# Nonlinear Classifiers

- At the beginning of the course we mentioned some broad approaches to learning nonlinear classifiers:
  - Considering good classes of nonlinear functions (Neural Networks)
  - Transforming the feature vector to a high dimensional space and learning a linear model
  - Splitting feature space into regions and learning a separate linear model in each region
- Next we look at the idea of learning a linear model in a transformed space.
- This is the approach of Support Vector Machines (SVM).

# The SVM approach

- We have briefly discussed Support Vector Machine (SVM) idea at the beginning of this course.
- The idea is to map the feature vectors nonlinearly into another space and learn a linear classifier there.
- The linear classifier in this new space would be an appropriate nonlinear classifier in the original space.

- Recall the simple example we saw earlier.
- Let $X = [x_1 \quad x_2]$ and let $\phi : \Re^2 \rightarrow \Re^5$ given by

  $Z = \phi(X) = [1 \quad x_1 \quad x_2 \quad x_1^2 \quad x_2^2 \quad x_1 x_2]$
- Now,

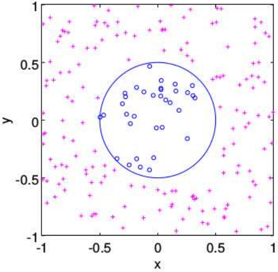
$$g(X) = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_1^2 + a_4 x_2^2 + a_5 x_1 x_2$$

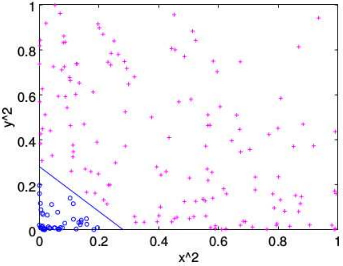
  is a quadratic discriminant function in $\Re^2$; but

$$g(Z) = a_0 + a_1 z_1 + a_2 z_2 + a_3 z_3 + a_4 z_4 + a_5 z_5$$

  is a linear dscriminant function in the '$\phi(X)$' space.

# Transforming Patterns to become Linearly Separable



- There are two major issues in naively using this idea.
- One is computational and the other is statistical
- If we want, e.g., $p^{th}$ degree polynomial discriminant function in the original feature space ($\Re^m$), then the transformed feature vector, Z, has dimension $O(m^p)$.
- Results in huge computational cost both for learning and and final operation of the classifier.
- We need to learn $O(m^p)$ parameters rather than $O(m)$ parameters. Hence may need much larger number of examples for achieving proper generalization.
- SVM offers an elegant solution to both.

# Support Vector Machines

- Learning of **optimal** hyperplane.
  - Separating hyperplane that maximizes separation between Classes.
- *Effectively* maps original feature vectors into a high dimensional space by use of **Kernel function**.
- By using Kernel function we never need to explicitly calculate the mapping.
- We need solve only a quadratic optimization problem.

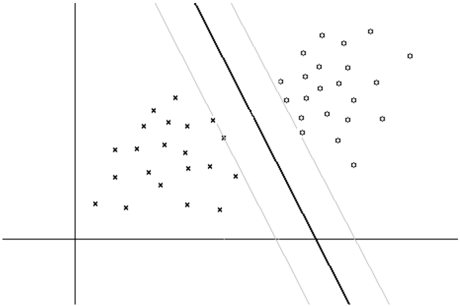
# A Separating Hyperplane

- Training set:
  $\{(X_i, y_i), \;\; i = 1, \ldots, n\}$, $X_i \in \Re^m$, $y_i \in \{+1, \;\; -1\}$.
- To start with, assume training set is linearly separable.
- That is, exist $W \in \Re^m$ and $b \in \Re$ such that

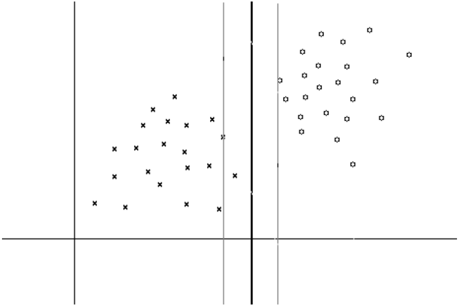
$$\begin{array}{rcl} W^T X_i + b & > & 0, \quad \forall i \;\; s.t. \;\; y_i = +1 \\ W^T X_i + b & < & 0, \quad \forall i \;\; s.t. \;\; y_i = -1 \end{array}$$

  (Note both inequalities are strict)
- $W^T X + b = 0$ – A separating hyperplane.
- Infinitely many separating hyperplanes exist.

# A good separating hyperplane (Ignore the two faint lines for now)



# Another separating hyperplane
# (Ignore the two faint lines for now)



- We assume training set is linearly separable and hence

$$\begin{aligned} W^T X_i + b &> 0, \quad \forall i \;\; s.t. \;\; y_i = +1 \\ W^T X_i + b &< 0, \quad \forall i \;\; s.t. \;\; y_i = -1 \end{aligned}$$

- Since the training set is finite, $\exists \epsilon_1, \epsilon_2 > 0 \; s.t.$

$$\begin{aligned} W^T X_i + b &\geq \epsilon_1, \quad \forall i \;\; s.t. \;\; y_i = +1 \\ W^T X_i + b &\leq -\epsilon_2, \quad \forall i \;\; s.t. \;\; y_i = -1 \end{aligned}$$

- By dividing by $\min\{\epsilon_1, \epsilon_2\}$,

$$\begin{aligned} \bar{W}^T X_i + \bar{b} &\geq +1 \quad \forall i \;\; s.t. \;\; y_i = +1 \\ \bar{W}^T X_i + \bar{b} &\leq -1 \quad \forall i \;\; s.t. \;\; y_i = -1 \end{aligned}$$

