Support Vector Machines and their Applications

Purushottam Kar

Department of Computer Science and Engineering,
Indian Institute of Technology Kanpur.

Summer School on "Expert Systems And Their Applications",
Indian Institute of Information Technology Allahabad.
June 14, 2009

Support Vector Machines

- What is being “supported” ?

Support Vector Machines

- What is being "supported" ?
- How can vectors support anything ?

Support Vector Machines

- What is being “supported” ?
- How can vectors support anything ?
- Wait !! Machines ?? - Is this a Mechanical Engineering Lecture ?

The Learning Methodology

Is it possible to write an algorithm to distinguish between ...

The Learning Methodology

Is it possible to write an algorithm to distinguish between ...

- a well-formed and an ill-formed C++ program ?

The Learning Methodology

Is it possible to write an algorithm to distinguish between ...

- a well-formed and an ill-formed C++ program ?
- a palindrome and a non-palindrome ?

The Learning Methodology

Is it possible to write an algorithm to distinguish between ...

- a well-formed and an ill-formed C++ program ?
- a palindrome and a non-palindrome ?
- a graph with and without cliques of size bigger than 1000 ?

The Learning Methodology

Is it possible to write an algorithm to distinguish between ...

- a handwritten 4 and a handwritten 9 ?

The Learning Methodology

Is it possible to write an algorithm to distinguish between ...

- a handwritten 4 and a handwritten 9 ?
- a spam and a non-spam e-mail ?

The Learning Methodology

Is it possible to write an algorithm to distinguish between ...

- a handwritten 4 and a handwritten 9 ?
- a spam and a non-spam e-mail ?
- a positive movie review and a negative movie review ?

Statistical Machine Learning

- “Synthesize” a program based on training data

Statistical Machine Learning

- "Synthesize" a program based on training data
- Assume training data that is randomly generated from some unknown but fixed distribution and a target function

Statistical Machine Learning

- “Synthesize” a program based on training data
- Assume training data that is randomly generated from some unknown but fixed distribution and a target function
- Give probabilistic error bounds

Statistical Machine Learning

- "Synthesize" a program based on training data
- Assume training data that is randomly generated from some unknown but fixed distribution and a target function
- Give probabilistic error bounds
- In other words be probably-approximately-correct

Statistical Machine Learning

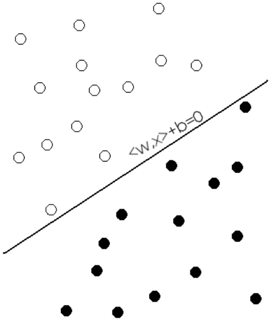
- "Synthesize" a program based on training data
- Assume training data that is randomly generated from some unknown but fixed distribution and a target function
- Give probabilistic error bounds
- In other words be probably-approximately-correct
- The motto - Let the data decide the algorithm

Expert Systems

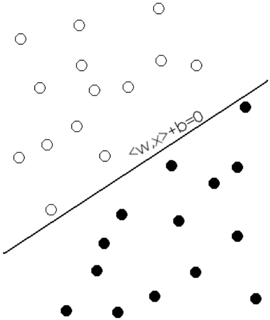
... a computing system capable of representing and reasoning about some knowledge rich domain, such as internal medicine or geology ...

Introduction to Expert Systems, Peter Jackson, Addison Wesley Publishing Company, 1986.

Linear Machines

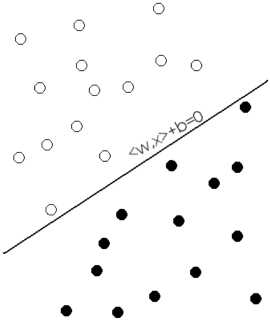


Linear Machines



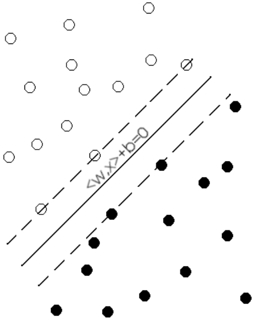
- Arguably the simplest of classifiers acting on vectoral data

Linear Machines

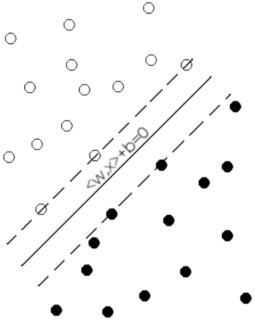


- Arguably the simplest of classifiers acting on vectoral data
- Numerous Learning Algorithms - Perceptron, SVM

