Support Vector Machine

STAT5241 Section 2

Statistical Machine Learning

Xiaofei Shi

Tasks

Input $\longrightarrow$ Regressor $\longrightarrow$ Predict real number

Input $\longrightarrow$ Classifier $\longrightarrow$ Predict category

Input $\longrightarrow$ Density Estimator $\longrightarrow$ Probability

Types of classifiers

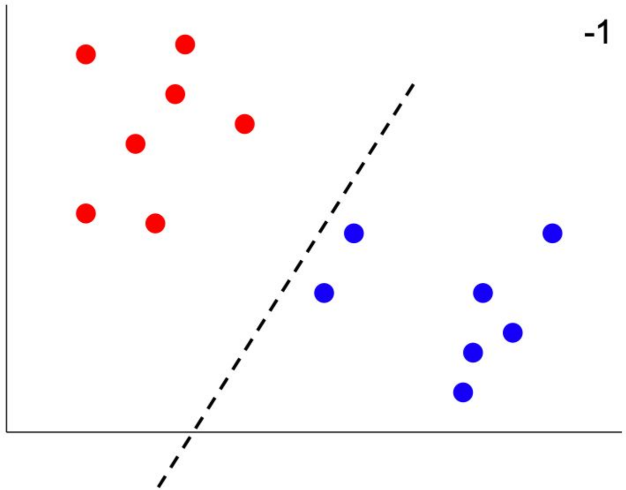
- Discriminative classifiers:
  - Directly estimate a decision rule/boundary
  - e.g. decision tree, SVM, logistic regression
- Instance based classifiers:
  - Use observation directly
  - e.g. K nearest neighborhood
- Generative classifiers:
  - Build a generative statistical model
  - e.g. Bayesian Network

Regression for classification

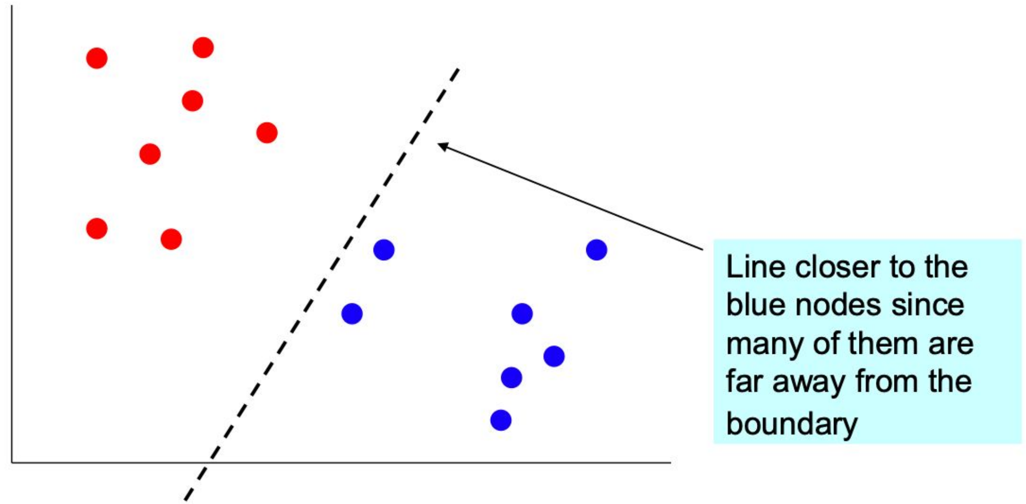
Recall our regression classifiers

+1 if $\text{sign}(w^T x+b) \geq 0$

-1 if $\text{sign}(w^T x+b) < 0$



Regression for classification

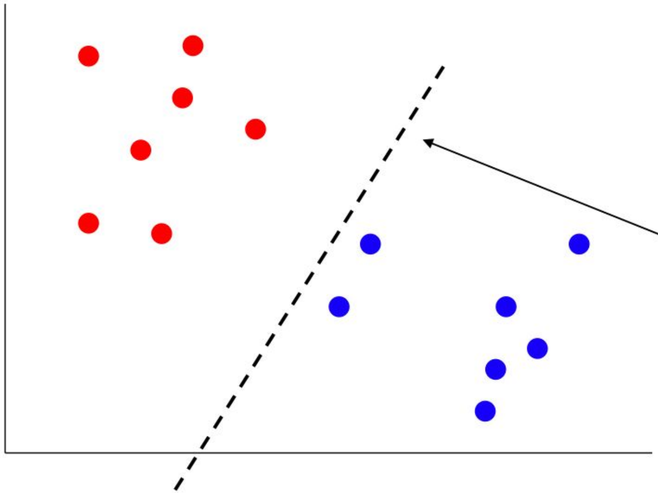


Regression classifier

Recall our regression classifiers

$$\min_w \sum_i (y_i - \mathrm{w}^{\mathrm{T}} x_i)^2$$

Goes over all points x (even for logistic regression)



