Support Vector Machines

Modified from Prof. Andrew W. Moore’s Lecture Notes
www.cs.cmu.edu/~awm

History

• SVM is a classifier derived from statistical learning theory by Vapnik and Chervonenkis
• SVMs introduced by Boser, Guyon, Vapnik in COLT-92
• Now an important and active field of all Machine Learning research.

Linear Classifiers

How would you classify this data?

e.g. \( x = (x_1, x_2), w = (w_1, w_2), w.x = x_1w_1 + x_2w_2 \)
\[ \text{sign}(w.x + b) = +1 \text{ iff } x_1w_1 + x_2w_2 - b > 0 \]
Linear Classifiers

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

\[ \text{sign}(w \cdot x + b) = +1 \text{ iff } x^T w_1 + w_2 x_2 + b > 0 \]

Any of these would be fine...

...but which is best?

Classifier Margin

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

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, um, 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. Empirically it works very well.
3. 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.
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.

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

Estimate the Margin

\[ d(x) = \frac{|x \cdot w + b|}{\sqrt{w^T w}} = \frac{|x \cdot w + b|}{\sqrt{\sum_{i=1}^{d} w_i^2}} \]

\[ X \text{ – Vector} \]
\[ W \text{ – Normal Vector} \]
\[ b \text{ – Scale Value} \]

http://mathworld.wolfram.com/Point-LineDistance2-Dimensional.html
Estimate the Margin

- What is the expression for margin, given $w$ and $b$?

$$\text{margin} = \arg \min_{w \in D} \arg \min_{x \in D} \frac{k \cdot x + b}{\sum_{i=1}^{d} w_i^2}$$

Maximize Margin

$$\text{argmax } \text{margin}(w, b, D) = \text{argmax } \arg \min_{x \in D} \frac{t \cdot x + b}{\sum_{i=1}^{d} w_i^2}$$

Maximize Margin

$$\text{argmax } \arg \min_{x \in D} \frac{t \cdot x + b}{\sum_{i=1}^{d} w_i^2}$$

Maximize Margin

$$\text{argmax } \text{margin}(w, b, D) = \text{argmax } \arg \min_{x \in D} \frac{t \cdot x + b}{\sum_{i=1}^{d} w_i^2}$$

Maximize Margin

$$\text{argmax } \arg \min_{x \in D} \frac{t \cdot x + b}{\sum_{i=1}^{d} w_i^2}$$

Maximize Margin

$$\text{argmax } \arg \min_{x \in D} \frac{t \cdot x + b}{\sum_{i=1}^{d} w_i^2}$$

Maximum Margin Linear Classifier

$$\text{argmax} \sum_{i=1}^{d} w_i^2$$

subject to

$$y_1 (\frac{w \cdot x_1 + b}{\sqrt{\sum_{i=1}^{d} w_i^2}}) \geq 1$$

$$y_2 (\frac{w \cdot x_2 + b}{\sqrt{\sum_{i=1}^{d} w_i^2}}) \geq 1$$

$$\ldots$$

$$y_{\text{num_classes}} (\frac{w \cdot x_{\text{num_classes}} + b}{\sqrt{\sum_{i=1}^{d} w_i^2}}) \geq 1$$

- How to solve it?
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.
- Detail solution of Quadratic Programming
  - Convex Optimization Stephen P. Boyd
  - Online Edition, Free for Download
  - www.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf

Quadratic Programming

\[
\begin{align*}
\max_{u} & \quad c + d^T u + \frac{u^T u}{2} \\
\text{subject to} & \quad a_1 u_1 + a_2 u_2 + \ldots + a_n u_n \leq b_1 \\
& \quad a_{n+1} u_1 + a_{n+2} u_2 + \ldots + a_n u_n \leq b_n \\
& \quad n \text{ additional linear inequality constraints}
\end{align*}
\]

Quadratic Programming for the Linear Classifier

\[
\begin{align*}
\{w^*, b^*\} = \min_{w,b} & \quad \sum \eta_i \\
\text{subject to} & \quad y_i (w^T x_i + b) \geq 1 \\
& \quad y_i (w^T x_i + b) \geq 1 \quad \text{inequality constraints}
\end{align*}
\]

\[
\{w^*, b^*\} = \max_{w,b} \left\{ 0 \mid w^T x_i + b \leq 1, i = 1, \ldots, l \right\}
\]

Uh-oh! This is going to be a problem! What should we do?