Support Vector Machines

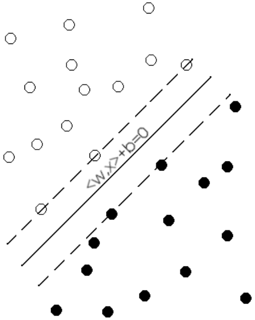


Support Vector Machines



- A "special" hyperplane - with the maximum margin

Support Vector Machines



- A "special" hyperplane - with the maximum margin
- Margin of a point measures how far is it from the hyperplane

Learning the Maximum Margin Classifier

$$
\begin{aligned}
\text{minimize}_{\vec{w},b} \quad & \|\vec{w}\|_2 \\
\text{subject to} \quad & y_i(\langle \vec{w} \cdot \vec{x_i} \rangle + b) \geq 1, \\
& i = 1, \ldots, l.
\end{aligned}
$$

- A *Linearly-constrained Quadratic program*

Learning the Maximum Margin Classifier

$$
\begin{aligned}
\text{minimize}_{\vec{w},b} \quad & \|\vec{w}\|_2 \\
\text{subject to} \quad & y_i(\langle \vec{w} \cdot \vec{x_i} \rangle + b) \geq 1, \\
& i = 1, \ldots, l.
\end{aligned}
$$

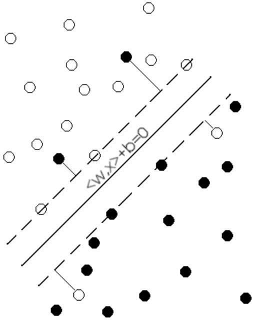
- A *Linearly-constrained Quadratic program*
- Solvable in polynomial time - several algorithms known

Learning the Maximum Margin Classifier

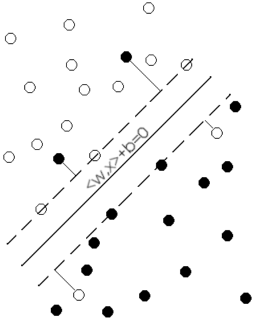
$$\begin{aligned} \text{minimize}_{\vec{w},b} \quad & \|\vec{w}\|_2 \\ \text{subject to} \quad & y_i(\langle \vec{w} \cdot \vec{x_i} \rangle + b) \geq 1, \\ & i = 1, \ldots, l. \end{aligned}$$

- A *Linearly-constrained Quadratic program*
- Solvable in polynomial time - several algorithms known
- Does not give us much insight into the nature of the hyperplane

Non-linearly Separable Data

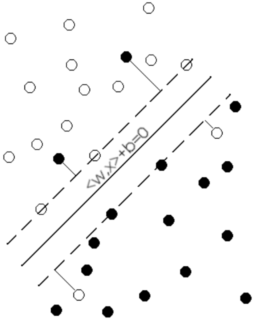


Non-linearly Separable Data



- Use slack variables to allow points to lie on the “wrong” side of the hyperplane

Non-linearly Separable Data



- Use slack variables to allow points to lie on the “wrong” side of the hyperplane
- Can still be solved using a QCQP

Learning the Soft Margin Classifier

$$\begin{aligned}
\text{minimize}_{\vec{w},b} \quad & \|\vec{w}\|_2 + C\sum_{i=1}^{l}\xi_i^2 \\
\text{subject to} \quad & y_i(\langle \vec{w}\cdot\vec{x}_i\rangle + b) \geq 1 - \xi_i, \\
& i = 1,\ldots,l.
\end{aligned}$$

- Again a *Linearly-constrained Quadratic program*

Learning the Soft Margin Classifier

$$\begin{aligned} \text{minimize}_{\vec{w},b} \quad & \|\vec{w}\|_2 + C\sum_{i=1}^{l}\xi_i^2 \\ \text{subject to} \quad & y_i(\langle \vec{w}\cdot\vec{x_i}\rangle + b) \geq 1 - \xi_i, \\ & i = 1,\ldots,l. \end{aligned}$$

- Again a *Linearly-constrained Quadratic program*
- More insight gained by looking at the *dual program*

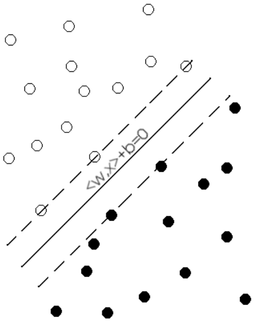
The Dual Program for the Hard Margin SVM

$$\text{maximize} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \vec{x_i} \cdot \vec{x_j} \rangle,$$

$$\text{subject to} \quad \sum_{i=1}^{l} y_i \alpha_i = 0,$$

$$\alpha_1 \geq 0, i = 1, \ldots, l.$$

Some properties of the optimum $(\vec{w}^*, b^*, \vec{\alpha}^*)$:



The Dual Program for the Hard Margin SVM

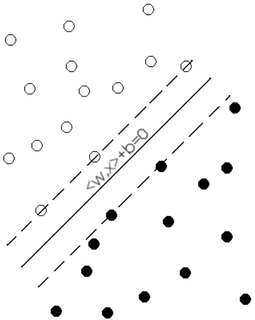
$$\text{maximize} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \vec{x_i} \cdot \vec{x_j} \rangle,$$

$$\text{subject to} \quad \sum_{i=1}^{l} y_i \alpha_i = 0,$$

$$\alpha_1 \geq 0, i = 1, \ldots, l.$$

Some properties of the optimum $(\vec{w}^*, b^*, \vec{\alpha}^*)$:

- $\alpha_i^* [y_i(\langle \vec{w}^* \cdot \vec{x_i} \rangle + b^*) - 1] = 0, i = 1, \ldots, l$

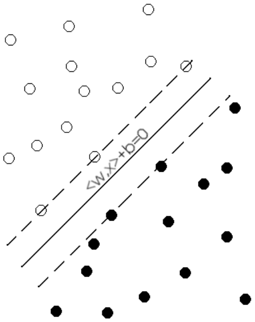


The Dual Program for the Hard Margin SVM

$$\text{maximize} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \vec{x_i} \cdot \vec{x_j} \rangle,$$

$$\text{subject to} \quad \sum_{i=1}^{l} y_i \alpha_i = 0,$$

$$\alpha_1 \geq 0, i = 1, \ldots, l.$$



Some properties of the optimum $(\vec{w}^*, b^*, \vec{\alpha}^*)$:

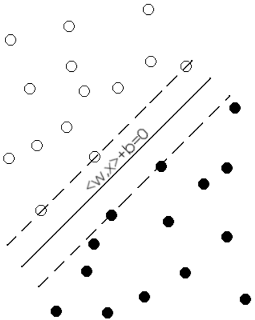
- $\alpha_i^*[y_i(\langle \vec{w}^* \cdot \vec{x_i} \rangle + b^*) - 1] = 0, i = 1, \ldots, l$
- $\vec{w}^* = \sum_{i=1}^{l} y_i \alpha_i^* \vec{x_i}$

The Dual Program for the Hard Margin SVM

$$\text{maximize} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \vec{x_i} \cdot \vec{x_j} \rangle,$$

$$\text{subject to} \quad \sum_{i=1}^{l} y_i \alpha_i = 0,$$

$$\alpha_1 \geq 0, i = 1, \ldots, l.$$



Some properties of the optimum $(\vec{w}^*, b^*, \vec{\alpha}^*)$:

- $\alpha_i^* [y_i(\langle \vec{w}^* \cdot \vec{x_i} \rangle + b^*) - 1] = 0, i = 1, \ldots, l$
- $\vec{w}^* = \sum_{i=1}^{l} y_i \alpha_i^* \vec{x_i}$
- $f(\vec{x}, \vec{\alpha}^*, b^*) = \sum_{i=1}^{l} y_i \alpha_i^* \langle \vec{x_i} \cdot \vec{x} \rangle + b^*$

The Support

- Consider each vector applying a force of α_i on the hyperplane in the direction $y_i \frac{\vec{w}}{\|\vec{w}\|_2}$.

The Support

- Consider each vector applying a force of α_i on the hyperplane in the direction $y_i \frac{\vec{w}}{\|\vec{w}\|_2}$.
- The conditions exactly correspond to the force and the torque on the hyperplane being zero

The Support

- Consider each vector applying a force of α_i on the hyperplane in the direction $y_i \frac{\vec{w}}{\|\vec{w}\|_2}$.
- The conditions exactly correspond to the force and the torque on the hyperplane being zero
- Hence the vectors lying on the margin, for whom $\alpha_i \neq 0$, "support" the hyperplane.