- Hence, when training set is linearly separable, we can scale $W,\ b$ such that

$$\begin{aligned} W^T X_i + b &\geq +1 \ \text{ if } \ y_i = +1 \\ W^T X_i + b &\leq -1 \ \text{ if } \ y_i = -1 \end{aligned}$$

or, equivalently

$$y_i(W^T X_i + b) \geq 1, \ \ \forall i.$$

(Recall that $y_i \in \{+1, \quad -1\}$)

- When the training set is separable, any separating hyperplane, $W,\ b$, can be scaled to satisfy

$$y_i(W^T X_i + b) \geq 1, \ \ \forall i.$$

- Then there are no training patterns between the two parallel hyperplanes

$$W^T X + b = +1$$

and

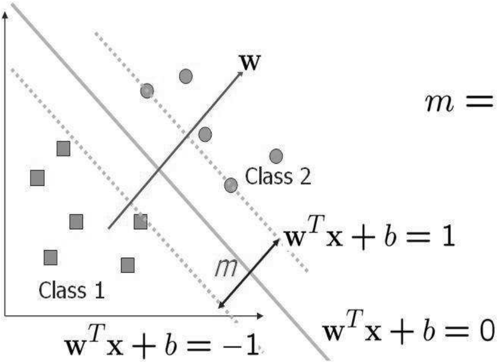
$$W^T X + b = -1$$

- The pattern nearest to the separating hyperplane is on one of these two.

# Optimal hyperplane

- Distance between these two hyperplanes is: $\frac{2}{||W||}$. Called **margin** of the separating hyperplane.
- The distance between the hyperplane and the closest pattern is $\frac{1}{||W||}$.
- Intuitively, more the margin, better is the chance of correct classification of new patterns.
- **Optimal Hyperplane** – separating hyperplane with maximum margin.

# Margin of a hyperplane




$$m = \frac{2}{||\mathbf{w}||}$$

# The optimization problem

- Among all separating hyperplanes, the one with largest margin is the optimal hyperplane.
- So, the optimal hyperplane is a solution to the following optimization problem.
- Find $W \in \Re^m$, $b \in \Re$ to

$$
\begin{array}{ll}
\text{minimize} & \frac{1}{2} W^T W \\
\text{subject to} & y_i (W^T X_i + b) \geq 1, \;\; i = 1, \ldots, n
\end{array}
$$

- This is a constrained optimization problem with quadratic cost function and linear inequality constraints.

# Constrained Optimization

- Consider the following optimization problem

$$\begin{array}{rl} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{a}_j^T\mathbf{x} + b_j \leq 0, \quad j = 1, \ldots, r \end{array}$$

  where $f : \Re^m \to \Re$ is a continuously differentiable function, and
  $\mathbf{a}_j \in \Re^m$, $b_j \in \Re$, $j = 1, \cdots, r$.
- A point, $\mathbf{x} \in \Re^m$, is called a **feasible** point (for this problem) if

$$\mathbf{a}_j^T\mathbf{x} + b_j \leq 0,\ j = 1, \cdots, r.$$

  A feasible point satisfies all the constraints.

# Constrained Optimization

- Any $\mathbf{x}^* \in \Re^m$ is called a **local minimum** of the problem if $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x}$ that is feasible and is in a small neighbourhood of $\mathbf{x}^*$.
- Unlike in unconstrained optimization, here we need to minimize only over the feasible set.
- For example,

$$\min_x \ f(x) = x \ \text{ subject to } \ x \geq 0$$

  has a solution eventhough the unconstrained version (that is, without $x \geq 0$) has no solution.
- If $f(\mathbf{x}^*) \leq f(\mathbf{x}), \forall \mathbf{x} \in \Re^m$ and $\mathbf{x}$ feasible, then $\mathbf{x}^*$ is a **global minimum**.

- Here we would consider only the case where $f$ is a convex function.
- $f : \Re^m \rightarrow \Re$ is said to be a convex function if for all $\mathbf{x}_1, \mathbf{x}_2 \in \Re^m$ and for all $\alpha \in (0,\ 1)$,

$$f(\alpha \mathbf{x}_1 + (1-\alpha)\mathbf{x}_2) \leq \alpha f(\mathbf{x}_1) + (1-\alpha) f(\mathbf{x}_2)$$

- For example, $f(\mathbf{x}) = \mathbf{x}^T \mathbf{x}$ is a convex function.
- When $f$ is convex, in our optimization problem, every local minimum is also a global minimum.

- Consider the optimization problem

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{a}_j^T \mathbf{x} + b_j \leq 0, \quad j = 1, \ldots, r \end{array}$$

- Define

$$L(\mathbf{x}, \boldsymbol{\mu}) = f(\mathbf{x}) + \sum_{j=1}^{r} \mu_j (\mathbf{a}_j^T \mathbf{x} + b_j)$$

- The $L$ is called the Lagrangian of the problem and the $\mu_j$ are called the Lagrange multipliers.
- Essentially, the constrained optimization problem can be solved through unconstrained optimization of $L$.

# Kuhn-Tucker Conditions

- Consider the optimization problem with $f$ convex.
- Any $\mathbf{x}^*$ is a global minimum if and only if
  1. $\mathbf{x}^*$ is feasible and
  2. there exist $\mu_j^*,\ j = 1, \cdots, r$, such that
     - 2.1 $\nabla_x L(\mathbf{x}^*, \boldsymbol{\mu}^*) = 0$
     - 2.2 $\mu_j^* \geq 0,\ \forall j$
     - 2.3 $\mu_j^*(\mathbf{a}_j^T \mathbf{x}^* + b_j) = 0,\ \forall j$
- These are the so called Kuhn-Tucker conditions for our optimization problem with convex cost function and linear constraints.

