COMS 4771
Support Vector Machines

Nakul Verma
Say there is a **linear** decision boundary which can **perfectly separate** the training data.

*Which linear separator will the Perceptron algorithm return?*

The separator with a **large margin** $\gamma$ is better for generalization.

*How can we incorporate the margin in finding the linear boundary?*
Solution: Support Vector Machines (SVMs)

Motivation:

• It returns a linear classifier that is stable solution by giving a maximum margin solution

• Slight modification to the problem provides a way to deal with non-separable cases

• It is kernelizable, so gives an implicit way of yielding non-linear classification.
SVM Formulation

Say the training data $S$ is **linearly separable** by some margin (but the linear separator does not necessarily passes through the origin).

Then:

**decision boundary:** $g(\vec{x}) = \vec{w} \cdot \vec{x} - b = 0$

**Linear classifier:**

\[
f(\vec{x}) = \text{sign}(g(\vec{x})) = \text{sign}(\vec{w} \cdot \vec{x} - b)
\]

*Idea: we can try finding two parallel hyperplanes that correctly classify all the points, and **maximize** the distance between them!*
SVM Formulation (contd. 1)

Decision boundary for the two hyperplanes:
\[
\vec{w} \cdot \vec{x} - b = +1 \\
\vec{w} \cdot \vec{x} - b = -1
\]

Distance between the two hyperplanes:
\[
\frac{2}{\|\vec{w}\|} \quad \text{why?}
\]

Training data is correctly classified if:
\[
\begin{align*}
\vec{w} \cdot \vec{x}_i - b & \geq +1 \quad \text{if } y_i = +1 \\
\vec{w} \cdot \vec{x}_i - b & \leq -1 \quad \text{if } y_i = -1
\end{align*}
\]

Together:
\[
y_i (\vec{w} \cdot \vec{x}_i - b) \geq +1 \quad \text{for all } i
\]
SVM Formulation (contd. 2)

Distance between the hyperplanes: \[ \frac{2}{\|\vec{w}\|} \]

Training data is correctly classified if: \[ y_i (\vec{w} \cdot \vec{x}_i - b) \geq +1 \quad (for \ all \ i) \]

Therefore, want:

Maximize the distance: \[ \frac{2}{\|\vec{w}\|} \]

Such that: \[ y_i (\vec{w} \cdot \vec{x}_i - b) \geq +1 \quad (for \ all \ i) \]

Let’s put it in the standard form...
SVM Formulation (finally!)

Maximize: \[ \frac{2}{\|w\|} \]

Such that: \[ y_i (\vec{w} \cdot \vec{x}_i - b) \geq 1 \] (for all \( i \))

Minimize: \[ \frac{1}{2} \|w\|^2 \]

Such that: \[ y_i (\vec{w} \cdot \vec{x}_i - b) \geq 1 \] (for all \( i \))

SVM standard (primal) form:

What can we do if the problem is not-linearly separable?
Idea: introduce a \textbf{slack} for the misclassified points, and \textbf{minimize} the slack!

\textbf{SVM standard (primal) form (with slack)}:

Minimize: \[
\frac{1}{2} \| \vec{w} \|^2 + C \sum_{i=1}^{n} \xi_i
\]

Such that: \[
y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1 - \xi_i \\
(\text{for all } i)\\n\xi_i \geq 0
\]
**SVM: Question**

**SVM standard (primal) form (with slack):**

Minimize: \[ \frac{1}{2} \| \vec{w} \|^2 + C \sum_{i=1}^{n} \xi_i \]

Such that: 

\[ y_i (\vec{w} \cdot \vec{x}_i - b) \geq 1 - \xi_i \] 
(for all i)

\[ \xi_i \geq 0 \]

**Questions:**

1. *How do we find the optimal \( w, b \) and \( \xi \)?*
2. *Why is it called “Support Vector Machine”?*
How to Find the Solution?