- denotes +1
- denotes -1

Popular Tools - LibSVM

Idea 1:

Find minimum \(w,w\), while minimizing number of training set errors. Problem: Two things to minimize makes for an ill-defined optimization.

Uh-oh! This is going to be a problem! What should we do?

- denotes +1
- denotes -1
Uh-oh!
This is going to be a problem!
What should we do?
Idea 1.1:
Minimize
\[ w \cdot w + C \text{ (#train errors)} \]
Tradeoff parameter
There’s a serious practical problem that’s about to make us reject this approach. Can you guess what it is?

Uh-oh!
This is going to be a problem!
What should we do?
Idea 1.1:
Minimize
\[ w \cdot w + C \text{ (#train errors)} \]
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?

Uh-oh!
This is going to be a problem!
What should we do?
Idea 2.0:
Minimize
\[ w \cdot w + C \text{ (distance of error points to their correct place)} \]
Support Vector Machine (SVM) for Noisy Data
\[
\{w^*, b^*\} = \min_{w, b} \sum_{i=1}^{d} y_i^2 + \sum_{j=1}^{N} \varepsilon_j
\]
\[
y_1 (\mathbf{w} \cdot \mathbf{x}_1 + b) \geq 1 - \varepsilon_1, e_1 \geq 0
\]
\[
y_2 (\mathbf{w} \cdot \mathbf{x}_2 + b) \geq 1 - \varepsilon_2, e_2 \geq 0
\]
\[
\vdots
\]
\[
y_N (\mathbf{w} \cdot \mathbf{x}_N + b) \geq 1 - \varepsilon_N, e_N \geq 0
\]
• Balance the trade off between margin and classification errors

Support Vector Machine for Noisy Data
\[
\{w^*, b^*\} = \arg \min_{w, b} \sum_{i=1}^{d} y_i^2 + \sum_{j=1}^{N} \varepsilon_j
\]
\[
y_1 (\mathbf{w} \cdot \mathbf{x}_1 + b) \geq 1 - \varepsilon_1, e_1 \geq 0
\]
\[
y_2 (\mathbf{w} \cdot \mathbf{x}_2 + b) \geq 1 - \varepsilon_2, e_2 \geq 0
\]
\[
\vdots
\]
\[
y_N (\mathbf{w} \cdot \mathbf{x}_N + b) \geq 1 - \varepsilon_N, e_N \geq 0
\]
inequality constraints
How do we determine the appropriate value for \(c\)?
The Dual Form of QP

Maximize \( \sum_{k=1}^{N} \alpha_k - \frac{1}{2} \sum_{k,l=1}^{N} \alpha_k \alpha_l Q_{kl} \), where \( Q_{kl} = y_k y_l (x_k, x_l) \)

Subject to these constraints:
\( 0 \leq \alpha_k \leq C \quad \forall k \)
\( \sum_{k=1}^{N} \alpha_k y_k = 0 \)

Then define:
\( w = \sum_{k=1}^{N} \alpha_k y_k x_k \)

Support Vectors

\( \alpha_k = 0 \) for non-support vectors
\( \alpha_k \neq 0 \) for support vectors

\( w = \sum_{k=1}^{N} \alpha_k y_k x_k \)

Decision boundary is determined only by those support vectors!