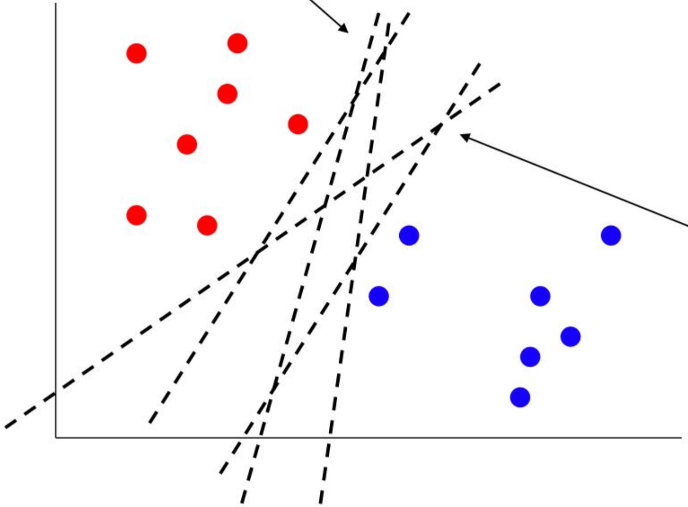
Line closer to the blue nodes since many of them are far away from the boundary

Regression classifier

Recall our regression classifiers

$$\min_w \sum_i (y_i - \mathrm{w}^T x_i)^2$$

Many more possible classifiers

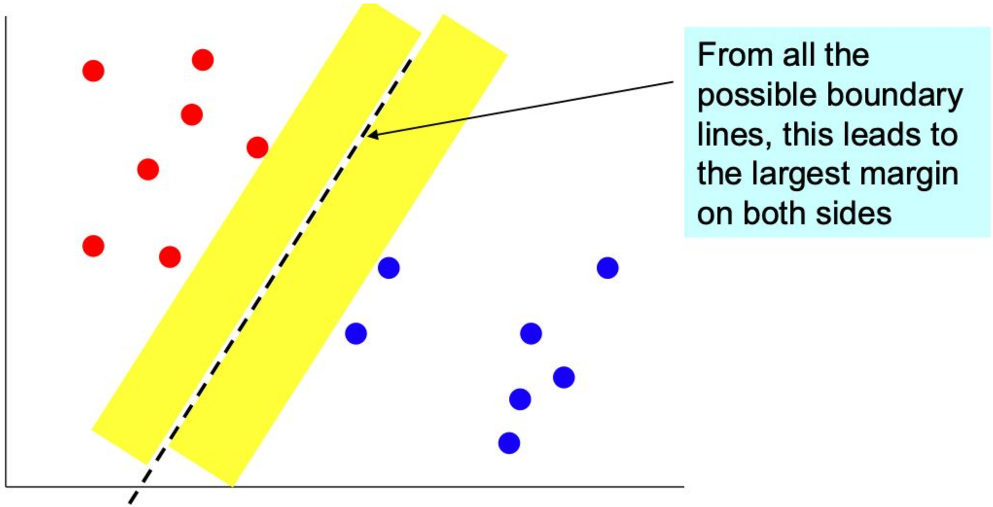


Goes over all points x (even in LR settings)

Line closer to the blue nodes since many of them are far away from the boundary

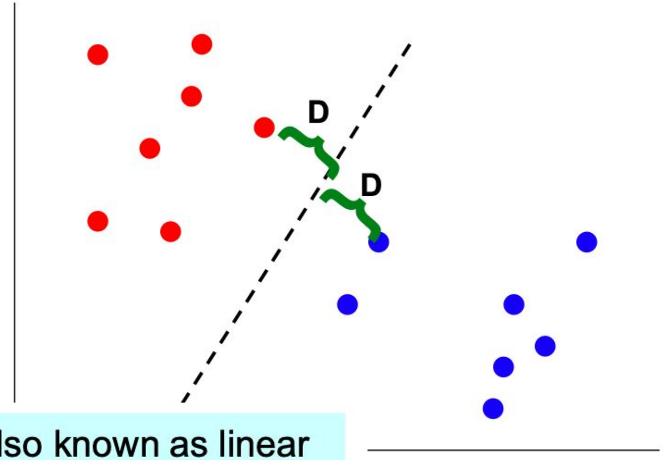
Columbia University

Maximum margin classifiers



- Instead of fitting all points, focus on boundary points
- Learn a boundary that leads to the largest margin from both sets of points (that is, largest distance to the closest point on either side)

Maximum margin classifiers



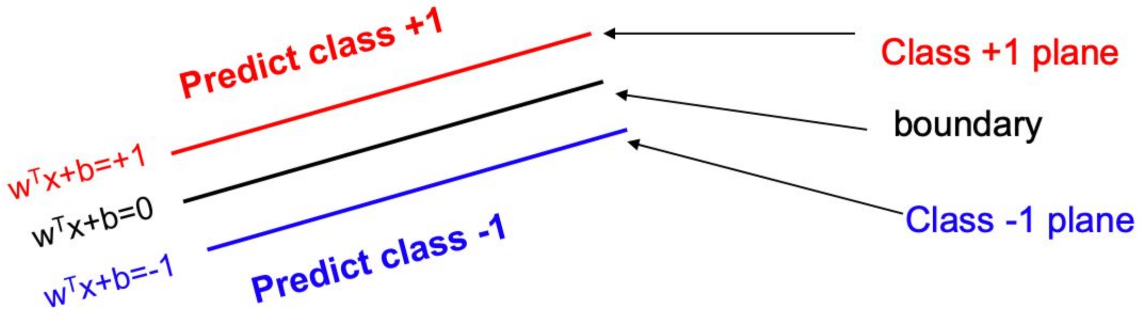
Why?

- Intuitive, 'makes sense'
- Some theoretical support
- Works well in practice

Also known as linear support vector machines (SVMs)

- Instead of fitting all points, focus on boundary points
- Learn a boundary that leads to the largest margin from both sets of points (that is, largest distance to the closest point on either side)

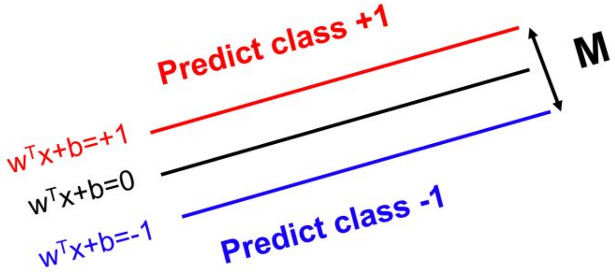
Classification rule of a max margin classifier



Classify as +1	if	$w^Tx+b \geq 1$
Classify as -1	if	$w^Tx+b \leq -1$
Undefined	if	$-1 < w^Tx+b < 1$

COLUMBIA UNIVERSITY

Classification rule of a max margin classifier



Classify as +1 if $w^Tx+b \geq 1$

Classify as -1 if $w^Tx+b \leq -1$

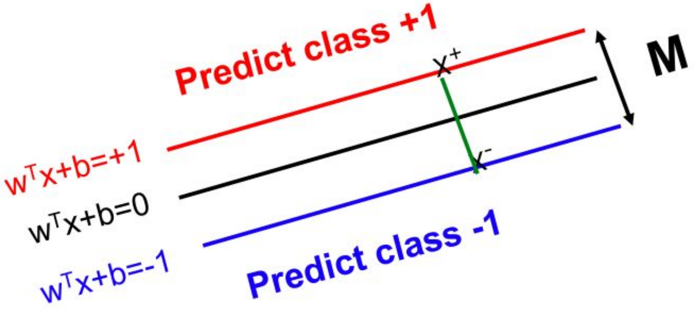
Undefined if $-1 < w^Tx+b < 1$

- Observation 1: the vector w is orthogonal to the +1 plane
- Why?

Let u and v be two points on the +1 plane, then for the vector defined by u and v we have $w^T(u-v) = 0$

Corollary: the vector w is orthogonal to the -1 plane

Classification rule of a max margin classifier



Classify as +1 if $w^Tx+b \geq 1$

Classify as -1 if $w^Tx+b \leq -1$

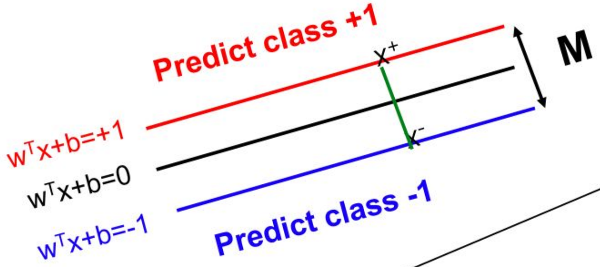
Undefined if $-1 < w^Tx+b < 1$

- Observation 1: the vector w is orthogonal to the +1 and -1 planes
- Observation 2: if x^+ is a point on the +1 plane and x^- is the *closest* point to x^+ on the -1 plane then

$$x^+ = \lambda w + x^-$$

Columbia University

How to choose a maximum margin



- $w^T x^+ + b = +1$
- $w^T x^- + b = -1$
- $x^+ = \lambda w + x^-$
- $| x^+ - x^- | = M$

We can now define M in terms of w and b

$w^T x^+ + b = +1$

$\Rightarrow$

$w^T (\lambda w + x^-) + b = +1$

$\Rightarrow$

$w^T x^- + b \;\; + \lambda w^T w = +1$

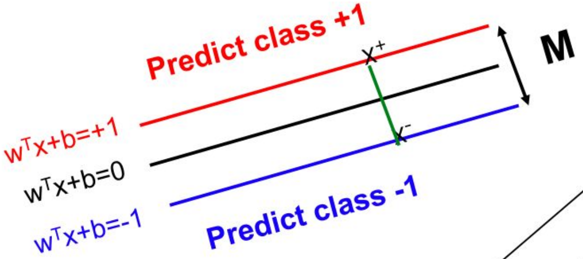
$\Rightarrow$

$-1 \;\; + \lambda w^T w = +1$

$\Rightarrow$

$\lambda = 2/w^T w$

Putting it together



- $w^T x^+ + b = +1$
- $w^T x^- + b = -1$
- $x^+ = \lambda w + x^-$
- $| x^+ - x^- | = M$
- $\lambda = 2/w^T w$

We can now define M in terms of w and b

$$M = |x^+ - x^-|$$

$$\Rightarrow$$

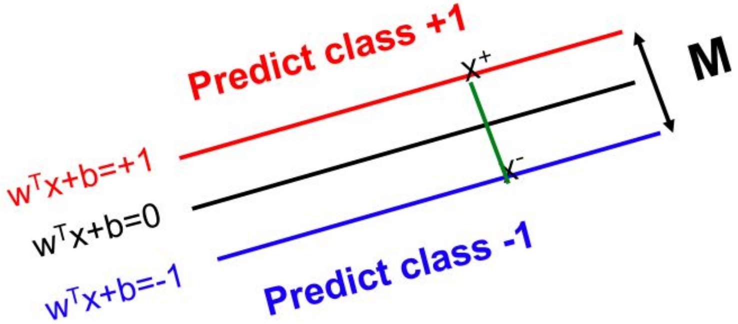
$$M = | \lambda w | = \lambda | w | = \lambda \sqrt{w^T w}$$

$$\Rightarrow$$

$$M = 2 \frac{\sqrt{w^T w}}{w^T w} = \frac{2}{\sqrt{w^T w}}$$

Columbia University

The optimal margin



$$M = \frac{2}{\sqrt{w^T w}}$$

We can now search for the optimal parameters by finding a solution that:

1. Correctly classifies all points
2. Maximizes the margin (or equivalently minimizes w^Tw)

A special convex optimization problem: Quadratic Programming (QP)

A special convex optimization problem: Quadratic Programming (QP)

$$\min_U \frac{u^T R u}{2} + d^T u + c$$

u -vector (unknown)
R – squared matrix
d – vector
c - scalar

Quadratic term

subject to n inequality constraints:

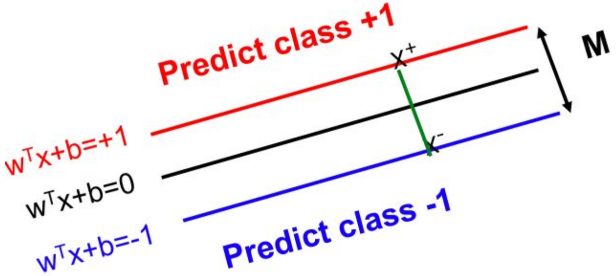
$$a_{11}u_1 + a_{12}u_2 + \ldots \leq b_1$$
$$\vdots \qquad \vdots \qquad \vdots$$
$$a_{n1}u_1 + a_{n2}u_2 + \ldots \leq b_n$$

and k equivalency constraints:

$$a_{n+1,1}u_1 + a_{n+1,2}u_2 + \ldots = b_{n+1}$$
$$\vdots \qquad \vdots \qquad \vdots$$
$$a_{n+k,1}u_1 + a_{n+k,2}u_2 + \ldots = b_{n+k}$$

When a problem can be specified as a QP problem we can use solvers that are better than gradient descent or simulated annealing

SVM as QP



$$M = \frac{2}{\sqrt{w^T w}}$$

Min $(w^Tw)/2$

subject to the following inequality constraints:

For all x in class + 1

$w^Tx+b \geq 1$

For all x in class - 1

$w^Tx+b \leq -1$

A total of n constraints if we have n input samples

$$\min_U \frac{u^T R u}{2} + d^T u + c$$

subject to n inequality constraints:

$$\begin{aligned} &a_{11}u_1 + a_{12}u_2 + \ldots \leq b_1 \\ &\vdots \qquad \vdots \qquad \vdots \\ &a_{n1}u_1 + a_{n2}u_2 + \ldots \leq b_n \end{aligned}$$

and k equivalency constraints:

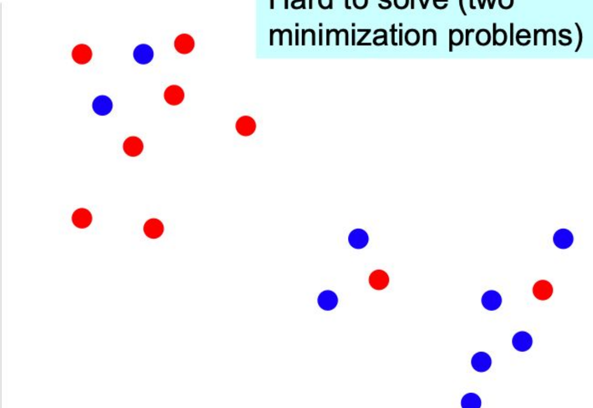
$$\begin{aligned} &a_{n+1,1}u_1 + a_{n+1,2}u_2 + \ldots = b_{n+1} \\ &\vdots \qquad \vdots \qquad \vdots \\ &a_{n+k,1}u_1 + a_{n+k,2}u_2 + \ldots = b_{n+k} \end{aligned}$$

What if……

- So far we assumed that a linear plane can perfectly separate the points
- But this is not usally the case

 - noise, outliers

Hard to solve (two minimization problems)



How can we convert this to a QP problem?

- Minimize training errors?

  min $w^T w$

  min #errors

- Penalize training errors:

 min $w^T w$+C*(#errors)

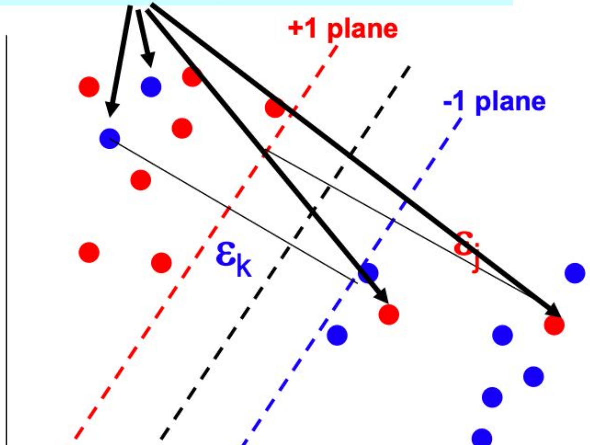
Hard to encode in a QP problem

COLUMBIA UNIVERSITY

Not linearly separable case

- Instead of minimizing the number of misclassified points we can minimize the *distance* between these points and their correct plane

These are also support vectors since they impact the parameters of the decision boundary



The new optimization problem is:

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\varepsilon_i$$

subject to the following inequality constraints:

For all x_i in class + 1

$w^T x + b \geq 1 - \varepsilon_i$

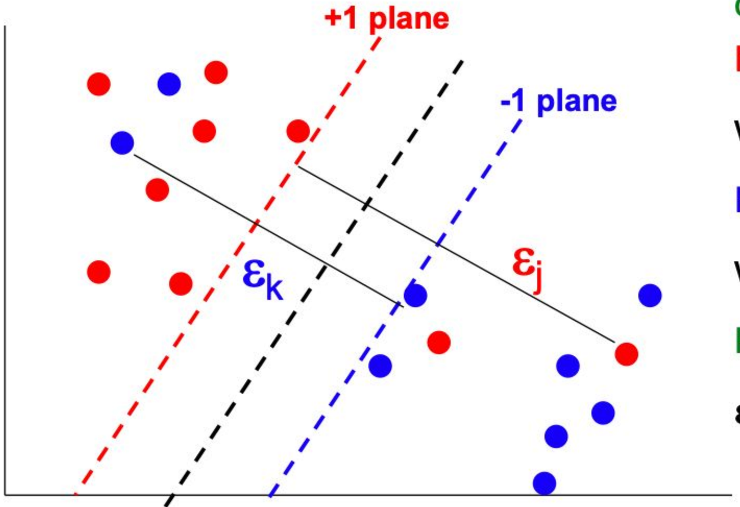
For all x_i in class - 1

$w^T x + b \leq -1 + \varepsilon_i$

Wait. Are we missing something?

COLUMBIA UNIVERSITY

Soft-threshold



The new optimization problem is:

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\varepsilon_i$$

subject to the following inequality constraints:

For all x_i in class + 1

$w^T x + b \geq 1 - \varepsilon_i$

For all x_i in class - 1

$w^T x + b \leq -1 + \varepsilon_i$

A total of n constraints

For all i

$\varepsilon_I \geq 0$

Another n constraints

Last time

Two optimization problems: For the separable and non separable cases

Min $(w^Tw)/2$

For all x in class + 1

$w^Tx+b \geq 1$

For all x in class - 1

$w^Tx+b \leq -1$

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\varepsilon_i$$

For all x_i in class + 1

$w^Tx+b \geq 1- \varepsilon_i$

For all x_i in class - 1

$w^Tx+b \leq -1+ \varepsilon_i$

For all i

$\varepsilon_i \geq 0$

Last time

Two optimization problems: For the separable and non separable cases

Min $(w^T w)/2$

For all x in class + 1

$w^T x + b \geq 1$

For all x in class - 1

$w^T x + b \leq -1$

$$\min_w \frac{w^T w}{2} + \sum_{i=1}^{n} C\varepsilon_i$$

For all x_i in class + 1

$w^T x + b \geq 1 - \varepsilon_i$

For all x_i in class - 1

$w^T x + b \leq -1 + \varepsilon_i$

For all i

$\varepsilon_i \geq 0$

- Instead of solving these QPs directly we will solve a dual formulation of the SVM optimization problem
- The main reason for switching to this type of representation is that it would allow us to use a neat trick that will make our lives easier (and the run time faster)

Columbia University

An alternative (dual) representation

Min $(w^T w)/2$

For all x in class +1

$w^T x + b \geq 1$

For all x in class -1

$w^T x + b \leq -1$

Why? $\Downarrow$

Min $(w^T w)/2$

$(w^T x_i + b) y_i \geq 1$

- We will start with the linearly separable case
- Instead of encoding the correct classification rule and constraint we will use Lagrange multipliers to encode it as part of the our minimization problem

COLUMBIA UNIVERSITY

An alternative (dual) representation

Min $(w^Tw)/2$

$(w^Tx_i+b)y_i \geq 1$

- We will start with the linearly separable case
- Instead of encoding the correct classification rule and constraint we will use Lagrange multipliers to encode it as part of the our minimization problem

Recall that Lagrange multipliers can be applied to turn the following problem:

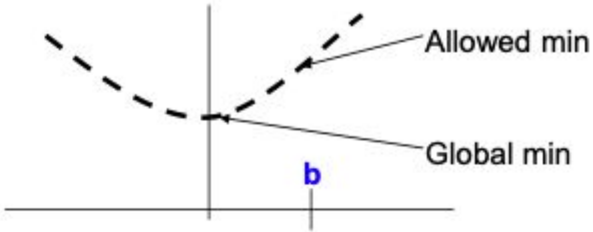
$\min_x x^2$

s.t. $x \geq b$

To

$\min_x \max_\alpha x^2 - \alpha(x-b)$

s.t. $\alpha \geq 0$



COLUMBIA UNIVERSITY

An alternative (dual) representation

Dual formulation

$$\min_{w,b} \max_{\alpha} \frac{\mathrm{w}^T\mathrm{w}}{2} - \sum_i \alpha_i[(\mathrm{w}^T x_i + b)y_i - 1]$$

$$\alpha_i \geq 0 \qquad \forall i$$

Using this new formulation we can derive w and b by taking the derivative w.r.t. w and α leading to:

$$w = \sum_i \alpha_i x_i y_i$$

$$b = y_i - \mathrm{w}^T x_i$$

$$for \quad i \quad s.t. \quad \alpha_i > 0$$

Finally, taking the derivative w.r.t. b we get:

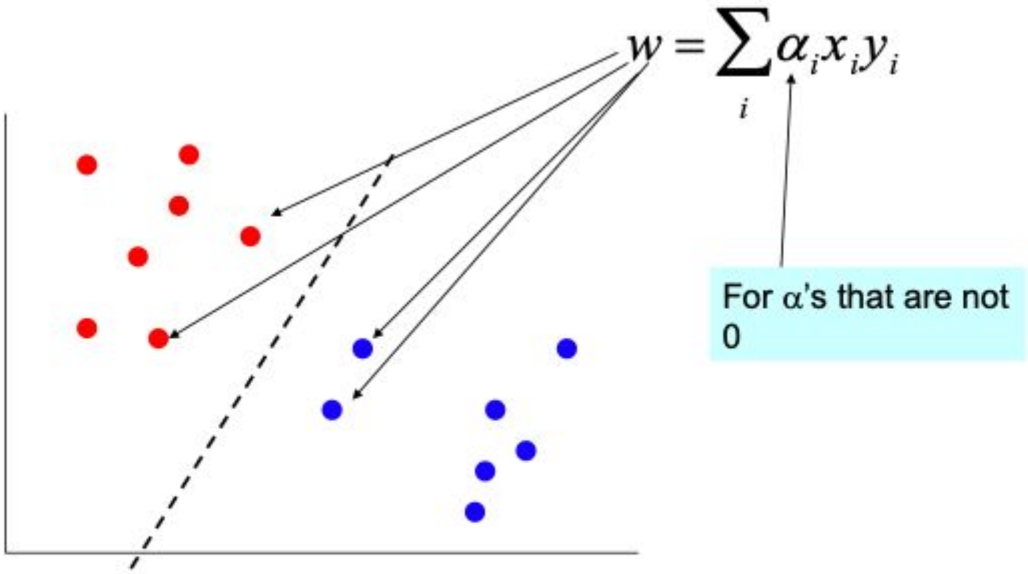
$$\sum_i \alpha_i y_i = 0$$

Original formulation

Min $(w^T w)/2$

$(w^T x_i + b) y_i \geq 1$

Dual SVM: interpretation



Dual SVM: linearly separable case

Substituting w into our target function and using the additional constraint we get:

Dual formulation

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2}\sum_{\mathrm{i,j}} \alpha_i \alpha_j \mathrm{y_i} y_j \mathbf{x_i^T x_j}$$

$$\sum_i \alpha_i \mathrm{y_i} = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

$$\min_{w,b} \max_{\alpha} \frac{\mathrm{w^T w}}{2} - \sum_i \alpha_i [(\mathrm{w^T} x_i + b) y_i - 1]$$

$$\alpha_i \geq 0 \qquad \forall i$$

$$w = \sum_i \alpha_i x_i y_i$$

$$b = y_i - \mathrm{w^T} x_i$$

$$for \quad i \quad s.t. \quad \alpha_i > 0$$

$$\sum_i \alpha_i y_i = 0$$

COLUMBIA UNIVERSITY

Dual SVM: linearly separable case

Our dual target function:

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{\mathrm{i,j}} \alpha_i \alpha_j \mathrm{y_i} y_j \mathbf{x_i^T} \mathbf{x_j}$$

$$\sum_i \alpha_i \mathrm{y_i} = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

Dot product for all training samples

Dot product with training samples

To evaluate a new sample x_k we need to compute:

$$\mathrm{w^T} x_j + b = \sum_{\mathrm{i}} \alpha_i \mathrm{y_i} \mathbf{x_i^T} \mathbf{x_k} + b$$

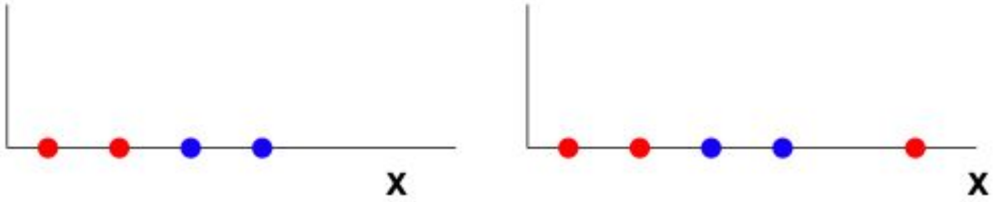
Is this too much computational work (for example when using transformation of the data)?