Cannot simply take the derivative (wrt $w$, $b$ and $\xi$) and examine the stationary points...

Why?

Minimize: $x^2$
Such that: $x \geq 5$

$\text{Gradient not zero at the function minima (respecting the constraints)!}$

$\text{SVM standard (primal) form:}$

Minimize: $\frac{1}{2} \|\vec{w}\|^2 + C \sum_{i=1}^{n} \xi_i$

Such that: $y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1 - \xi_i$
(for all $i$) $\xi_i \geq 0$

$\text{(infeasible region)}$

Need a way to do optimization with constraints
Constrained optimization (standard form):

\[
\begin{align*}
\text{minimize} \quad & f(\mathbf{x}) \\
\text{subject to:} \quad & g_i(\mathbf{x}) \leq 0 \quad \text{for} \ 1 \leq i \leq n
\end{align*}
\]

What to do?

- **Projection methods**
  
  start with a feasible solution \( x_0 \),
  
  find \( x_1 \) that has slightly lower objective value,
  
  if \( x_1 \) violates the constraints, **project back** to the constraints.
  
  iterate.

- **Penalty methods**
  
  use a **penalty function** to incorporate the constraints into the objective

- ...
The Lagrange (Penalty) Method

Consider the augmented function:

\[ L(\vec{x}, \vec{\lambda}) := f(\vec{x}) + \sum_{i=1}^{n} \lambda_i g_i(\vec{x}) \]

(Lagrange function)

(Lagrange variables, or dual variables)

Observation:

For any feasible \( \vec{x} \) and all \( \lambda_i \geq 0 \), we have

\[ L(\vec{x}, \vec{\lambda}) \leq f(\vec{x}) \]

\[ \implies \max_{\lambda_i \geq 0} L(\vec{x}, \vec{\lambda}) \leq f(\vec{x}) \]

• if \( \vec{x} \) is infeasible, then \( \max_{\lambda_i \geq 0} L(\vec{x}, \vec{\lambda}) = \infty \)
• if \( \vec{x} \) is feasible, then \( \max_{\lambda_i \geq 0} L(\vec{x}, \vec{\lambda}) = f(\vec{x}) \)

So, the optimal value/solution to the original constrained optimization:

\[ p^* := \min_{\vec{x}} \max_{\lambda_i \geq 0} L(\vec{x}, \vec{\lambda}) \]

The problem becomes unconstrained in \( \vec{x} \)!
The Dual Problem

Optimal value: \[ p^* = \min \max_{\vec{x}, \lambda_i \geq 0} L(\vec{x}, \vec{\lambda}) \]
(also called the primal)

Let \( x^* \) be the minimum feasible (over \( f \)),
For all \( \lambda_i \geq 0 \)
\[
\min_{\vec{x}} L(\vec{x}, \vec{\lambda}) \leq L(x^*, \vec{\lambda}) \leq f(x^*) = p^*
\]

Hence:
\[
d^* := \max_{\lambda_i \geq 0} \min_{\vec{x}} L(\vec{x}, \vec{\lambda}) \leq p^*
\]
(also called the dual)

Optimization problem:

Minimize: \( f(\vec{x}) \)
Such that: \( g_i(\vec{x}) \leq 0 \)
(for all \( i \))

Lagrange function:
\[
L(\vec{x}, \vec{\lambda}) := f(\vec{x}) + \sum_{i=1}^{n} \lambda_i g_i(\vec{x})
\]
Theorem (weak Lagrangian duality):
\[ d^* \leq p^* \]
(also called the minimax inequality)

\[ p^* - d^* \quad \text{(called the duality gap)} \]

Under what conditions can we achieve equality?