An Equivalent QP

Maximize \( \sum_{k=1}^{N} \alpha_k - \frac{1}{2} \sum_{k,l=1}^{N} \alpha_k \alpha_l Q_{kl} \), where \( Q_{kl} = y_k y_l (x_k, x_l) \)

Subject to these constraints:
\( 0 \leq \alpha_k \leq C \quad \forall k \)
\( \sum_{k=1}^{N} \alpha_k y_k = 0 \)

Then define:
\( w = \sum_{k=1}^{N} \alpha_k y_k x_k \)

An Equivalent QP: Determine \( b \)

\[ \begin{align*}
\begin{cases}
\widehat{b}^* = \arg\min_{\epsilon, b} \sum_{i=1}^{N} w_i^2 + \epsilon \sum_{i=1}^{N} \eta_i \\
y_1 \left( \frac{w^T x_1 + b}{\sqrt{w^T w}} \right) \geq 1 - \epsilon_i \alpha_i \geq 0 \\
y_2 \left( \frac{w^T x_2 + b}{\sqrt{w^T w}} \right) \geq 1 - \epsilon_i \alpha_i \geq 0 \\
\vdots \\
y_N \left( \frac{w^T x_N + b}{\sqrt{w^T w}} \right) \geq 1 - \epsilon_i \alpha_i \geq 0
\end{cases}
\end{align*} \]

Fix \( w \)

A linear programming problem!

Parameter \( c \) is used to control the fitness
Feature Transformation?

- The problem is non-linear
- Find some trick to transform the input
- Linear separable after Feature Transformation
- What features should we use?

Basic Idea:

Suppose we're in 1-dimension

What would SVMs do with this data?

Suppose we're in 1-dimension

Not a big surprise

Harder 1-dimensional dataset

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

What can be done about this?

Harder 1-dimensional dataset

Map the data from low-dimensional space to high-dimensional space

Feature Enumeration:

\[ z_k = (x_k, x_k^2) \]
Non-linear SVMs: Feature spaces

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:

$$\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^N$$

$$(x_1, x_2) \rightarrow (x_1, x_2, x_1^2, \sqrt{3}x_1x_2, x_2^2)$$

- Polynomial features for the XOR problem

Efficiency Problem in Computing Feature

- Feature space Mapping

Proposes the data with:

$$\Phi : X \rightarrow \mathcal{N}$$

where $\mathcal{N}$ is a dot product space, and learn the mapping from $\Phi(x)$ to $y$.

- Example: all 2 degree Monomials

$$\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^N$$

$$(x_1, x_2) \rightarrow (x_1, x_2, x_1^2, \sqrt{3}x_1x_2, x_2^2)$$

$$(\Phi(x), \Phi(y)) = \left( x_1 y_1, \sqrt{3} x_1 y_2, x_1^2 y_1, \sqrt{3} x_1 x_2 y_2, x_2^2 y_2 \right)^T$$

This use of kernel function to avoid carrying out $\Phi(x)$ explicitly is known as the kernel trick.

- $n$ Multiplication

$=$ $3$ Multiplication

$\rightarrow$ the dot product in $\mathcal{N}$ can be computed in $\mathbb{R}^2$

Common SVM basis functions

- $z_k = (polynomial \ terms \ of \ x_k \ of \ degree \ 1 \ to \ q)$

- $z_k = (radial \ basis \ functions \ of \ x_k)$

- $z_k = (sigmoid \ functions \ of \ x_k)$

“Radius basis functions” for the XOR problem

- Could solve complicated Non-Linear Problems

- $y$ and $c$ control the complexity of decision boundary

$$y = \frac{1}{\sigma^2}$$
How to Control the Complexity

Which reasoning below is the most probable?

- Bob got up and found that breakfast was ready (Underfitting)
- His Child
- His Wife (Reasonable)
- The Alien (Overfitting)

How to Control the Complexity

- SVM is powerful to approximate any training data
- The complexity affects the performance on new data
- SVM supports parameters for controlling the complexity
- SVM does not tell you how to set these parameters
- Determine the Parameters by Cross-Validation

SVM Performance

- Anecdotally they work very well indeed.
- Example: They are currently the best-known classifier on a well-studied hand-written-character recognition benchmark
- There is a lot of excitement and religious fervor about SVMs.
- Despite this, some practitioners are a little skeptical.

References

- An excellent tutorial on VC-dimension and Support Vector Machines:
  http://citeseer.nj.nec.com/burges98tutorial.html
- The VC/SRM/SVM Bible: (Not for beginners including myself)
  LibSVM, http://www.csie.ntu.edu.tw/~cjlin/libsvm/
  SMO in Weka