CMSC726 Spring 2006: Support Vector Machines

readings: available from class page
sources:
Andrew Moore: http://www.cs.cmu.edu/~awm/tutorials
Tom Dietterich, Andrew Ng, Michael Littman, Rich Makin.

- 3 Views
  - Geometric
    - Maximizing Margin
  - Kernel Methods
    - Making nonlinear decision boundaries linear
    - Efficiently!
  - Capacity
    - Structural Risk Minimization

SVM History
- SVM is a classifier derived from statistical learning theory by Vapnik and Chervonenkis
- SVM was first introduced by Boser, Guyon and Vapnik in COLT-92
- SVM became famous when, using pixel maps as input, it gave accuracy comparable to NNs with hand-designed features in a handwriting recognition task
- SVM is closely related to:
  - Kernel machines (a generalization of SVMs), large margin classifiers, reproducing kernel Hilbert space, Gaussian process, Boosting

Linear Classifiers
\[ f(x,w,b) = \text{sign}(w \cdot x - b) \]

How would you classify this data?
**Linear Classifiers**

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

How would you classify this data?

**Classifier Margin**

Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

**Maximum Margin**

The maximum margin linear classifier is the linear classifier with the maximum margin. This is the simplest kind of SVM (Called an LSVM).

**Support Vectors** are those datapoints that the margin pushes up against.

**Why Maximum Margin?**

1. Intuitively this feels safest.
2. If we've made a small error in the location of the boundary (it's been jolted in its perpendicular direction) this gives us least chance of causing a misclassification.
3. LOOCV is easy since the model is immune to removal of any non-support-vector datapoints.
4. There's some theory (using VC dimension) that is related to (but not the same as) the proposition that this is a good thing.
5. Empirically it works very very well.
A “Good” Separator

Noise in the Observations

Ruling Out Some Separators

Lots of Noise

Maximizing the Margin

“Fat” Separators
Specifying a line and margin

How do we represent this mathematically?
...in $m$ input dimensions?

Plus-plane = \{ $x$ : $w \cdot x + b = +1$ \}
Minus-plane = \{ $x$ : $w \cdot x + b = -1$ \}

Claim: The vector $w$ is perpendicular to the Plus Plane. Why?

Software para resolver:

Computing the margin width

How do we compute $M$ in terms of $w$ and $b$?

Plus-plane = \{ $x$ : $w \cdot x + b = +1$ \}
Minus-plane = \{ $x$ : $w \cdot x + b = -1$ \}

Claim: The vector $w$ is perpendicular to the Plus Plane. Why?

And so of course the vector $w$ is also perpendicular to the Minus Plane. What is $w \cdot (u - v)$?

Any location in $\mathbb{R}^m$: not necessarily a datapoint

Copyright © 2001, 2003, Andrew W. Moore
Computing the margin width

The line from $x^-$ to $x^+$ is perpendicular to the planes.

So to get from $x^-$ to $x^+$ travel some distance in direction $w$.

The vector $w$ is perpendicular to the Plus Plane.

Let $x^-$ be any point on the minus plane.

Let $x^+$ be the closest plus-plane-point to $x^-$. Why?

Claim: $x^+ = x^- + \lambda w$ for some value of $\lambda$. Why?

What do we compute $M$ in terms of $w$ and $b$?

Copyright © 2001, 2003, Andrew W. Moore

Learning the Maximum Margin Classifier

Given a guess of $w$ and $b$ we can

- Compute whether all data points are in the correct half-planes
- Compute the width of the margin

So now we just need to write a program to search the space of $w$'s and $b$'s to find the widest margin that matches all the datapoints.

How?


Copyright © 2001, 2003, Andrew W. Moore

Useful Stuff

- Linear Programming

\[
\text{find } w \\
\text{argmax } c \cdot w \\
\text{subject to } \\
\begin{align*}
 w \cdot x_i & \leq b_i, & i = 1, \ldots, m \\
 w_j & \geq 0, & j = 1, \ldots, n
\end{align*}
\]

There are fast algorithms for solving linear programs including the simplex algorithm and Karmarkar’s algorithm
Learning via Quadratic Programming

- QP is a well-studied class of optimization algorithms to maximize a quadratic function of some real-valued variables subject to linear constraints.

Quadratic Programming

Find

\[ \arg \max_u c + d^T u = \frac{u^T R u}{2} \]

Subject to

\[ a_1 u_1 + a_2 u_2 + \ldots + a_n u_n \leq b_1 \]
\[ a_{n+1} u_1 + a_{n+2} u_2 + \ldots + a_{2n} u_n \leq b_2 \]
\[ a_{2n+1} u_1 + a_{2n+2} u_2 + \ldots + a_{3n} u_n \leq b_3 \]
\[ \vdots \]
\[ a_{(n-1)n+1} u_1 + a_{(n-1)n+2} u_2 + \ldots + a_{n^2} u_n \leq b_{n-1} \]

And subject to

\[ a_{(n-1)n+1} u_1 + a_{(n-1)n+2} u_2 + \ldots + a_{n^2} u_n = b_{n^2} \]
\[ a_{(n-2)n+1} u_1 + a_{(n-2)n+2} u_2 + \ldots + a_{n^2} u_n = b_{n^2-1} \]
\[ \vdots \]
\[ a_{n(n-1)+1} u_1 + a_{n(n-1)+2} u_2 + \ldots + a_{n^2} u_n = b_{2n-1} \]

Quadratic criterion

\[ \text{Minimize } w \cdot w \]

Compute whether all data points are in the correct half-planes:

- If \( y_k = 1 \)
  \[ w \cdot x_k + b \geq 1 \]
- If \( y_k = -1 \)
  \[ w \cdot x_k + b \leq -1 \]

What should our quadratic optimization criterion be?