Optimization problem:
Minimize: \( f(\vec{x}) \)
Such that: \( g_i(\vec{x}) \leq 0 \) (for all \( i \))

Lagrange function:
\[ L(\vec{x}, \vec{\lambda}) := f(\vec{x}) + \sum_{i=1}^{n} \lambda_i g_i(\vec{x}) \]

Primal:
\[ p^* = \min_{\vec{x}} \max_{\lambda_i \geq 0} L(\vec{x}, \vec{\lambda}) \]

Dual:
\[ d^* := \max_{\lambda_i \geq 0} \min_{\vec{x}} L(\vec{x}, \vec{\lambda}) \]
A function \( f : \mathbb{R}^d \rightarrow \mathbb{R} \) is called convex iff for any two points \( x, x' \) and \( \beta \in [0,1] \)

\[
f(\beta \vec{x} + (1 - \beta) \vec{x}') \leq \beta f(\vec{x}) + (1 - \beta) f(\vec{x}')
\]
Convexity

A set $S \subseteq \mathbb{R}^d$ is called convex iff for any two points $x, x' \in S$ and any $\beta \in [0,1]$

$$\beta \bar{x} + (1 - \beta) \bar{x}' \in S$$

Examples:
Convex Optimization

A constrained optimization

\[
\begin{align*}
\text{minimize} & \quad f(\overline{x}) \\
\text{subject to:} & \quad g_i(\overline{x}) \leq 0 \quad \text{for } 1 \leq i \leq n
\end{align*}
\]

is called convex a convex optimization problem

If:

- the objective function \( f(\overline{x}) \) is convex function, and
- the feasible set induced by the constraints \( g_i \) is a convex set

\( (\text{if all } f \text{ and } g \text{ are convex, then the constraint problem is a convex optimization}) \)

Why do we care?

We and find the optimal solution for convex problems efficiently!
Convex Optimization: Niceties

• Every local optima is a **global optima** in a convex optimization problem.
  
  Example convex problems:
  
  Linear programs, quadratic programs, 
  Conic programs, semi-definite program.

  Several **solvers exist** to find the optima:
  
  CVX, SeDuMi, C-SALSA, ...

• We can use a **simple** ‘descend-type’ algorithm for finding the minima!
Gradient Descent (for finding local minima)

Theorem (Gradient Descent):
Given a smooth function $f : \mathbb{R}^d \to \mathbb{R}$
Then, for any $\bar{x} \in \mathbb{R}^d$ and $\bar{x}' := \bar{x} - \eta \nabla_x f(\bar{x})$
For sufficiently small $\eta > 0$, we have: $f(\bar{x}') \leq f(\bar{x})$

Can derive a **simple algorithm** (the projected Gradient Descent):

Initialize $\bar{x}^0$
for t = 1,2,...do

\[ \bar{x}'^t := \bar{x}^{t-1} - \eta \nabla_x f(\bar{x}^{t-1}) \] \hspace{1cm} (step in the gradient direction)

\[ \bar{x}^t := \Pi_{g_i}(\bar{x}^t) \] \hspace{1cm} (project back onto the constraints)

terminate when no progress can be made, ie, $|f(\bar{x}^t) - f(\bar{x}^{t-1})| \leq \epsilon$
Theorem (weak Lagrangian duality):
\[ d^* \leq p^* \]

Theorem (strong Lagrangian duality):
For a convex optimization problem, if there exists a feasible point \( x \), s.t.
\[ g_i(\bar{x}) < 0 \quad (\text{for all } i) \] or
\[ g_i(\bar{x}) \leq 0 \quad \text{whenever } g_i \text{ is affine} \]

Then
\[ d^* = p^* \quad \text{(aka Slater’s condition; sufficient for strong duality)} \]

Optimization problem:

Minimize: \( f(\bar{x}) \)  
Such that: \( g_i(\bar{x}) \leq 0 \) (for all \( i \))

Lagrange function:
\[ L(\bar{x}, \bar{\lambda}) := f(\bar{x}) + \sum_{i=1}^{n} \lambda_i g_i(\bar{x}) \]

Primal:
\[ p^* = \min_{\bar{x}} \max_{\lambda_i \geq 0} L(\bar{x}, \bar{\lambda}) \]

Dual:
\[ d^* := \max_{\lambda_i \geq 0} \min_{\bar{x}} L(\bar{x}, \bar{\lambda}) \]
Ok, Back to SVMs

Observations:
• object function is convex
• the constraints are affine, inducing a polytope constraint set.