Luckiness and Generalization

- Essentially the idea is to show that the probability of the training sample misleading us into learning an erroneous classifier is small

Luckiness and Generalization

- Essentially the idea is to show that the probability of the training sample misleading us into learning an erroneous classifier is small
- To do this model selection has to be done carefully

Luckiness and Generalization

- Essentially the idea is to show that the probability of the training sample misleading us into learning an erroneous classifier is small
- To do this model selection has to be done carefully
- The classifier family should not be too powerful to prevent overfitting

Bounds for Linear Classifiers

Theorem (Vapnik and Chervonenkis)

For any probability distribution on the input domain $\mathbb{R}^d$*, and any target function* g*, with probability no less than* $1-\delta$*, any linear hyperplane* f *classifying a randomly chosen training set of size* l *perfectly cannot disagree with the target function on more than* ϵ *fraction of the input domain (with respect to the underlying distribution) where*

$$\epsilon = \frac{2}{l}\left((d+1)\log\frac{2el}{d+1} + \log\frac{2}{\delta}\right)$$

Bounds for Large Margin Classifiers

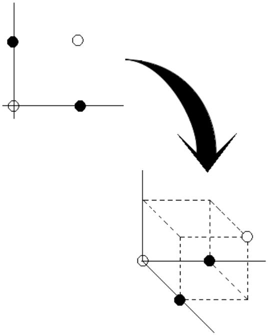
Theorem (Vapnik)

For any probability distribution on the input domain - a ball of radius R, and any target function g, with probability no less than $1-\delta$, any linear hyperplane f classifying a randomly chosen training set of size l perfectly with margin $\geq \gamma$ cannot disagree with the target function on more than ϵ fraction of the input domain (with respect to the underlying distribution) where

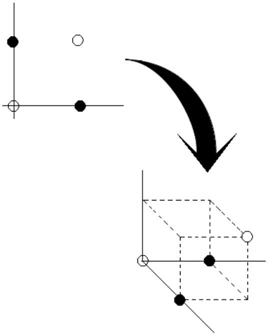
$$\epsilon = \tilde{O}\left(\frac{1}{l}\left(\frac{R^2}{\gamma^2} + \log\frac{1}{\delta}\right)\right)$$

NOTE : Error bound Independent of the dimension !

The XOR problem

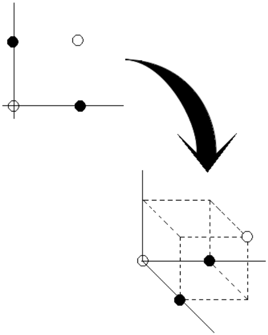


The XOR problem



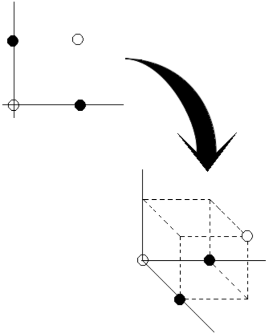
- The feature map $\Phi : (x, y) \longmapsto (x^2, y^2, \sqrt{2}xy)$ makes the problem linearly a separable one.

The XOR problem



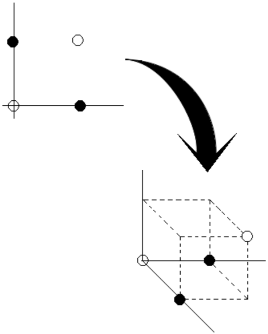
- The feature map $\Phi : (x, y) \longmapsto (x^2, y^2, \sqrt{2}xy)$ makes the problem linearly a separable one.
- But do we need to perform the actual feature map ?

The XOR problem



- The feature map $\Phi : (x, y) \longmapsto (x^2, y^2, \sqrt{2}xy)$ makes the problem linearly a separable one.
- But do we need to perform the actual feature map ?
- No ! Since all that the SVM training algorithm requires are values of $\langle \Phi(\vec{x_i}) \cdot \Phi(\vec{x_j}) \rangle$

The XOR problem



- The feature map $\Phi : (x, y) \longmapsto (x^2, y^2, \sqrt{2}xy)$ makes the problem linearly a separable one.
- But do we need to perform the actual feature map ?
- No ! Since all that the SVM training algorithm requires are values of $\langle \Phi(\vec{x_i}) \cdot \Phi(\vec{x_j}) \rangle$
- But $\langle \Phi(\vec{x_i}) \cdot \Phi(\vec{x_j}) \rangle = \langle \vec{x_i} \cdot \vec{x_j} \rangle^2$

The Kernel Trick

- Many algorithms admit the Kernel trick - SVM, SVM-regression, Kernel-PCA, Kernel-clustering, Perceptron

The Kernel Trick

- Many algorithms admit the Kernel trick - SVM, SVM-regression, Kernel-PCA, Kernel-clustering, Perceptron
- However kernel trick not used for the Perceptron algorithm - why ?

