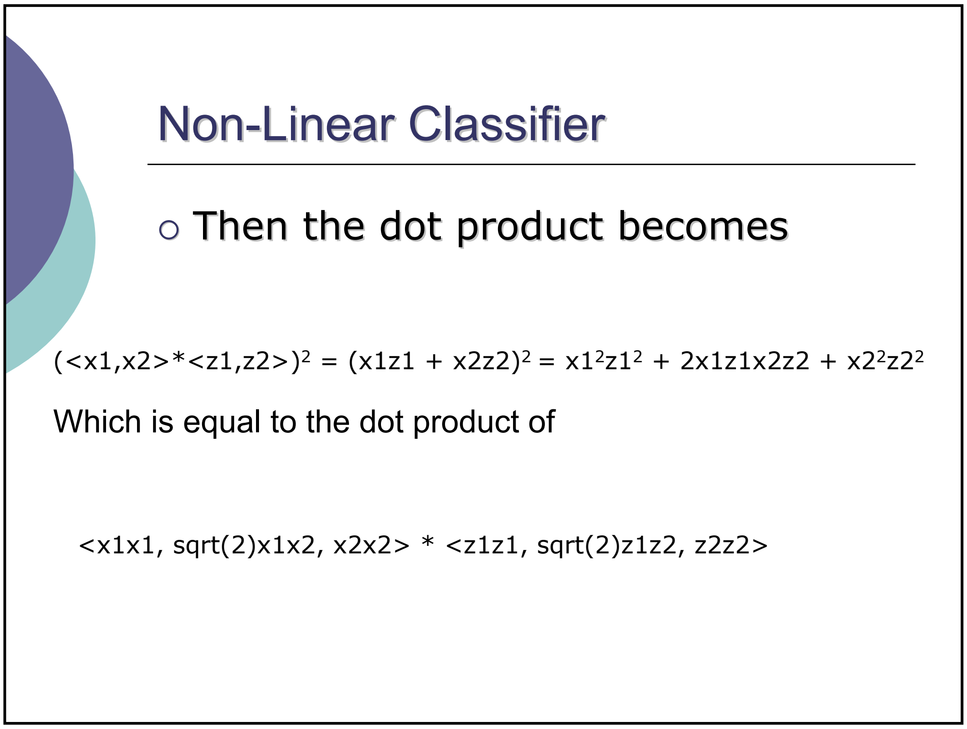
- The trick lies in the implicit projection into higher dimensional space.

<x1,x2>   <z1,z2>

<x1,x2>*<z1,z2> = x1z1 + x2z2

Suppose we have a kernel function defined as

$K(\vec{x},\vec{y}) = <\vec{x}*\vec{z}>^{2}$



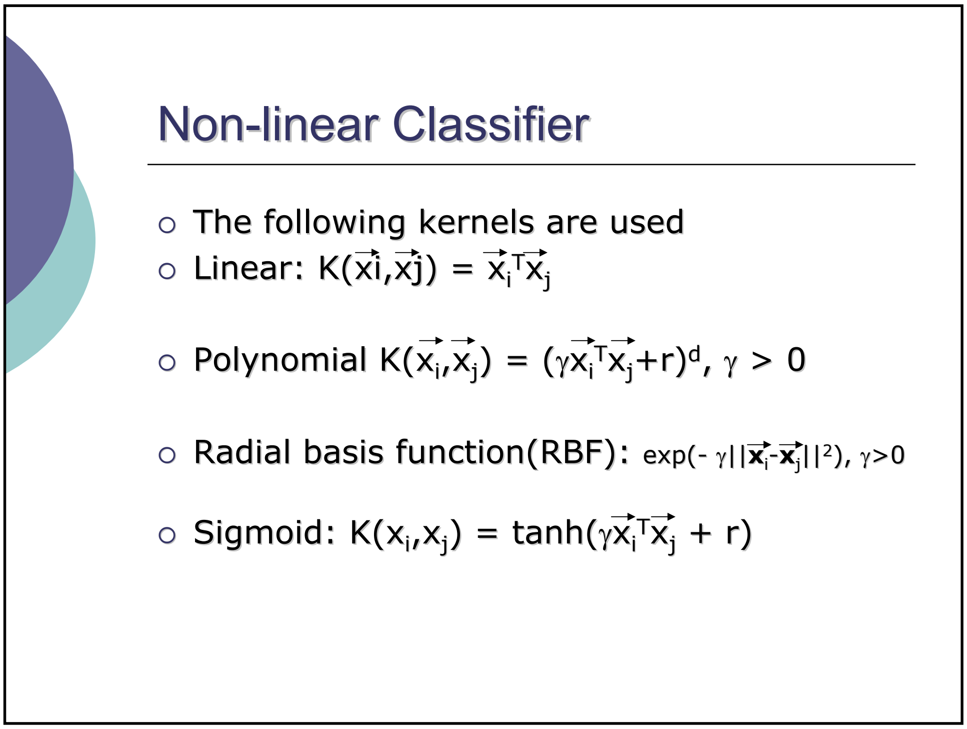
Non-Linear Classifier

- Then the dot product becomes

$(<x1,x2>*<z1,z2>)^2 = (x1z1 + x2z2)^2 = x1^2z1^2 + 2x1z1x2z2 + x2^2z2^2$

Which is equal to the dot product of

$<x1x1, \text{sqrt}(2)x1x2, x2x2> * <z1z1, \text{sqrt}(2)z1z2, z2z2>$



Non-linear Classifier

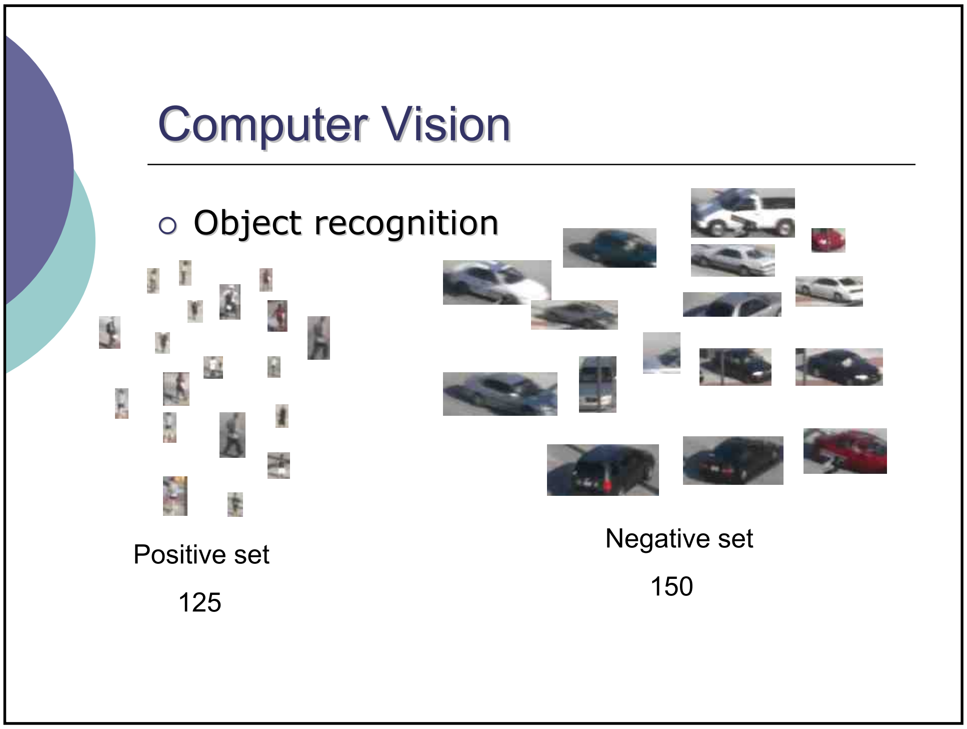
- The following kernels are used
- Linear: $K(\vec{xi},\vec{xj}) = \vec{x}_i^T\vec{x}_j$
- Polynomial $K(\vec{x}_i,\vec{x}_j) = (\gamma\vec{x}_i^T\vec{x}_j+r)^d$, $\gamma > 0$
- Radial basis function(RBF): $\exp(- \gamma||\vec{\mathbf{x}}_i-\vec{\mathbf{x}}_j||^2)$, $\gamma>0$
- Sigmoid: $K(x_i,x_j) = \tanh(\gamma\vec{x}_i^T\vec{x}_j + r)$

Non-linear Classifier

- When kernel function is used the decision function becomes

$$f(x) = \sum_{j=1}^{\infty} \lambda_j \psi_j \phi_j(\vec{\mathbf{x}}) + b = \sum_{j=1}^{l} \alpha_j y_j K(\vec{\mathbf{x}},\vec{\mathbf{x}}_j) + b$$

$$\vec{\psi} = \sum_{j=1}^{l} \alpha_j y_j \vec{\phi}_i(\vec{\mathbf{x}}_j)$$



Computer Vision

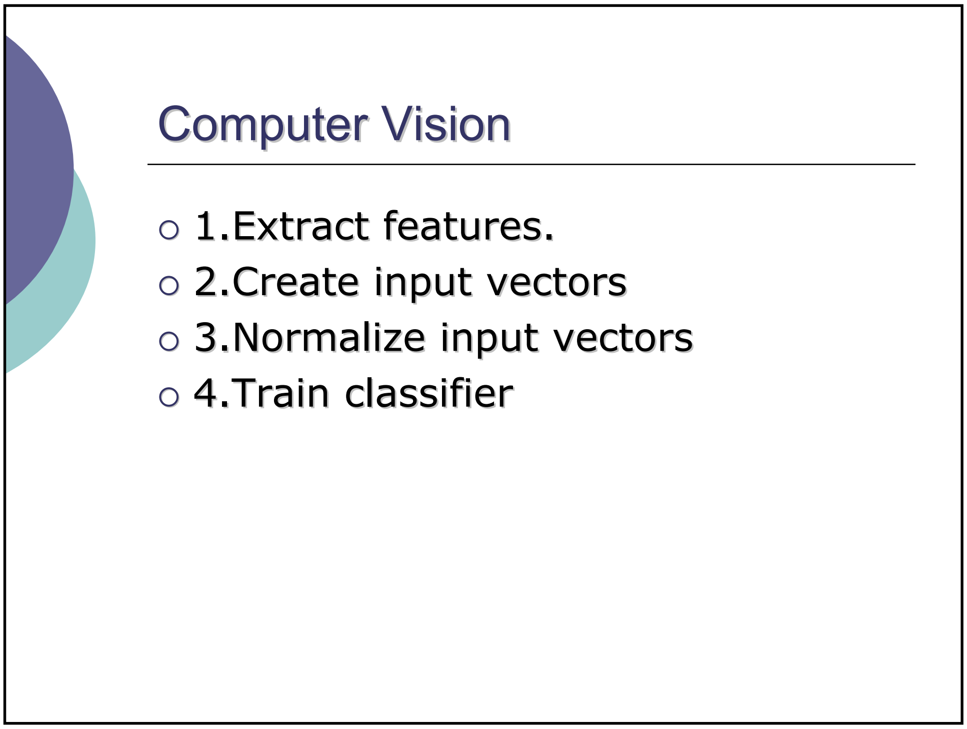
- Object recognition

Positive set

125

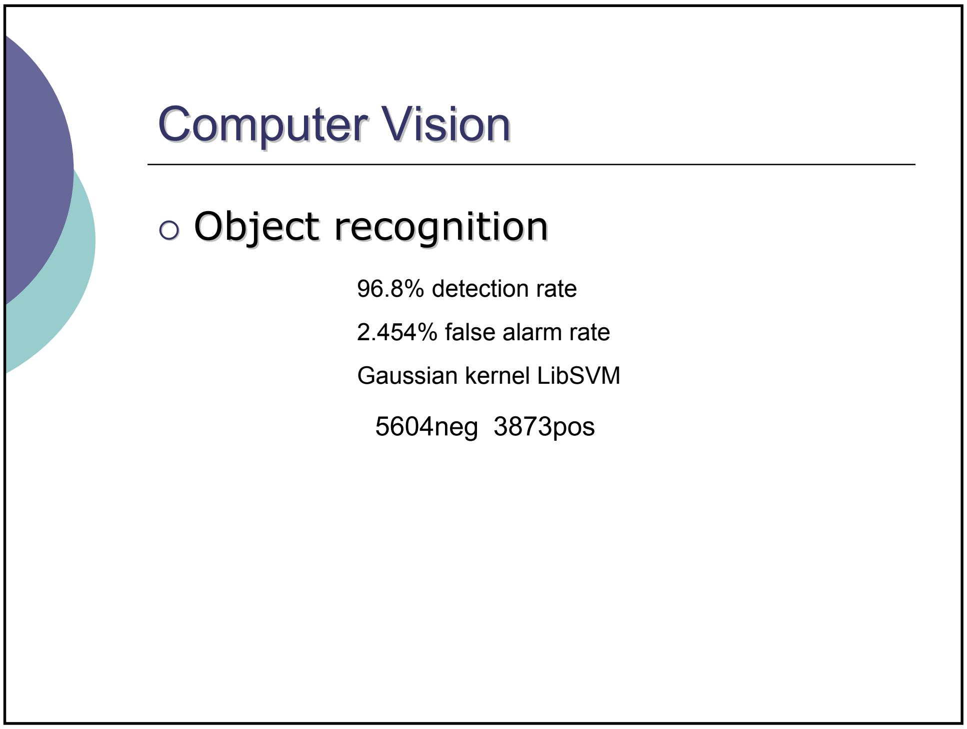
Negative set

150



Computer Vision

- 1.Extract features.
- 2.Create input vectors
- 3.Normalize input vectors
- 4.Train classifier



Computer Vision

- Object recognition

96.8% detection rate

2.454% false alarm rate

Gaussian kernel LibSVM

5604neg 3873pos

LibSVM

SVM implementation

- http://www.csie.ntu.edu.tw/~cjlin/libsvm/