So, SVM is a convex optimization problem (in fact a quadratic program)

Moreover, strong duality holds.

Let’s examine the dual... the Lagrangian is:

\[ L(\vec{w}, b, \alpha) = \frac{1}{2} \| \vec{w} \|^2 + \sum_{i=1}^{n} \alpha_i (1 - y_i (\vec{w} \cdot \vec{x}_i - b)) \]
SVM Dual

Lagrangian:

\[
L(\vec{w}, b, \vec{\alpha}) = \frac{1}{2} \| \vec{w} \|^2 + \sum_{i=1}^{n} \alpha_i (1 - y_i (\vec{w} \cdot \vec{x_i} - b))
\]

Primal: \( p^* = \min_{\vec{w}, b} \max_{\alpha_i \geq 0} L(\vec{w}, b, \vec{\alpha}) \)

Dual: \( d^* = \max_{\alpha_i \geq 0} \min_{\vec{w}, b} L(\vec{w}, b, \vec{\alpha}) \)

Unconstrained, let’s calculate

\[
\frac{\partial}{\partial \vec{w}} L(\vec{w}, b, \vec{\alpha}) = \vec{w} - \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i
\]

\[
\Rightarrow \vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i
\]

• when \( \alpha_i > 0 \), the corresponding \( x_i \) is the support vector
• \( w \) is only a function of the support vectors!

\[
\frac{\partial}{\partial b} L(\vec{w}, b, \vec{\alpha}) = \sum_{i=1}^{n} \alpha_i y_i
\]

\[
\Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0
\]

SVM standard (primal) form:

Minimize: \( \frac{1}{2} \| \vec{w} \|^2 \) (\( w, b \))

Such that: \( y_i (\vec{w} \cdot \vec{x}_i - b) \geq 1 \) (for all \( i \))
Lagrangian:
\[ L(\vec{w}, b, \alpha) = \frac{1}{2} ||\vec{w}||^2 + \sum_{i=1}^{n} \alpha_i \left(1 - y_i (\vec{w} \cdot \vec{x}_i - b)\right) \]

Primal:  
\[ p^* = \min_{\vec{w}, b} \max_{\alpha_i \geq 0} L(\vec{w}, b, \alpha) \]

Dual:  
\[ d^* = \max_{\alpha_i \geq 0} \min_{\vec{w}, b} L(\vec{w}, b, \alpha) \]

**Unconstrained, let’s calculate**

\[ \min_{\vec{w}, b} L(\vec{w}, b, \alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \]

So:
\[ d^* = \max_{\alpha_i \geq 0} \left( \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \right) \]

subject to  
\[ \sum_{i=1}^{n} \alpha_i y_i = 0 \]
**SVM Optimization Interpretation**

**SVM standard (primal) form:**

Minimize: \( \frac{1}{2} \| \vec{w} \|^2 \)

\((w,b)\)

Such that: \( y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1 \)

(for all \(i\))

\(\text{Maximize } \gamma = 2/\|w\|\)

**SVM standard (dual) form:**

Maximize: \( \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \)

\((\alpha_i)\)

Such that: \( \sum_{i=1}^{n} \alpha_i y_i = 0 \)

\(\alpha_i \geq 0 \)

**Kernelized version**

Only a function of “support vectors”
What We Learned...

- Support Vector Machines
- Maximum Margin formulation
- Constrained Optimization
- Lagrange Duality Theory
- Convex Optimization
- SVM dual and Interpretation
- How get the optimal solution
Questions?
Next time...

Parametric and non-parametric Regression