ML (cont.): SUPPORT VECTOR MACHINES

CS540 Bryan R Gibson University of Wisconsin-Madison

Slides adapted from those used by Prof. Jerry Zhu, CS540-1
Support Vector Machines (SVMs)

The No-Math Version
Example: Lake Mendota, Madison, WI

- Identify areas of land cover (land, ice, water, snow) in a scene
- Three algorithms tried:
  - Scientist manually derived
  - Automatic best ratio
  - SVM

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Expert Derived</th>
<th>Automated Ratio</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>45.7%</td>
<td>43.7%</td>
<td>58.5%</td>
</tr>
<tr>
<td>ice</td>
<td>60.1%</td>
<td>34.3%</td>
<td>80.4%</td>
</tr>
<tr>
<td>land</td>
<td>93.6%</td>
<td>94.7%</td>
<td>94.0%</td>
</tr>
<tr>
<td>snow</td>
<td>63.5%</td>
<td>90.4%</td>
<td>71.6%</td>
</tr>
<tr>
<td>water</td>
<td>84.2%</td>
<td>74.3%</td>
<td>89.1%</td>
</tr>
<tr>
<td>unclassified</td>
<td>43.7%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

- The state-of-the-art classifier

Class Labels:
- ▪ denotes +1
- ◯ denotes -1

How would you classify this data?
Example: Linear Classifier

- If all you can do is draw a straight line, or a linear decision boundary ...
Example: Linear Classifier (cont.)

- Another “ok” decision boundary ...
Example: Linear Classifier (cont.)

- And another ...
Example: Linear Classifier (cont.)

- Any of these would work ... 

... but which is the best?
The Margin

- **Margin**: the width that the boundary can be increased to before hitting a data point . . .
The simplest SVM (linear SVM) is the linear classifier with the maximum margin...
SVM: Linearly Non-Separable Data

- What if the data is not linearly separable?
SVM: Linearly Non-Separable Data (cont.)

- Two solutions:
  - Allow a few points on the wrong side (slack variables) and/or
  - Map data to higher dimensional space, classify there (kernel)
SVM: More Than 2 Classes

- N class problem: Split the task into N binary tasks
  - Class 1 vs. the rest (Class 2 to N)
  - Class 2 vs. the rest (Classes 1, 3 to N)
  - ...
  - Class N vs. the rest (Classes 1 to N-1)
- Finally, pick the class that puts the point furthest into the positive region.
SVM: Getting your hands on it

- There are many implementations
- http://www.support-vector.net/software.html
- http://svmlight.joachims.org/
- You know enough now to use SVMs
SVM: Getting your hands on it

▶ There are many implementations
▶ http://www.support-vector.net/software.html
▶ http://svmlight.joachims.org/
▶ You know enough now to use SVMs

end of lecture?
SVM: Getting your hands on it

- There are many implementations
- http://www.support-vector.net/software.html
- http://svmlight.joachims.org/
- You know enough now to use SVMs

end of lecture?

- You need to know a little more to understand SVMs . . .
Support Vector Machines (SVMs)

The Math Version
Vectors

- A vector $\mathbf{w}$ in $d$-dimensional space is a list of $d$ numbers
  
  e.g. $\mathbf{w} = [-1, 2]' = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$

- Written as a vertical column, $[\ldots]'$ means matrix transpose

- A vector is a line segment, with direction, in space

- The norm of a vector, written $\|\mathbf{w}\|$, is its length

\[
\|\mathbf{w}\|_2 = \sqrt{\mathbf{w}'\mathbf{w}} = \sqrt{\sum_{i=1}^{d} w_i^2}
\]

\[
\mathbf{x}'\mathbf{y} = \sum_i x_i y_i \text{ (inner product)}
\]

What $d$-dimensional point $\mathbf{x}$ makes $\mathbf{w}'\mathbf{x} = 0$?
Lines

- $w'x$ is a **scalar** (single number): $x$'s projection onto $w$
- $w'x = 0$ specifies set of points $x$, which is the line perpendicular to $w$ and intersects at $(0, 0)$

$$w'x = 1 \text{ or } w'x - 1 = 0$$
$$w'x = 0$$
$$w'x = -1 \text{ or } w'x + 1 = 0$$

- $w'x = 1$ is the line parallel to $w'x = 0$, shifted by $\frac{1}{\|w\|}$
- What if the boundary doesn’t go through the origin?
SVM Boundary and Margin

- Want to find a $w$ and $b$ offset such that:
  - all positive training points $(x, y = 1)$ are in red zone
  - all negative training points $(x, y = -1)$ are in blue zone
  - margin $m$ is maximized

\[ w'x = 1 \]
\[ w'x = 0 \]
\[ w'x = -1 \]

\[ m = \frac{2}{\|w\|} \]

- How do we find $w$ and $b$?
SVM as Constrained Optimization

- **Variables**: \( w, b \)

- **Objective Function**: maximize the margin \( m = \frac{2}{\|w\|} \)

  Equiv. to **minimize** \( \|w\| \) or \( \|w\|^2 = w'w \) or \( \frac{1}{2}w'w \)

- **Subject to each training point being on the correct side** (the constraints)

- **Assume** \( n \) training points \((x_i, y_i)_{i=1:n}, y \in \{-1, 1\}\)

- **How many constraints do we have?**
SVM as Constrained Optimization (cont.)

- Variables: $w, b$
- Objective Function: maximize the margin $m = \frac{2}{\|w\|}$
- Subject to each training point being on the correct side (the constraints)

Assume $n$ training points $(x_i, y_i)_{i=1:n}$, $y \in \{-1, 1\}$

- How many constraints do we have? $n$
  - $w'x_i + b \geq 1$ if $y_i = 1$
  - $w'x_i + b \leq -1$ if $y_i = -1$
  - Can unify as $y_i(w'x_i + b) \geq 1$

- We’ve got a continuous constrained optimization problem. What do we do?
SVM as Quadratic Program (QP)

\[
\begin{align*}
\min_{w,b} & \quad \frac{1}{2} \|w\| \\
\text{s.t.} & \quad y_i(w'x_i + b) \geq 1, \quad \text{for all } i
\end{align*}
\]

- Objective is convex, quadratic
- with linear constraints
- This is known as a Quadratic Program (QP) for which efficient global solution algorithms exist
SVM: Non-Separable Data

- What if the data is not linearly separable?

- Can we insist on $y_i(w'x_i) \geq 1$, for all $i$?
SVM: Non-Separable Data - Slack Variables

- Relax the constraints – allow a few “bad apples”
- For a given linear boundary $w, b$ we can compute how “wrong” a bad point is by how far onto the wrong side it is ($\varepsilon$)

$$w'x = 1$$
$$w'x = 0$$
$$w'x = -1$$

- we relax the constraints:

$$y_i(w'x_i + b) \geq 1 - \varepsilon_i$$
We want to reduce the amount of “slack” there is in the system

$$\min_{w,b,\varepsilon} \frac{1}{2} \|w\| + c \sum_{i} \varepsilon_i$$

s.t. \( y_i(w'x_i + b) \geq 1 - \varepsilon_i \) for all \( i \)

\( \varepsilon_i \geq 0 \) for all \( i \)

The variable \( c \) is a trade-off parameter (how to set?)

Why do we require \( \varepsilon \geq 0 \)?
\[
\begin{align*}
\min_{w,b,\varepsilon} & \quad \frac{1}{2} \|w\| + c \sum_i \varepsilon_i \\
\text{s.t.} & \quad y_i(w^T x_i + b) \geq 1 - \varepsilon_i \quad \text{for all } i \\
& \quad \varepsilon_i \geq 0 \quad \text{for all } i
\end{align*}
\]

- Originally we were optimizing over \(w\) (\(d\)-dimensional) and \(b\): so \(d+1\) variables
- Now we’re optimizing over \(\varepsilon\) as well: so \(n+d+1\) variables
- With \(2n\) constraints
- Still a QP: called a Soft-Margin SVM
Here is a non-separable dataset in 1-$d$ space.
We could use slack variables, but instead we’ll use another trick . . .
Let’s map the data from 1-d to 2-d by $x \rightarrow (x, x^2)$
SVM: Non-Separable Data - Map to Higher Dimensions

- Let’s map the data from 1-d to 2-d by \( x \rightarrow (x, x^2) \)
- We’ll write this as \( \Phi(x) = (x, x^2) \)
Now the data is linearly separable in this new space!

Can run SVM in the new space without slack

linear boundary in new space $\rightarrow$ non-linearly boundary in old space
SVM: Non-Sep. Data - Another Example

\[(x_1, x_2) \rightarrow \left(x_1, x_2, \sqrt{x_1^2 + x_2^2}\right)\]
We might want to map a high dimensional example 
\(\mathbf{x} = (x_1, x_2, \ldots, x_d)\) into some much higher, 
even infinite dimensional space using \(\Phi(\mathbf{x})\)

Some problems with that:

- How do you represent infinite dimensions?
- How do we learn (among other things) \(\mathbf{w}\), 
  which lives in this new space
- Learning a large (or infinite) number of variables in a QP is 
  not a good idea
We’ll do several things to fix this:
- Convert into an equivalent QP problem, which doesn’t use \( w \) or even \( \Phi(x) \) alone!
  - Only uses inner product \( \Phi(x_i)' \Phi(x_j) \)
  - Solution only uses this inner product as well
  - This still seems infeasible for high (infinite) dimensions?