- We can use the above conditions to obtain a $\mathbf{x}^*$ which is a minimum of the optimization problem.
- We can also solve the constrained optimization problem using the so called dual of this problem.
- This is the approach taken in SVM algorithm.
- Duality is an important concept in optimization.
- Here we discuss only one way of formulating the dual which is useful when the objective function is convex and constraints are linear.

- Our optimization problem is

$$\begin{array}{ll} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{a}_j^T \mathbf{x} + b_j \leq 0, \quad j = 1, \ldots, r \end{array}$$

  where $f : \Re^m \to \Re$ is a continuously differentiable convex function, and

  $\mathbf{a}_j \in \Re^m$, $b_j \in \Re$, $j = 1, \cdots, r$.
- This is known as the **primal** problem.
- Here the optimization variables are $\mathbf{x} \in \Re^m$.

- Recall that the Lagrangian is

$$L(\mathbf{x}, \boldsymbol{\mu}) = f(\mathbf{x}) + \sum_{j=1}^{r} \mu_j(\mathbf{a}_j^T\mathbf{x} + b_j)$$

  Here, $\mathbf{x} \in \Re^m$ and $\boldsymbol{\mu} \in \Re^r$.
- Define the *dual function*, $q : \Re^r \to [-\infty,\ \infty)$ by

$$q(\boldsymbol{\mu}) = \inf_{\mathbf{x}}\ L(\mathbf{x}, \boldsymbol{\mu})$$

- If for a particular $\boldsymbol{\mu}$, if the infimum is not attained then $q(\boldsymbol{\mu})$ would take value $-\infty$.

# The Dual problem

- The **dual** problem is:

$$\begin{array}{ll} \text{maximize} & q(\boldsymbol{\mu}) \\ \text{subject to} & \mu_j \geq 0, \;\; j = 1, \ldots, r \end{array}$$

- This is also a constrained optimization problem.
- Here the optimization is over $\Re^r$ and $\boldsymbol{\mu} \in \Re^r$ are the optimization variables.
- There is a nice connection between the primal and dual problems.

# The Primal and the Dual

- The Primal problem:

$$\begin{array}{rl} \text{minimize} & f(\mathbf{x}) \\ \text{subject to} & \mathbf{a}_j^T\mathbf{x} + b_j \leq 0, \quad j = 1, \ldots, r \end{array}$$

- The *dual function*, $q : \Re^r \to [-\infty,\ \infty)$ is

$$q(\boldsymbol{\mu}) = \inf_{\mathbf{x}}\ L(\mathbf{x}, \boldsymbol{\mu}) = \inf_{\mathbf{x}}\ \left( f(\mathbf{x}) + \sum_{j=1}^{r} \mu_j(\mathbf{a}_j^T\mathbf{x} + b_j) \right)$$

- The Dual Problem:

$$\begin{array}{rl} \text{maximize} & q(\boldsymbol{\mu}) \\ \text{subject to} & \mu_j \geq 0, \quad j = 1, \ldots, r \end{array}$$

# Primal-Dual Relationship

- Now we have the following.
  1. If the primal has a solution so does the dual and the optimal values are equal.
  2. $\mathbf{x}^*$ is optimal for primal and $\boldsymbol{\mu}^*$ is optimal for dual if and only if
     **(i)**. $\mathbf{x}^*$ is feasible for primal and $\boldsymbol{\mu}^*$ is feasible for dual, and,
     **(ii)**. $f(\mathbf{x}^*) = L(\mathbf{x}^*,\ \boldsymbol{\mu}^*) = \min_{\mathbf{x}}\ L(\mathbf{x},\ \boldsymbol{\mu}^*)$.
- We would be using the dual formulation for the optimization problem in SVM

# The optimization problem for SVM

- The optimal hyperplane is a solution of the following constrained optimization problem.
- Find $W \in \Re^m$, $b \in \Re$ to

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2} W^T W \\ \text{subject to} \quad & 1 - y_i(W^T X_i + b) \leq 0, \;\; i = 1, \ldots, n \end{aligned}$$

- Quadratic cost function and linear (inequality) constraints.
- Kuhn-Tucker conditions are necessary and sufficient. Every local minimum is global minimum.

- The Lagrangian is given by

$$L(W,\ b,\ \boldsymbol{\mu}) = \frac{1}{2}W^T W + \sum_{i=1}^{n} \mu_i \left[1 - y_i(W^T X_i + b)\right]$$

- The Kuhn-Tucker conditions give

$$\nabla_W L = 0 \Rightarrow W^* = \sum_{i=1}^{n} \mu_i^* y_i X_i$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{n} \mu_i^* y_i = 0$$

$$1 - y_i(X_i^T W^* + b^*) \leq 0, \quad \forall i$$

$$\mu_i^* \geq 0, \quad \& \quad \mu_i^*[1 - y_i(X_i^T W^* + b^*)] = 0, \quad \forall i$$

- Let $S = \{i \mid \mu_i^* > 0\}$.
- From Kuhn-Tucker conditions, we have

$$\mu_i^*[1 - y_i(X_i^T W^* + b^*)] = 0$$

- Hence

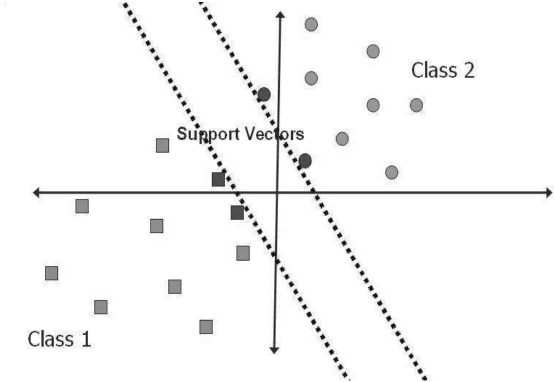
$$i \in S \quad \Rightarrow \quad \mu_i^* > 0 \quad \Rightarrow \quad y_i(X_i^T W^* + b^*) = 1$$

  Implies $X_i$ is closest to separating hyperplane.
- $\{X_i \mid i \in S\}$ are called Support vectors. We have

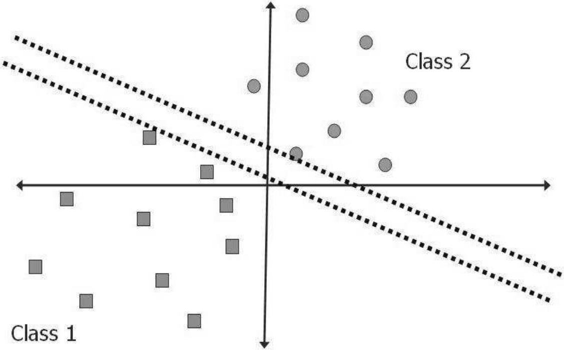
$$W^* = \sum_i \mu_i^* y_i X_i = \sum_{\mu_i^* > 0} \mu_i^* y_i X_i = \sum_{i \in S} \mu_i^* y_i X_i$$

- Optimal W is a linear combination of Support vectors.
- Support vectors constitute a very useful output of the method.

# Optimal hyperplane



# Non-optimal hyperplane



# The SVM solution

- The optimal hyperplane – $W^*$, $b^*$ is given by:

$$W^* = \sum_i \mu_i^* y_i X_i = \sum_{i \in S} \mu_i^* y_i X_i$$

$$b^* = y_j - X_j^T W^*, \quad j \text{ s.t. } \mu_j^* > 0$$

(Note that $\mu_j^* > 0 \;\; \Rightarrow \;\; y_j(X_j^T W^* + b^*) = 1$)

- Thus, $W^*, b^*$ are determined by $\mu_i^*, \;\; i = 1, \ldots, n$.
- We use the dual of the optimization problem to get $\mu_i^*$.

# Dual optimization problem for SVM

- The dual function is

$$q(\boldsymbol{\mu}) = \inf_{W,b} \left\{ \frac{1}{2} W^T W + \sum_{i=1}^{n} \mu_i [1 - y_i (W^T X_i + b)] \right\}$$

- Since we have a term $b \sum \mu_i y_i$ in the above, if $\sum \mu_i y_i \neq 0$ then $q(\boldsymbol{\mu}) = -\infty$.
- Hence we need to maximize $q$ only over those $\boldsymbol{\mu}$ s.t. $\sum \mu_i y_i = 0$.
- Infimum w.r.t. $W$ is attained at $W = \sum \mu_i y_i X_i$.
- We obtain the dual by substituting $W = \sum \mu_i y_i X_i$ and imposing $\sum \mu_i y_i = 0$.

- By substituting $W = \sum \mu_i y_i X_i$ and $\sum \mu_i y_i = 0$ we get

$$
\begin{aligned}
q(\boldsymbol{\mu}) &= \frac{1}{2} W^T W + \sum_{i=1}^{n} \mu_i - \sum_{i=1}^{n} \mu_i y_i (W^T X_i + b) \\
&= \frac{1}{2} \left( \sum_i \mu_i y_i X_i \right)^T \sum_j \mu_j y_j X_j + \sum_i \mu_i \\
&\qquad - \sum_i \mu_i y_i X_i^T \left( \sum_j \mu_j y_j X_j \right) \\
&= \sum_i \mu_i - \frac{1}{2} \sum_i \sum_j \mu_i y_i \mu_j y_j X_i^T X_j
\end{aligned}
$$

- Thus, the dual problem is:

$$\max_{\boldsymbol{\mu}} \qquad q(\boldsymbol{\mu}) = \sum_{i=1}^{n} \mu_i - \frac{1}{2} \sum_{i,j=1}^{n} \mu_i \mu_j y_i y_j X_i^T X_j$$

$$\text{subject to} \qquad \mu_i \geq 0, \;\; i = 1, \ldots, n, \;\; \sum_{i=1}^{n} y_i \mu_i = 0$$

- Quadratic cost function and linear constraints
- Training data vectors appear only as innerproduct
- Optimization is over $\Re^n$ irrespective of the dimension of $X_i$.

# Optimal hyperplane

- The optimal hyperplane is a solution of

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}W^T W \\ \text{subject to} \quad & y_i(W^T X_i + b) \geq 1, \;\; i = 1, \ldots, n \end{aligned}$$

- We solve the dual given by

$$\begin{aligned} \max_{\boldsymbol{\mu}} \quad & q(\boldsymbol{\mu}) = \sum_{i=1}^{n} \mu_i - \frac{1}{2} \sum_{i,j=1}^{n} \mu_i \mu_j y_i y_j X_i^T X_j \\ \text{subject to} \quad & \mu_i \geq 0, \;\; i = 1, \ldots, n, \;\; \sum_{i=1}^{n} y_i \mu_i = 0 \end{aligned}$$

- Then the final solution is:

$$W^* = \sum \mu_i^* y_i X_i, \;\; b^* = y_j - X_j^T W^*, \;\; j \text{ such that } \mu_j > 0$$

- So far, we assumed that the training data is linearly separable.
- What happens if the data is non-separable?
- Optimization problem has no feasible point (and hence no solution) if data are not linearly separable.
- We will modify the problem by introducing slack variables so that we can handle the general case.