COLUMBIA UNIVERSITY

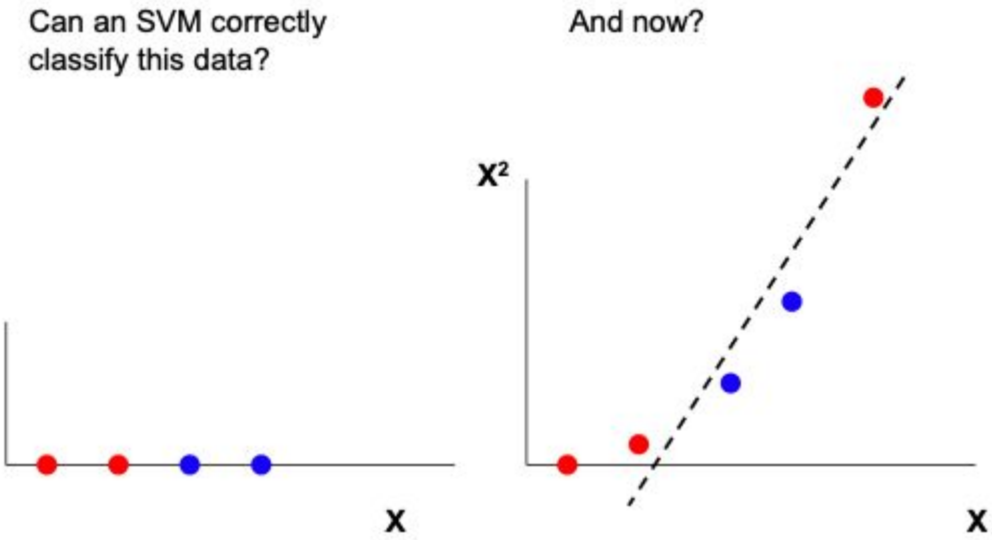
Classifying in 1-D

Can an SVM correctly classify this data?

What about this?

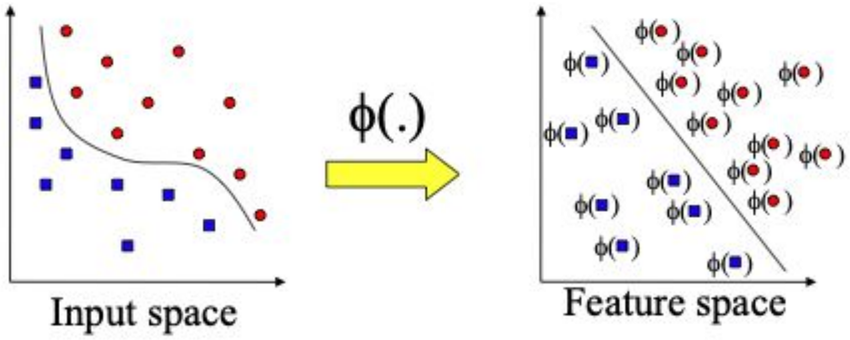


Classifying in 1-D



Nonlinear SVMs: 2-D

- The original input space (x) can be mapped to some higher-dimensional feature space (φ(x))where the training set is separable
- If data is mapped into sufficiently high dimension, then samples will in general be linearly separable
- N data points are in general separable in a space of N-1 dimensions or more!!



Columbia University

Transformation of inputs

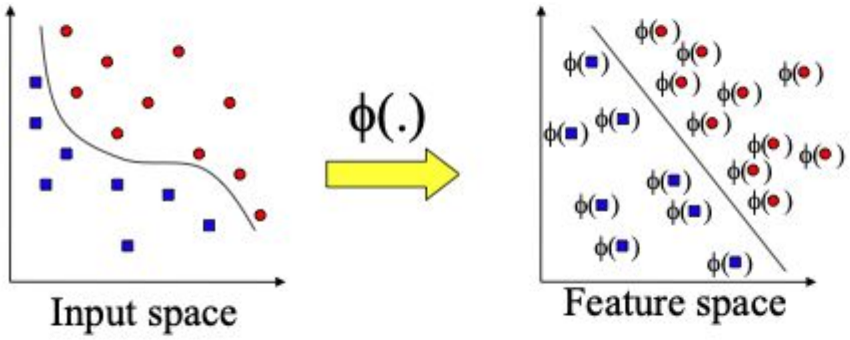
- Possible problems

  - High computation burden due to high-dim

  - Many more parameters
- SVM solves these two issues simultaneously

  - "Kernel tricks" for efficient computation

  - Dual formulation only assigns parameters to samples, not features



Columbia University

- While working in higher dimensions is beneficial, it also increases our run time because of the dot product computation
- However, there is a neat trick we can use
- consider all quadratic terms for $x^1, x^2 \dots x^m$

$$\max_\alpha \sum_i \alpha_i - \sum_{i,j} \alpha_i \alpha_j y_i y_j \Phi(\mathbf{x}_i)\Phi(\mathbf{x}_j)$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

m is the number of features in each vector

The √2 term will become clear in the next slide

$$\Phi(x) = \begin{pmatrix} 1 \\ \sqrt{2}x^1 \\ \vdots \\ \sqrt{2}x^m \\ (x^1)^2 \\ \vdots \\ (x^m)^2 \\ \sqrt{2}x^1x^2 \\ \vdots \\ \sqrt{2}x^{m-1}x^m \end{pmatrix}$$