But there are smart ways to compute inner products: kernels
- kernels are a function of two variables
  - \( \text{kernel}(x_i,x_j) \Leftrightarrow \text{inner product} \Phi(x_i)'\Phi(x_j) \)

Why you should care:
- Each kernel is a new (higher dim.) space
- Fancy word to use at parties
Here’s the original QP problem:

\[
\min_{w,b} \quad \frac{1}{2} w'w \\
\text{s.t.} \quad y_i(w'x_i + b) \geq 1, \quad \text{for all } i
\]

Remember Lagrange multipliers?

\[
L = \frac{1}{2} w'w - \sum a_i [y_i(w'x_i + b) - 1] \\
\text{s.t.} \quad a_i \geq 1, \quad \text{for all } i
\]

New constraints due to original inequality constraints

We want the gradient of \(L\) to vanish w.r.t. \(w, b\) and \(a\)

We should get:

\[
w = \sum a_i y_i x_i \\
\sum a_i y_i = 0
\]

and then we stick these back into the Lagrangian \(L\) . . .
We get:

\[
\max_{a_i} \sum a_i - \frac{1}{2} \sum_{i,j} a_i a_j y_i y_j x_i' x_j
\]

s.t. \( a_i \geq 0, \) for all \( i \)

\[
\sum a_i y_i = 0
\]

This is an equivalent QP problem (the dual)

Before we were optimizing \( w \) (\( d \) variables)

Now we optimize \( a \) (\( n \) variables)

Which is better?

Important: \( x \) only appears in the inner product!
Let’s map to our new space

\[
\max_{a_i} \sum a_i - \frac{1}{2} \sum_{i,j} a_ia_jy_iy_j\Phi(x_i)'\Phi(x_j)
\]

s.t. \( a_i \geq 0 \), for all \( i \)

\[
\sum a_iy_i = 0
\]

Again, this is just an inner product

What function have we seen that we can replace this with?

The Kernel function: \( K(x_i, x_j) = \Phi(x_i)'\Phi(x_j) \)
\[
\max_{a_i} \quad \sum a_i - \frac{1}{2} \sum_{i,j} a_i a_j y_i y_j K(x_i, x_j)
\]
\[
\text{s.t.} \quad a_i \geq 0, \quad \text{for all } i \\
\sum a_i y_i = 0
\]
SVM: What's so special about kernels?

- Say data is 2-d: \( s = (s_1, s_2) \)
- We decide to use a particular mapping into 6-d space:
  \[
  \Phi(s) = (s_1^2, s_2^2, \sqrt{2}s_1s_2, \sqrt{2}s_1, \sqrt{2}s_2, 1)
  \]
SVM: What’s so special about kernels?

- Say data is 2-d: \( s = (s_1, s_2) \)
- We decide to use a particular mapping into 6-d space: 
  \[
  \Phi(s) = (s_1^2, s_2^2, \sqrt{2}s_1 s_2, \sqrt{2}s_1, \sqrt{2}s_2, 1)
  \]
- Let another point be \( t = (t_1, t_2) \), so we get the inner product: 
  \[
  \Phi(s)' \Phi(t) = s_1^2 t_1^2 + s_2^2 t_2^2 + 2s_1 s_2 t_1 t_2 + 2s_1 t_1 + 2s_2 t_2 + 1
  \]
SVM: What’s so special about kernels?

- Say data is 2-d: \( s = (s_1, s_2) \)
- We decide to use a particular mapping into 6-d space:

\[
\Phi(s) = (s_1^2, s_2^2, \sqrt{2}s_1s_2, \sqrt{2}s_1, \sqrt{2}s_2, 1)
\]

- Let another point be \( t = (t_1, t_2) \), so we get the inner product:

\[
\Phi(s)'\Phi(t) = s_1^2t_1^2 + s_2^2t_2^2 + 2s_1s_2t_1t_2 + 2s_1t_1 + 2s_2t_2 + 1
\]

- Let the kernel be \( K(s, t) = (s't + 1)^2 \)
SVM: What’s so special about kernels?