The Kernel Trick

- Many algorithms admit the Kernel trick - SVM, SVM-regression, Kernel-PCA, Kernel-clustering, Perceptron
- However kernel trick not used for the Perceptron algorithm - why ?
- Note that not all kernels correspond to feature maps

Implementations

- Many techniques known - Ellipsoid method, Gradient descent, Sequential Minimal Optimization - the latter is tuned to the QP problem

Implementations

- Many techniques known - Ellipsoid method, Gradient descent, Sequential Minimal Optimization - the latter is tuned to the QP problem
- LIBSVM `http://www.csie.ntu.edu.tw/~cjlin/libsvm/` - supports multi-class classification and regression

Implementations

- Many techniques known - Ellipsoid method, Gradient descent, Sequential Minimal Optimization - the latter is tuned to the QP problem
- LIBSVM `http://www.csie.ntu.edu.tw/~cjlin/libsvm/` - supports multi-class classification and regression
- SVM$^{\text{light}}$ `http://svmlight.joachims.org/` - good scalability

Implementations

- Many techniques known - Ellipsoid method, Gradient descent, Sequential Minimal Optimization - the latter is tuned to the QP problem
- LIBSVM `http://www.csie.ntu.edu.tw/~cjlin/libsvm/` - supports multi-class classification and regression
- SVM$^{\text{light}}$ `http://svmlight.joachims.org/` - good scalability
- General Convex Optimization Solvers - CVX, SeDuMi - compatible with Matlab©

Handwritten Digit Recognition

- Boser-Guyon-Vapnik First real-world application of SVMS

Handwritten Digit Recognition

- Boser-Guyon-Vapnik First real-world application of SVMS
- Two datasets USPS and NIST used for training and testing
- Performance stable even across a range of kernel choices

Handwritten Digit Recognition

- Boser-Guyon-Vapnik First real-world application of SVMS
- Two datasets USPS and NIST used for training and testing
- Performance stable even across a range of kernel choices
- Even simple polynomial kernels gave improvements (3.2% error) over MSE techniques like backpropagation or ridge-regression (12.7% error) on the USPS dataset.

Text Categorization

- Joachims - used insight from information retrieval research

Text Categorization

- Joachims - used insight from information retrieval research
- Reuters-21578 - a collection of labeled news stories used.

Text Categorization

- Joachims - used insight from information retrieval research
- Reuters-21578 - a collection of labeled news stories used.
- Native kernel used.

Text Categorization

- Joachims - used insight from information retrieval research
- Reuters-21578 - a collection of labeled news stories used.
- Native kernel used.
- Performance measured using Precision-Recall break-even point

Text Categorization

- Joachims - used insight from information retrieval research
- Reuters-21578 - a collection of labeled news stories used.
- Native kernel used.
- Performance measured using Precision-Recall break-even point
- Aggregate performance of Bayes, k-NN, C4.5 around or less than 80% whereas performance of even native kernel $> 84\%$

Text Categorization

- Joachims - used insight from information retrieval research
- Reuters-21578 - a collection of labeled news stories used.
- Native kernel used.
- Performance measured using Precision-Recall break-even point
- Aggregate performance of Bayes, k-NN, C4.5 around or less than 80% whereas performance of even native kernel $> 84\%$
- Use of RBF kernels improved performance to $> 86\%$

Image based Gender Identification

- Varma-Babu - application *Kernel Learning*

Image based Gender Identification

- Varma-Babu - application *Kernel Learning*
- Learn the kernel as a linear combination of base kernels

Image based Gender Identification

- Varma-Babu - application *Kernel Learning*
- Learn the kernel as a linear combination of base kernels
- Performance gains of 5-10% observed over other kernel based learning techniques

Topic Drift in Page-ranking Algorithms

- Karnick-Saradhi - application of multi-class Suport vector data description

Topic Drift in Page-ranking Algorithms

- Karnick-Saradhi - application of multi-class Suport vector data description
- Use SVDD to obtain representative pages for a topic and prune irrelevant pages

Topic Drift in Page-ranking Algorithms

- Karnick-Saradhi - application of multi-class Suport vector data description
- Use SVDD to obtain representative pages for a topic and prune irrelevant pages
- Use a kernel based on both link and content information
- Performance gains in terms of precision and recall observed over other existing topic distillation techniques

Bibliography

- An Introduction to Support Vector Machines, Nello Cristianini and John Shawe-Taylor, Cambridge University Press, 2000.

Bibliography

- An Introduction to Support Vector Machines, Nello Cristianini and John Shawe-Taylor, Cambridge University Press, 2000.
- Learning with Kernels, Bernhard Schlkopf and Alexander J. Smola, The MIT Press, 2002.

Bibliography

- An Introduction to Support Vector Machines, Nello Cristianini and John Shawe-Taylor, Cambridge University Press, 2000.
- Learning with Kernels, Bernhard Schlkopf and Alexander J. Smola, The MIT Press, 2002.
- `http://www.support-vector.net/`