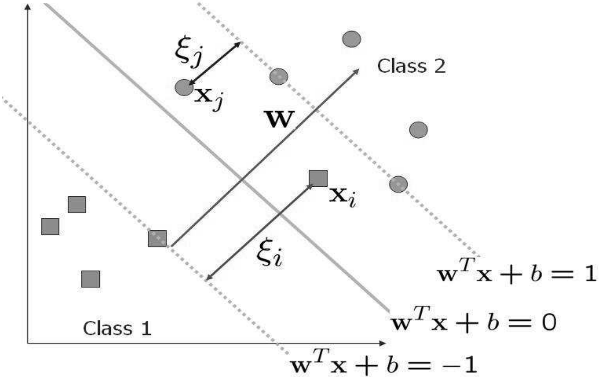
# Using slack variables

- When data are not linearly separable, we can try:

$$
\begin{array}{ll}
\text{minimize} & \frac{1}{2}W^TW \,+\, C\sum_{i=1}^{n} \xi_i \\
\text{subject to} & y_i(W^TX_i + b) \geq 1 - \xi_i, \;\; i = 1, \ldots, n \\
 & \xi_i \geq 0, \;\; i = 1, \ldots, n
\end{array}
$$

- Opt. variables: $W, b, \xi_i$.

- Feasible solution always exists.
- $\xi_i$ measure extent of violation of optimal separation.
- When $\xi > 0$, there is a 'margin error'. When $\xi_i > 1$, $X_i$ is wrongly classified.
- $C$ – user-specified constant. (Like regularization parameter).



$\xi_j$
Class 2
$\mathbf{x}_j$
$\mathbf{w}$
$\mathbf{x}_i$
$\xi_i$
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}^T\mathbf{x} + b = 0$
Class 1
$\mathbf{w}^T\mathbf{x} + b = -1$

# $L_1$ and $L_2$ SVM

- The formulation we saw is called $L_1$ SVM

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}W^TW \; + \; C\sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & y_i(W^TX_i + b) \geq 1 - \xi_i, \;\; i = 1, \ldots, n \\ & \xi_i \geq 0, \;\; i = 1, \ldots, n \end{aligned}$$

- A variant is $L_2$ SVM

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}W^TW \; + \; C\sum_{i=1}^{n} \xi_i^2 \\ \text{subject to} \quad & y_i(W^TX_i + b) \geq 1 - \xi_i, \;\; i = 1, \ldots, n \end{aligned}$$

- This distinction is recent and $L_2$ SVM is used along with some deep neural networks.

- Here we will consider only the $L_1$ SVM. This is the original and standard formulation.
- The optimization problem now is

$$\begin{aligned} \min_{W,b,\boldsymbol{\xi}} \quad & \frac{1}{2}W^T W \; + \; C \sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & 1 - \xi_i - y_i(W^T X_i + b) \leq 0, \;\; i = 1, \ldots, n \\ & -\xi_i \leq 0, \;\; i = 1, \ldots, n \end{aligned}$$

- The Lagrangian now is

$$
\begin{aligned}
L(W, b, \boldsymbol{\xi}, \boldsymbol{\mu}, \boldsymbol{\lambda}) \;\; &= \;\; \frac{1}{2} W^T W \; + \; C \sum_{i=1}^{n} \xi_i \\
&\quad + \sum_{i=1}^{n} \mu_i (1 - \xi_i - y_i(W^T X_i + b)) - \sum_{i=1}^{n} \lambda_i \xi_i
\end{aligned}
$$

- $\mu_i$ are the lagrange multipliers for the separability constraints as earlier.
- $\lambda_i$ are the lagrange multipliers for the constraints $-\xi_i \leq 0$.

$$L(W, b, \boldsymbol{\xi}, \boldsymbol{\mu}, \boldsymbol{\lambda}) \;=\; \frac{1}{2} W^T W \;+\; C \sum_{i=1}^{n} \xi_i$$
$$+ \sum_{i=1}^{n} \mu_i (1 - \xi_i - y_i(W^T X_i + b)) - \sum_{i=1}^{n} \lambda_i \xi_i$$

The Kuhn-Tucker conditions give us

- $\nabla_W L = 0 \Rightarrow W^* = \sum_{i=1}^{\ell} \mu_i^* y_i X_i$
- $\frac{\partial L}{\partial b} = 0 \Rightarrow \sum \mu_i^* y_i = 0$
- $\frac{\partial L}{\partial \xi_i} = 0 \Rightarrow \mu_i^* + \lambda_i^* = C, \ \forall i$
- $1 - \xi_i - y_i(W^T X_i + b) \leq 0; \ \ \xi_i \geq 0; \ \forall i$
- $\mu_i \geq 0; \ \lambda_i \geq 0, \ \forall i$
- $\mu_i (1 - \xi_i - y_i(W^T X_i + b)) = 0; \ \ \lambda_i \xi_i = 0, \ \forall i$