- Say data is 2-d: \( s = (s_1, s_2) \)
- We decide to use a particular mapping into 6-d space:
  \[
  \Phi(s) = (s_1^2, s_2^2, \sqrt{2}s_1s_2, \sqrt{2}s_1, \sqrt{2}s_2, 1)
  \]
- Let another point be \( t = (t_1, t_2) \), so we get the inner product:
  \[
  \Phi(s)'\Phi(t) = s_1^2t_1^2 + s_2^2t_2^2 + 2s_1s_2t_1t_2 + 2s_1t_1 + 2s_2t_2 + 1
  \]
- Let the kernel be \( K(s, t) = (s't + 1)^2 \)
- Verify that they’re the same.
  We saved on some computation!
"So is there a good kernel $K$ for any $\Phi$ that I pick?"

Mercer's condition: the inverse question is true... if for any $g(s)$ such that $\int g(s)^2 ds$ is finite we have $\int \int K(s, t) g(s) g(t) ds dt \geq 0$.

(This is positive semi-definiteness)

$\Phi$ may be infinite dimensional: we may not be able to explicitly write down $\Phi$.
“So is there a good kernel $K$ for any $\Phi$ that I pick?”

The inverse question:

“Given some $K$, is there a $\Phi$ so that $K(s, t) = \Phi(s)'\Phi(t)$?”

Mercer’s condition: the inverse question is true...
SVM: Choosing kernels

▶ “So is there a good kernel $K$ for any $\Phi$ that I pick?”
▶ The inverse question:
  “Given some $K$, is there a $\Phi$ so that $K(s, t) = \Phi(s)’\Phi(t)$?”
▶ Mercer’s condition: the inverse question is true . . .

if for any $g(s)$ such that $\int g(s)^2ds$ is finite we have

$$\int \int K(s, t)g(s)g(t)dsdt \geq 0.$$
“So is there a good kernel $K$ for any $\Phi$ that I pick?”

The inverse question:

“Given some $K$, is there a $\Phi$ so that $K(s, t) = \Phi(s)'\Phi(t)$?”

Mercer’s condition: the inverse question is true . . .

if for any $g(s)$ such that $\int g(s)^2 ds$ is finite we have

$$\int \int K(s, t)g(s)g(t)dsdt \geq 0.$$
“So is there a good kernel $K$ for any $\Phi$ that I pick?”

The inverse question:
“Given some $K$, is there a $\Phi$ so that $K(s, t) = \Phi(s)'\Phi(t)$?”

Mercer’s condition: the inverse question is true . . .

if for any $g(s)$ such that $\int g(s)^2 ds$ is finite we have

$$\int \int K(s, t)g(s)g(t)dsdt \geq 0.$$  

(This is positive semi-definiteness)

$\Phi$ may be infinite dimensional:
we may not be able to explicitly write down $\Phi$
SVM: Some frequently used kernels

- **Linear kernel:** \( K(s, t) = s't \)
- **Quadratic kernel:** \( K(s, t) = (s't + 1)^2 \)
- **Polynomial kernel:** \( K(s, t) = (s't + 1)^n \)
- **Radial Basis Function kernel:** \( K(s, t) = \exp(-\|s - t\|^2/\sigma) \)

- ...and many, many more

- Hacking with SVM: create various kernels, hope their space \( \Phi \) is meaningful, plug into SMV, pick the one with good classification accuracy (equivalent to feature engineering)

- Kernel summary:
  - QP of size \( N \), nonlinear SVM in the original space, new space in possibly high/infinite \( d \), efficient if \( K \) is easy to compute

- Kernel can be combined with slack variables
SVM: why “support vector machine”?  

▶ Remember, our problem can be written as:

$$\max_{a_i} \sum a_i - \frac{1}{2} \sum_{i,j} a_i a_j y_i y_j K(x_i, x_j)$$

s.t.  \( a_i \geq 0 \), for all \( i \)

$$\sum a_i y_i = 0$$

▶ The decision boundary is:

$$f(x_{\text{new}}) = w' x_{\text{new}} + b = \sum a_i y_i x'_i x_{\text{new}} + b$$

▶ In practice, many \( a \)'s will be zero in the solution!

▶ Those few \( x \) with \( a > 0 \) lie on the margins
  they are the “support vectors”
What you should know

- the intuition, and where to find the software
- Vector, line, length, norm
- Margin
- QP with linear constraints
- How to handle non-separable data
  - Slack variables
  - Kernels $\Leftrightarrow$ new feature space

Refs:
- A tutorial on Support Vector Machines for Pattern Recognition (1998) Christopher J. C. Burges
- An Introduction to Support Vector Machines (2000) Nello Cristianini and John Shawe-Taylor