m+1 linear terms

m quadratic terms

m(m-1)/2 pairwise terms

COLUMBIA UNIVERSITY

Quadratic kernels

$$\Phi(x)\Phi(z) = \begin{bmatrix} 1 \\ \sqrt{2}x^1 \\ \vdots \\ \sqrt{2}x^1 \\ (x^1)^2 \\ \vdots \\ (x^m)^2 \\ \sqrt{2}x^1x^2 \\ \vdots \\ \sqrt{2}x^{m-1}x^m \end{bmatrix} \bullet \begin{bmatrix} 1 \\ \sqrt{2}z^1 \\ \vdots \\ \sqrt{2}z^2 \\ (z^1)^2 \\ \vdots \\ (x^m)^2 \\ \sqrt{2}z^1z^2 \\ \vdots \\ \sqrt{2}z^{m-1}z^m \end{bmatrix} = \underbrace{\sum_i 2x^i z^i}_{m} + \underbrace{\sum_i (x^i)^2 (z^i)^2}_{m} + \underbrace{\sum_i \sum_{j=i+1} 2x^i x^j z^i z^j}_{m(m-1)/2} + 1$$

=~ m²

To summarize...

Our dual target function:

$$\max_\alpha \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\sum_i \alpha_i y_i = 0$$

$$\alpha_i \geq 0 \qquad \forall i$$

*mn*² operations at each iteration

To evaluate a new sample x_j we need to compute:

$$w^T x_j + b = \sum_i \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b$$

mr operations where *r* are the number of support vectors (α_i>0)

For other kernels

- Same kernel tricks work for higher order polynomial kernels as well
- For example, a polynomial of degree 4 can be computed using (<x,z>+1)^4, and for polynomial of degree p can be computed using (<x,z>+1)^p
- Beyond polynomials, we can use other kernel functions (basis functions)

  - Radial basis style kernel function (Gaussian kernel): $$K(x,z) = \exp\left(-\frac{(x-z)^2}{2\sigma^2}\right)$$

  - Neural net style kernel function: $$K(x,z) = \tanh(\kappa x.z - \delta)$$

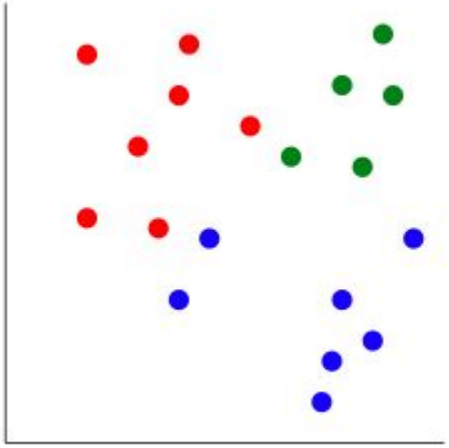
Columbia University

Why do SVM work

If we are using huge features spaces (with kernels), how come we are not overfitting the data?

- Number of parameters remains the same (and most are set to 0)
- While we have a lot of input values, at the end we only care about the support vectors and these are usually a small group of samples
- The minimization (or the maximizing of the margin) function acts as a sort of regularization term leading to reduced overfitting

Multiclass classification tasks with SVMs



• Most common solution: One vs. all

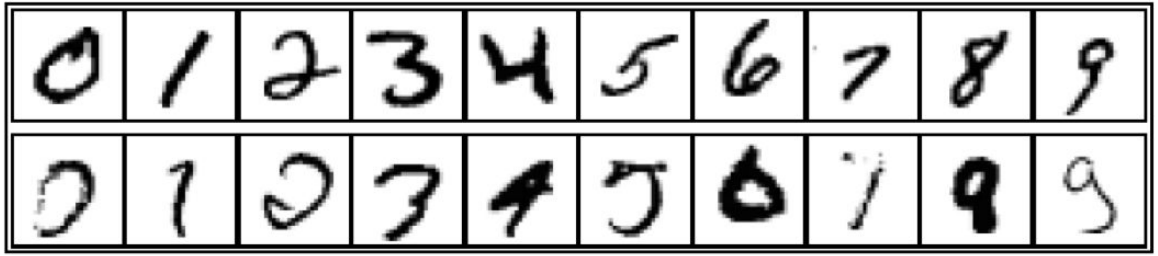
- create a classifier for each class against all other data

- for a new point use all classifiers and compare the margin for all selected classes

Note that this is not necessarily valid since this is not what we trained the SVM for, but often works well in practice

Columbia University

Handwritten recognition (MNIST)



3-nearest-neighbor = 2.4% error
400–300–10 unit MLP = 1.6% error
LeNet: 768–192–30–10 unit MLP = 0.9% error

Current best (kernel machines, vision algorithms) $\approx$ 0.6% error

COLUMBIA UNIVERSITY

Takeaways

- Difference between regression classifiers and SVMs
- Maximum margin principle
- Target function for SVMs and the dual formulation
- Linearly separable and non-separable cases
- Kernel tricks and computational complexity

References

- Christopher Bishop: Pattern Recognition and Machine Learning, Chapter 6 & 7
- Ziv Bar-Joseph, Pradeep Ravikumar and Aarti Singh: CMU 10-701