- The $W^*$ is given by the same expression.
- We also have $0 \leq \mu_i + \lambda_i = C, \ \forall i$.
- If $0 < \mu_i < C$, then, $\lambda_i > 0$ which implies $\xi_i = 0$.
- Now the complementary slackness condition, we have $1 - y_i(W^T X_i + b) = 0$.
- Thus we get $b^*$ as

$$b^* = y_j - X_j^T W^*, \ j \ \text{ such that } \ 0 < \mu_j < C$$

- We can derive the dual as before. The dual function is

$$q(\boldsymbol{\mu}, \boldsymbol{\lambda}) = \inf_{W, b, \boldsymbol{\xi}} L(W, b, \boldsymbol{\xi}, \boldsymbol{\mu}, \boldsymbol{\lambda})$$

where the lagrangian is given by

$$\begin{aligned} L(W, b, \boldsymbol{\xi}, \boldsymbol{\mu}, \boldsymbol{\lambda}) &= \frac{1}{2} W^T W + C \sum_{i=1}^{n} \xi_i \\ &\quad + \sum_{i=1}^{n} \mu_i (1 - \xi_i - y_i (W^T X_i + b)) - \sum_{i=1}^{n} \lambda_i \xi_i \end{aligned}$$

- In the lagrangian we have the term $\sum_i (C - \mu_i - \lambda_i)\xi_i$.
- Since we take infimum w.r.t. $\xi_i$, we need to impose $C = \mu_i + \lambda_i, \ \forall i$.
- When we impose this, all the terms containing $\lambda_i$ or $\xi_i$ drop out and hence now the $q$ function would be same as earlier.
- We only need to ensure (in the dual) that $\lambda_i \geq 0$ and $C = \mu_i + \lambda_i, \ \forall i$.
- This is easily done by ensuring $0 \leq \mu_i \leq C$.

# The dual

- The dual problem now is:

$$
\begin{array}{ll}
\max\limits_{\boldsymbol{\mu}} & q(\boldsymbol{\mu}) = \sum_{i=1}^{n} \mu_i - \frac{1}{2} \sum_{i,j=1}^{n} \mu_i \mu_j y_i y_j X_i^T X_j \\
\text{subject to} & 0 \le \mu_i \le C, \;\; i = 1, \ldots, n, \;\; \sum_{i=1}^{n} y_i \mu_i = 0
\end{array}
$$

- The only difference – upper bound also on $\mu_i$.

- The primal problem is

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}W^TW \; + \; C\sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & y_i(W^TX_i + b) \geq 1 - \xi_i, \;\; i = 1, \ldots, n \\ & \xi_i \geq 0, \;\; i = 1, \ldots, n \end{aligned}$$

- The dual problem is:

$$\begin{aligned} \max_{\boldsymbol{\mu}} \quad & q(\boldsymbol{\mu}) = \sum_{i=1}^{n} \mu_i - \frac{1}{2}\sum_{i,j=1}^{n} \mu_i\mu_j y_i y_j X_i^T X_j \\ \text{subject to} \quad & 0 \leq \mu_i \leq C, \;\; i = 1, \ldots, n, \;\; \sum_{i=1}^{n} y_i\mu_i = 0 \end{aligned}$$

- As $C \to \infty$, we get back the old problem.
- Solving the dual is a better strategy

- The dual problem is:

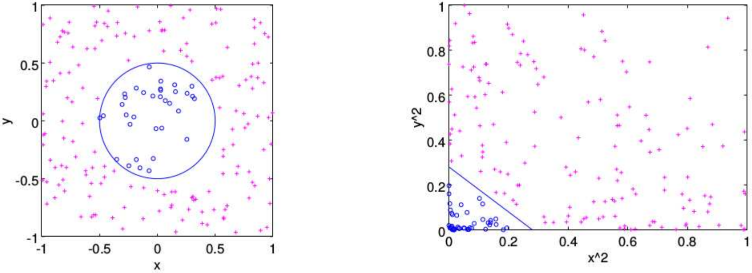
$$\max_{\boldsymbol{\mu}} \qquad q(\boldsymbol{\mu}) = \sum_{i=1}^{n} \mu_i - \frac{1}{2} \sum_{i,j=1}^{n} \mu_i \mu_j y_i y_j X_i^T X_j$$

$$\text{subject to} \qquad 0 \leq \mu_i \leq C, \;\; i = 1, \ldots, n, \;\; \sum_{i=1}^{n} y_i \mu_i = 0$$

- We solve dual and the final optimal hyperplane is
  $W^* = \sum \mu_i^* y_i X_i$,
  $b^* = y_j - X_j^T W^*$, $j$ such that $0 < \mu_j < C$.

- By using slack variables, $\xi_i$, we can find 'best' hyperplane classifier.
- In the dual, the only difference is an upperbound on $\mu_i$.
- How can we learn non-linear classifiers?
- Recall that the SVM idea is to transform $X_i$ into some other high-dimensional space and learn a linear classifier there.

# Transforming Patterns to become Linearly Separable



# Non-linear classifiers

- In general, we can use a mapping, $\phi : \Re^m \to \Re^{m'}$.
- In $\Re^{m'}$, the training set is $\{(Z_i,\ y_i),\ i = 1, \ldots, \ell\}$, $Z_i = \phi(X_i)$.
- We can find optimal hyperplane by solving the dual (replacing $X_i^T X_j$ with $Z_i^T Z_j$).
- The dual problem now would be the following.

$$\max_{\mu} \qquad q(\mu) = \sum_{i=1}^{n} \mu_i - \frac{1}{2} \sum_{i,j=1}^{n} \mu_i \mu_j y_i y_j \phi(X_i)^T \phi(X_j)$$

$$\text{subject to} \qquad 0 \leq \mu_i \leq C, \ \ i = 1, \ldots, n, \ \ \sum_{i=1}^{n} y_i \mu_i = 0$$

- This is an optimization problem over $\Re^n$ (with quadratic cost function & linear constraints) **irrespective of $\phi$ and $m'$.**
- But computationally expensive?

# Kernel function

- Suppose we have a function, $K : \Re^m \times \Re^m \rightarrow \Re$, such that

  $K(X_i,\ X_j) = \phi(X_i)^T \phi(X_j)$

  Called Kernel function.
- Suppose computation of $K(X_i,\ X_j)$ is about as expensive as that of $X_i^T X_j$.
- Replacing $Z_i^T Z_j$ by $K(X_i,\ X_j)$, we can solve dual without ever computing any $\phi(X_i)$. Efficient for obtaining optimal hyperplane.
- What about storing $W^*$? Computing $\phi(X)^T W^*$ for new patterns?

# Kernel function based classifier

- Let $\mu_i^*$ be soln of Dual. Then $W^* = \sum \mu_i^* y_i \phi(X_i)$.
- Then we have

$$b^* = y_j - \phi(X_j)^T W^* = (y_j - \sum_i \mu_i^* y_i \phi(X_i)^T \phi(X_j))$$

- Given a new pattern $X$ we only need to compute

$$\begin{aligned} f(X) &= Z^T W^* + b^* = \phi(X)^T W^* + b^* \\ &= \sum_i \mu_i^* y_i \phi(X_i)^T \phi(X) + b^* \\ &= \sum_i \mu_i^* y_i K(X_i,\ X) + (y_j - \sum_i \mu_i^* y_i K(X_i,\ X_j)) \end{aligned}$$

- This is an interesting way of learning nonlinear classifiers.
- We solve the dual whose dimension is that of $n$, number of examples.
- All we need to store are:
  - non-zero Lagrange multipliers: $\mu_i^* > 0$,
  - Support vectors: $X_i, \ i$ s.t. $\mu_i^* > 0$.
- Then we compute

  $$f(X) = \sum_i \mu_i^* y_i K(X_i,\ X) + (y_j - \sum_i \mu_i^* y_i K(X_i,\ X_j))$$

  and classify $X$ based on sign of $f(X)$.
- Never need to enter '$\phi(X)$' space!

# Support Vector Machine

- Obtain $\mu_i^*$ by solving the Dual with $Z_i^T Z_j$ replaced by $K(X_i,\ X_j)$. (Choose a suitable Kernel function. Use 'penalty const', $C$ as needed).
- Store non-zero $\mu_i^*$ and the corresponding support vectors.
- Classify any new pattern $X$ by sign of

$$f(X) = \sum \mu_i^* y_i K(X_i,\ X) + (y_j - \sum_i \mu_i^* y_i K(X_i,\ X_j))$$

- If we have a suitable Kernel function, we never need to compute $\phi(X)$.
- The range space of $\phi$ can even be infinite dimensional!

# Example kernel function

- We start with an example kernel function in $\Re^2$.
- Consider $K(X_i,\ X_j) = (1 + X_i^T X_j)^2$.
- Let $X_i = (x_{i1},\ x_{i2})^T \in \Re^2$ and similarly for $X_j$.
- Then
$$K(X_i,\ X_j) = (1 + x_{i1}x_{j1} + x_{i2}x_{j2})^2$$
- We now show that there exists a mapping $\phi$ such that $K(X_i, X_j) = \phi(X_i)^T \phi(X_j)$.

- Consider $\phi : \Re^2 \rightarrow \Re^6$ given by

$$Z = \phi(X) = [1 \quad x_1^2 \quad x_2^2 \quad \sqrt{2}x_1 \quad \sqrt{2}x_2 \quad \sqrt{2}x_1x_2]$$

(Here, $X = (x_1 \;\; x_2) \in \Re^2$).

- It is easy to see that a linear discriminant function in terms of $Z$ (i.e., in $\Re^6$) would be a quadratic discriminant function in terms of $X$ (i.e., in $\Re^2$).
- Now we show that

$$K(X_i, X_j) = (1 + X_i^T X_j)^2 = Z_i^T Z_j = \phi(X_i)^T \phi(X_j)$$

- Recall

$$Z_i = \phi(X_i) = [1 \quad x_{i1}^2 \quad x_{i2}^2 \quad \sqrt{2}x_{i1} \quad \sqrt{2}x_{i2} \quad \sqrt{2}x_{i1}x_{i2}]$$

We have

$$\begin{aligned} Z_i^T Z_j &= 1 + x_{i1}^2 x_{j1}^2 + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2} + 2x_{i1}x_{i2}x_{j1}x_{j2} \\ &= (1 + x_{i1}x_{j1} + x_{i2}x_{j2})^2 \\ &= K(X_i,\ X_j) \end{aligned}$$

- Easy to see it works for $X \in \Re^n$ in general.
- Thus $K(X_i,\ X_j) = (1 + X_i^T X_j)^2$ results in a quadratic discriminant function or a quadratic classifier.

- From this example, it is also easy to see that for a given Kernel function, the mapping $\phi$ (or the dimension of its range space) is not unique.
- Consider the same Kernel fn $K(X_i, X_j) = (1 + X_i^T X_j)^2$.
- Consider the mapping $\phi : \Re^2 \rightarrow \Re^7$ given by

$$Z = \phi(X) = [1 \quad \sqrt{2}x_1 \quad \sqrt{2}x_2 \quad x_1^2 \quad x_2^2 \quad x_1x_2 \quad x_1x_2]$$

- It is easy to see that this mapping also works.

- We saw that the Kernel $K(X, X') = (1 + X^T X')^2$ results in a quadratic discriminant function (in the original feature space)
- This is because the effective $\phi$ function is such that each $x_i x_j$ term is a component of $\phi(X)$.
- Thus, if $X \in \Re^m$, then any reasonable $\phi$ function corresponding to this kernel would have range space with dimension $O(m^2)$.
- Hence, $\phi(X_i)^T \phi(X_j)$ would need $O(m^2)$ multiplications.
- If we are using a linear SVM, we only need $X_i^T X_j$ which needs $m$ multiplications.
- When we use the Kernel for the quadratic case, we need only $m + 1$ multiplications.