Minimize \( w \cdot w \)

How many constraints will we have?

- Given guess of \( w, b \) we can compute whether all data points are in the correct half-planes.
- Compute the margin width
  Assume \( R \) datapoints, each \((x_k, y_k)\) where \( y_k = +/- 1 \)

What should our quadratic optimization criterion be?

Minimize \( w \cdot w \)

How many constraints will we have?

- \( R \) datapoints
- Compute whether all data points are in the correct half-planes:
  - If \( y_k = 1 \)
    \[ w \cdot x_k + b \geq 1 \]
  - If \( y_k = -1 \)
    \[ w \cdot x_k + b \leq -1 \]

This is going to be a problem!

What should we do?

\* denotes +1
\* denotes -1

Yay, we're done!!
This is going to be a problem! What should we do?

Idea 1:
Find minimum $\mathbf{w}\cdot\mathbf{w}$ while minimizing number of training set errors.

Problem: Two things to minimize makes for an ill-defined optimization.

---

Idea 1.1:
Minimize $\mathbf{w}\cdot\mathbf{w} + C$ (#train errors)

There's a serious practical problem that's about to make us reject this approach. Can you guess what it is?

Tradeoff parameter

Can't be expressed as a Quadratic Programming problem. Solving it may be too slow. (Also, doesn't distinguish between disastrous errors and near misses)

So... any other ideas?

---

Learning Maximum Margin with Noise

Given guess of $\mathbf{w}$, $\mathbf{b}$ we can
- Compute sum of distances of points to their correct zones
- Compute the margin width

Assume $R$ datapoints, each $(\mathbf{x}_k, y_k)$ where $y_k = \pm 1$

$M = \frac{1}{2} \mathbf{w}\cdot\mathbf{w}$

What should our quadratic optimization criterion be?

How many constraints will we have?

What should they be?
Learning Maximum Margin with Noise

Given guess of $w$, $b$ we can
- Compute sum of distances of points to their correct zones
- Compute the margin width
Assume $R$ datapoints, each $(x_k, y_k)$ where $y_k = \pm 1$

$wx + b = 1$
$wx + b = 0$
$wx + b = -1$

$M = \langle w, w \rangle$

What should our quadratic optimization criterion be?

Minimize
$\frac{1}{2} \langle w, w \rangle + C \sum_{k=1}^{R} \epsilon_k$

How many constraints will we have?
$R$

What should they be?
$w \cdot x_k + b \geq 1 - \epsilon_k$ if $y_k = 1$
$w \cdot x_k + b \leq -1 + \epsilon_k$ if $y_k = -1$

There's a bug in this QP. Can you spot it?

Copyright © 2001, 2003, Andrew W. Moore

An Equivalent Dual QP

Maximize $\sum_{k=1}^{R} y_{i,k} \alpha_{i,k}$ where $Q_{i,k} = y_{i,k} (x_i, x_k)$

Subject to these constraints:
$0 \leq \alpha_{i,k} \leq C \quad \forall k$
$\sum_{k=1}^{R} \alpha_{i,k} y_{i,k} = 0$

Then define:
$w = \sum_{k=1}^{R} \alpha_{i,k} y_{i,k} x_k$
$b = y_{\star, k} (1 - \epsilon_k) - x_k \cdot w_k$
$\epsilon_k = \text{arg max} \alpha_{i,k}$

Then classify with:
$f(x, w, b) = \text{sign}(w \cdot x - b)$

Yay, we're done!!
Maximize where $\sum_k a_k - \frac{1}{2} \sum_k \sum_{j=k} a_i a_j Q_{ij} = y_k y_j (x_k \cdot x_j)$

Subject to these constraints: $0 \leq a_k \leq C \quad \forall k \quad \sum_k a_k y_k = 0$

Then define:

$$w = \sum_k a_k y_k x_k$$

$$b = y_K \left(1 - e_K\right) - x_K \cdot w$$

where $K = \text{arg max} a_k$

Datapoints with $a_k > 0$ will be the support vectors.

An Equivalent Dual QP

Why did I tell you about this equivalent QP?

• It's a formulation that QP packages can optimize more quickly
• Because of further developments you're about to learn.

Suppose we're in 1-dimension

What would SVMs do with this data?

Not a big surprise

Harder 1-dimensional dataset

That's wiped the smirk off SVM's face.

What can be done about this?

Remember how permitting non-linear basis functions made linear regression so much nicer?