# Kernel functions

- How do we obtain Kernel functions in general?
- What kind of symmetric functions capture the inner product in an appropriate space?
- We look at two important charectarizations for Kernel functions.

# Mercer Kernels

- **Mercer Theorem**:
  Given a symmetric function, $K : \Re^m \times \Re^m \to \Re$,

  there exists an inner product space $\mathcal{H}$ and a mapping $\phi : \Re^m \to \mathcal{H}$ so that
  $K(X_1,\ X_2) = \phi(X_1)^T \phi(X_2)$

  if for all square-integrable functions $g$,

  $$\int K(X_1,\ X_2) g(X_1) g(X_2) dX_1\, dX_2 \geq 0.$$

# Positive definite kernels

- Let $\bar{K}$ be a $n \times n$ matrix with $\bar{K}_{i,j} = K(X_i,\ X_j)$.
- A **positive definite kernel** is the function $K$ such that $\bar{K}$ is positive semi-definite for all $n$ and all data sets $\{X_1, \ldots, X_n\}$.
- That is, given any $n$, and any feature vectors, $X_1, \cdots, X_n$, we have, for all scalars $c_1, \cdots, c_n$,

$$\sum_{i,j=1}^{n} c_i c_j K(X_i, X_j) \geq 0$$

- If input space is compact, both these notions are same.

- Now we use Mercer's theorem to show that the function we gave earlier would be a Kernel function.
- Consider the function

$$K(U,\ V) = (U^T V)^p = \left( \sum_{i=1}^{m} u_i v_i \right)^p$$

  where $p > 0$ is an integer and $U = [u_1 \cdots u_m]^T$ and $V = [v_1 \cdots v_m]^T$ are in $\Re^m$.
- We want to show that this satisfies the Mercer theorem.

- By expanding the $(U^T V)^p$ we get an expression

$$\left(\sum_{i=1}^{m} u_i v_i\right)^p = \sum_{r1,\, \cdots\, rm} \frac{p!}{r1!\, r2!\, \cdots\, rm!} \prod_{i=1}^{m} (u_i v_i)^{ri}$$

where the summation is over all non-negative integers, $r1,\, \cdots,\, rm$ such that

$$r1 \,+\, r2 \,+\, \cdots \,+\, rm \,=\, p$$

- We need to show

$$\int_{\Re^m} \int_{\Re^m} \left( \sum_{i=1}^{m} u_i v_i \right)^p g(U)\, g(V) dU\, dV > 0.$$

- This becomes a sum of integrals by expanding $(\sum u_i v_i)^p$.
- A typical term here is

$$\frac{p!}{r1!\, r2! \cdots rm!} \int \int (u_1 v_1)^{r1} (u_2 v_2)^{r2} \cdots (u_m v_m)^{rm} g(U) g(V) dU dV$$

$$= \frac{p!}{r1!\, r2! \cdots rm!} \int \, (u_1)^{r1} (u_2)^{r2} \cdots (u_m)^{rm} g(U) dU \int (v_1)^{r1} (v_2)^{r2} \cdots (v_m)^{rm} g(V)\, dV$$

$$= \frac{p!}{r1!\, r2! \cdots rm!} \left( \int \; u_1^{r1}\, u_2^{r2} \; \cdots \; u_m^{rm} g(U)\, dU \right)^2 \geq 0$$

- Now consider the function

$$K(U,V) = \sum_{j=0}^{p} a_j \,(U^T V)^j, \;\; a_j \geq 0$$

- We can show this also satisfies Mercer theorem

$$\begin{gathered}\int \sum_{j=0}^{p} a_j \,(U^T V)^j \, g(U)\, g(V)\, dU\, dV \\ = \sum_{j=0}^{p} a_j \int (U^T V)^j \, g(U)\, g(V)\, dU\, dV \\ \geq 0\end{gathered}$$

- Hence functions of the form

$$K(X_1, X_2) = \sum_{j=0}^{p} a_j \, (X_1^T X_2)^j, \;\; a_j \geq 0$$

  are kernels (satisfying Mercer's theorem).
- A special case is

$$K(X_1, X_2) = (1 + X_1^T X_2)^p$$

  which is an example we considered earlier.
- This is called a polynomial kernel.

- Now consider the functions of the type

$$K(U, V) = \sum_{j=0}^{\infty} a_j \, (U^T V)^j, \;\; a_j \geq 0$$

- Our proof only involved interchanging integration and summation.
- For finite sum it is always possible.
- For infinite sum, a sufficient condition is that the above sum is uniformly convergent
- Then the above would also satisfy Mercer's theorem.

- Consider the function

$$K(X_1,\ X_2) = e^{-\frac{(X_1 - X_2)^T (X_1 - X_2)}{2\sigma^2}}$$

- We can show it satisfies the theorem by noting

$$e^{-(X_1 - X_2)^T (X_1 - X_2)} = e^{-X_1^T X_1}\ e^{-X_2^T X_2} e^{2X_1^T X_2},$$

and

$$e^{2X_1^T X_2} = \sum_{p=0}^{\infty} \frac{(2X_1^T X_2)^p}{p!}$$

# Some Popular Kernel functions

- Polynomial kernel:

$$K_p(X_1,\ X_2) = (1 + X_1^T X_2)^p$$

- Gaussian kernel

$$K_G(X_1,\ X_2) = e^{-\frac{||X_1 - X_2||^2}{\sigma^2}}$$