Let's permit them here too

$z_k = (x_k, x_k^2)$
Maximize

Then define:

Subject to these constraints:

We must do $R^2/2$ dot products to get this matrix ready.
Each dot product requires $m^2/2$ additions and multiplications.
The whole thing costs $R^2 m^2/4$. Yeeks!

...or does it?

Then classify with:

Then define:

$w = \sum_{k: \alpha_k > 0} \alpha_k y_j \Phi(\mathbf{x}_k)$

$b = y_K (1 - e_K) - \mathbf{x}_K \cdot \mathbf{w}_K$

where $K = \arg \max \alpha_k$.

You may be wondering what those $\sqrt{2}$'s are doing.

You should be happy that they do no harm.
You'll find out why they're there soon.

This is sensible.
Is that the end of the story?
No...there's one more trick!
Maximize \( \sum a_i y_i (\Phi(x_i), \Phi(x_i)) \) subject to these constraints: 
\[ 0 \leq a_i \leq 1 \]

Then define:
\[ w = \sum a_i y_i \Phi(x_i) \]
\[ b = y_K (1 - \epsilon_K) - x_K \cdot w_K \]
where \( K = \arg \max a_i \)

We must do \( R/2 \) dot products to get this matrix ready. 
In 100-d, each dot product now needs 103 operations instead of 75 million. 
But there are still worrying things lurking away. 
What are they?

Then define:
\[ w = \sum a_i y_i \Phi(x_i) \]
\[ b = y_K (1 - \epsilon_K) - x_K \cdot w_K \]
where \( K = \arg \max a_i \)

Higher Order Polynomials

<table>
<thead>
<tr>
<th>Polynomial</th>
<th>Cost to build Q matrix traditionally</th>
<th>Cost to build Q matrix efficiently</th>
<th>Cost if 100 inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quadratic</td>
<td>( m^2 \cdot R^2 )</td>
<td>( m \cdot R )</td>
<td>( m \cdot R )</td>
</tr>
<tr>
<td>Cubic</td>
<td>( m^3 \cdot R^2 )</td>
<td>( 81,000 \cdot R^2 )</td>
<td>( 50 \cdot R^2 )</td>
</tr>
<tr>
<td>Quartic</td>
<td>( m^4 \cdot R^2 )</td>
<td>( 1,960,000 \cdot R^2 )</td>
<td>( 50 \cdot R^2 )</td>
</tr>
</tbody>
</table>

\( \Phi(x) = (x, x^2, x^3, ..., x^d) \)
What are they? In 100-d, each dot product now needs 10^3 operations instead of 75 million matrix ready. But there are still worrying things lurking away. We must do R^2/2 dot products to get this matrix ready.

Then define:
\[
w = \sum_{k \text{ s.t. } a_k > 0} a_k y_k \Phi(x_k)
\]
and:
\[
b = y_k (1 - e_k) - x_k \cdot w
\]
where \( K = \arg \max_{a_k} a_k \).

Then classify with:
\[
f(x; w, b) = \text{sign}(w \cdot \Phi(x) - b)
\]

What Makes a Kernel

Mercer’s theorem characterizes when a function \( f(\tilde{x}, \tilde{z}) \) is a kernel

If \( K_1() \) and \( K_2() \) are kernels, then so are:
1) \( K_1() + K_2() \)
2) \( c \cdot K_1() \) where \( c \) is constant
3) \( K_1() \ast K_2() \)
4) \( f(\tilde{x}) \ast f(\tilde{z}) \) where \( f() \) returns a real
### Key SVM Ideas
- Maximize the **margin** between positive and negative examples (connects to PAC theory)
- Penalize errors in non-separable case
- Only the **support vectors** contribute to the solution
- Kernels map examples into a new, usually non-linear space
  - We implicitly do dot products in this new space (in the “dual” form of the SVM program)

### SVM Performance
- Anecdotally they work very very well indeed.
- Example: They are currently the best-known classifier on a well-studied hand-written-character recognition benchmark
- Another Example: AWM knows several reliable people doing practical real-world work who claim that SVMs have saved them when their other favorite classifiers did poorly.
- There is a lot of excitement and religious fervor about SVMs and Kernel machines as of 2004.
- Despite this, some practitioners are a little skeptical.

### Doing multi-class classification
- SVMs can only handle two-class outputs (i.e. a categorical output variable with arity 2).
- What can be done?
  - Answer: with output arity N, learn N SVM’s
    - SVM 1 learns "Output==1" vs "Output != 1"
    - SVM 2 learns "Output==2" vs "Output != 2"
    - ...
    - SVM N learns "Output==N" vs "Output != N"
- Then to predict the output for a new input, just predict with each SVM and find out which one puts the prediction the furthest into the positive region.

### SVM Implementations
- Sequential Minimal Optimization, SMO, efficient implementation of SVMs, Platt
  - in Weka
- SVM^light
  - [http://svmlight.joachims.org/](http://svmlight.joachims.org/)
- LibSVM

### References
- Tutorial on VC-dimension and Support Vector Machines:
  - [http://citeseer.nj.nec.com/burges98tutorial.html](http://citeseer.nj.nec.com/burges98tutorial.html)
- The VC/SRM/SVM Bible: