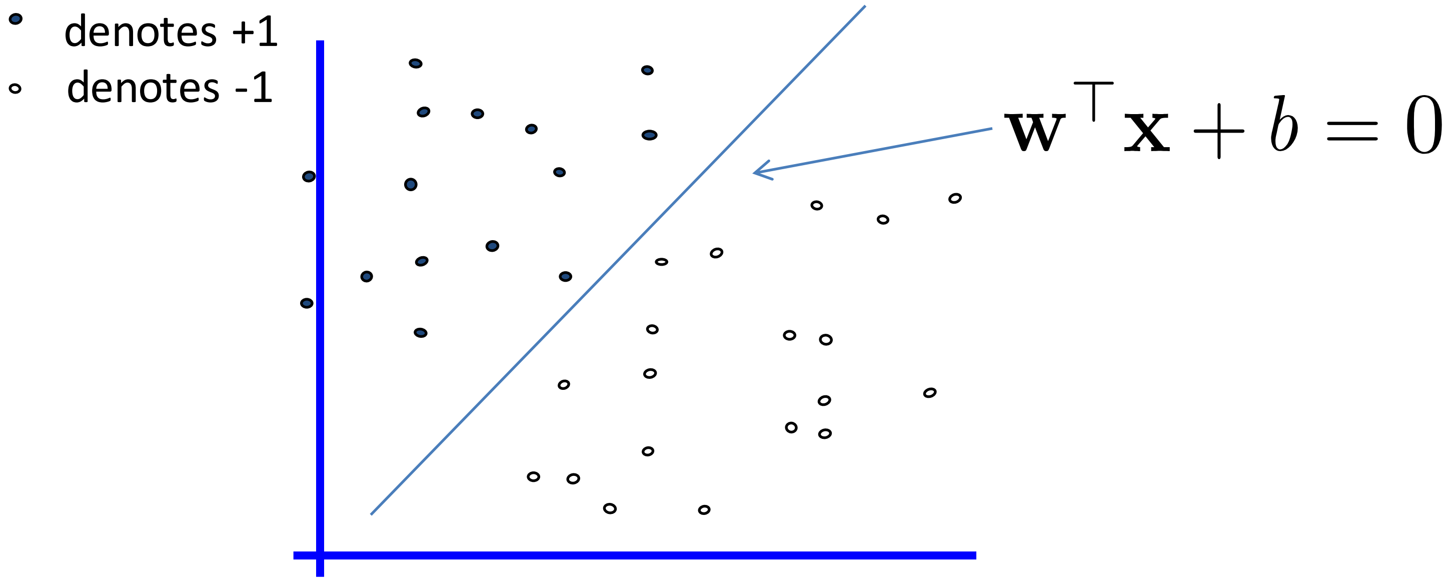
Support Vector Machine

Jiayu Zhou

Linear Classifiers

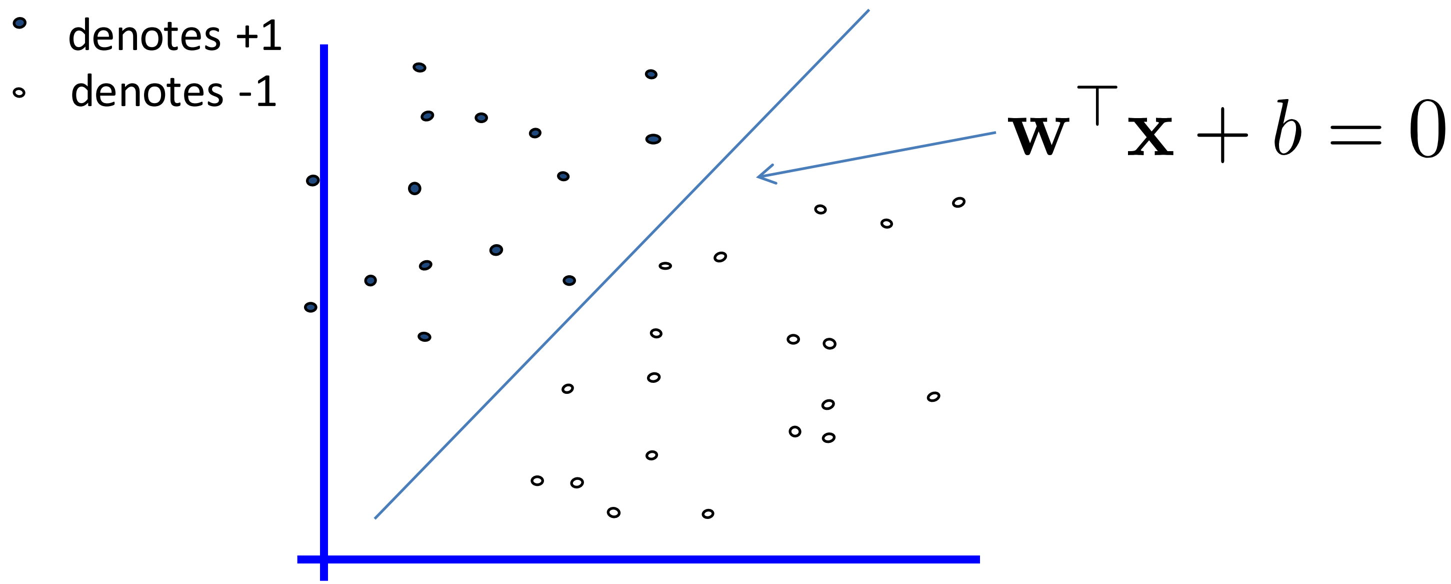
$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$$



- How to find the linear decision boundary that linearly separates data points from two classes?

Linear Classifiers

$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$$

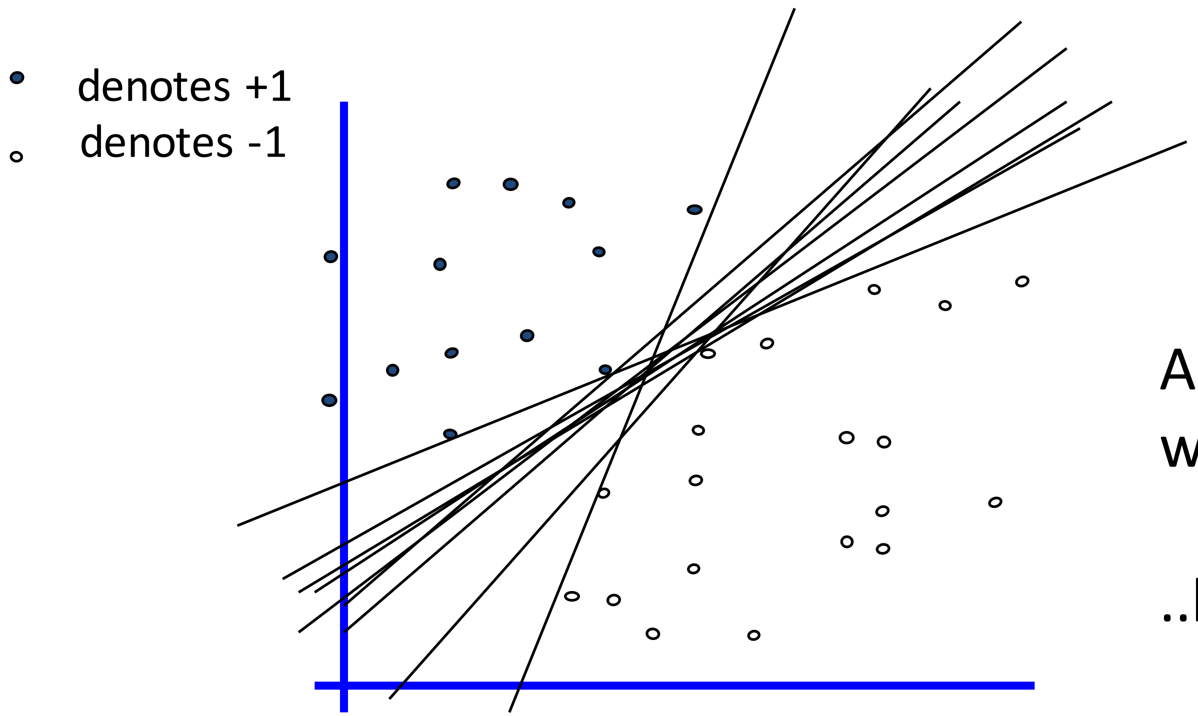


$$\max_{\mathbf{w},b} \quad 0$$

$$\text{s. t.} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) > 0, i = 1, \ldots, N$$

Linear Classifiers

$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$$

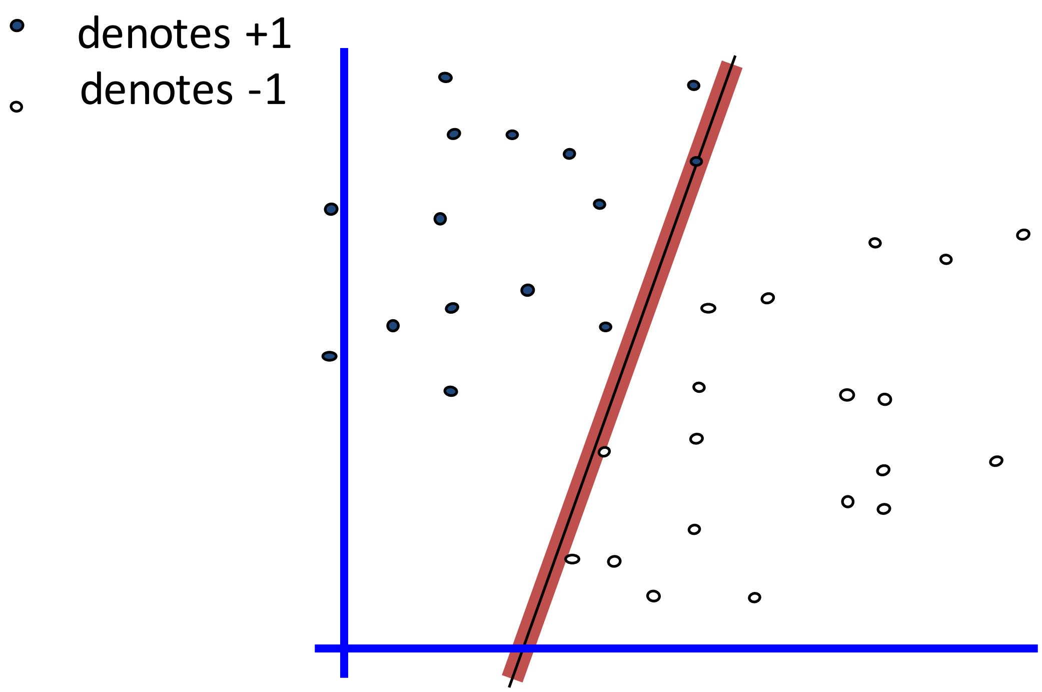


Any of these would be fine..

..but which is best?

Classifier Margin

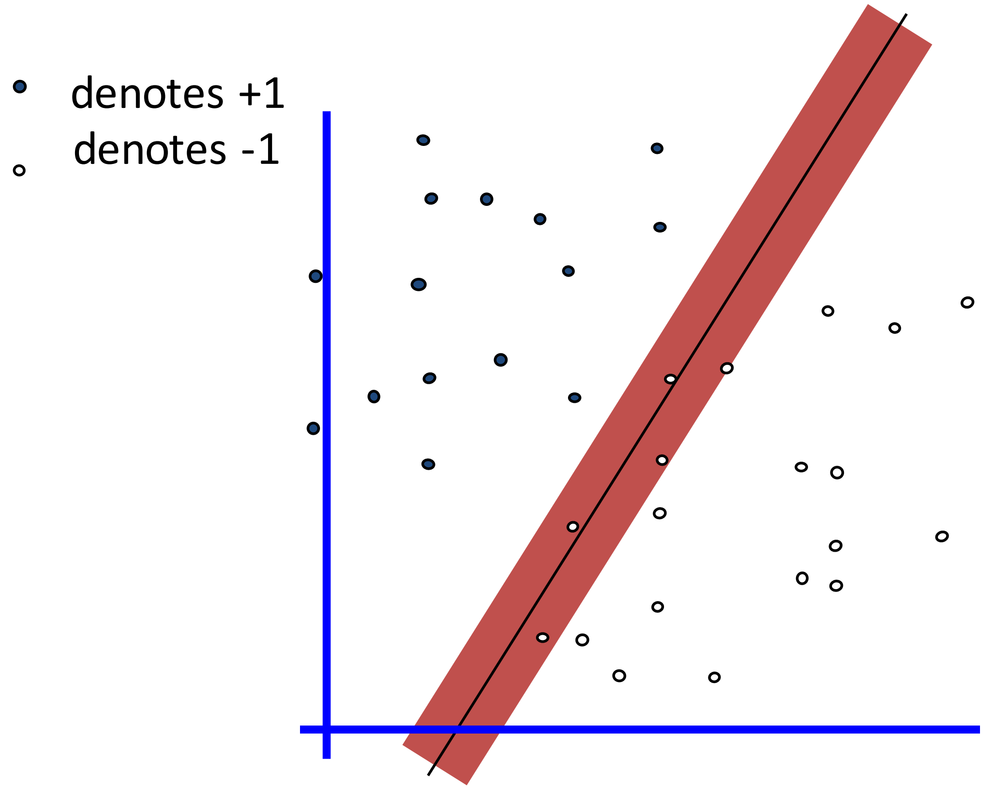
$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$$



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

Maximum Margin

$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$$

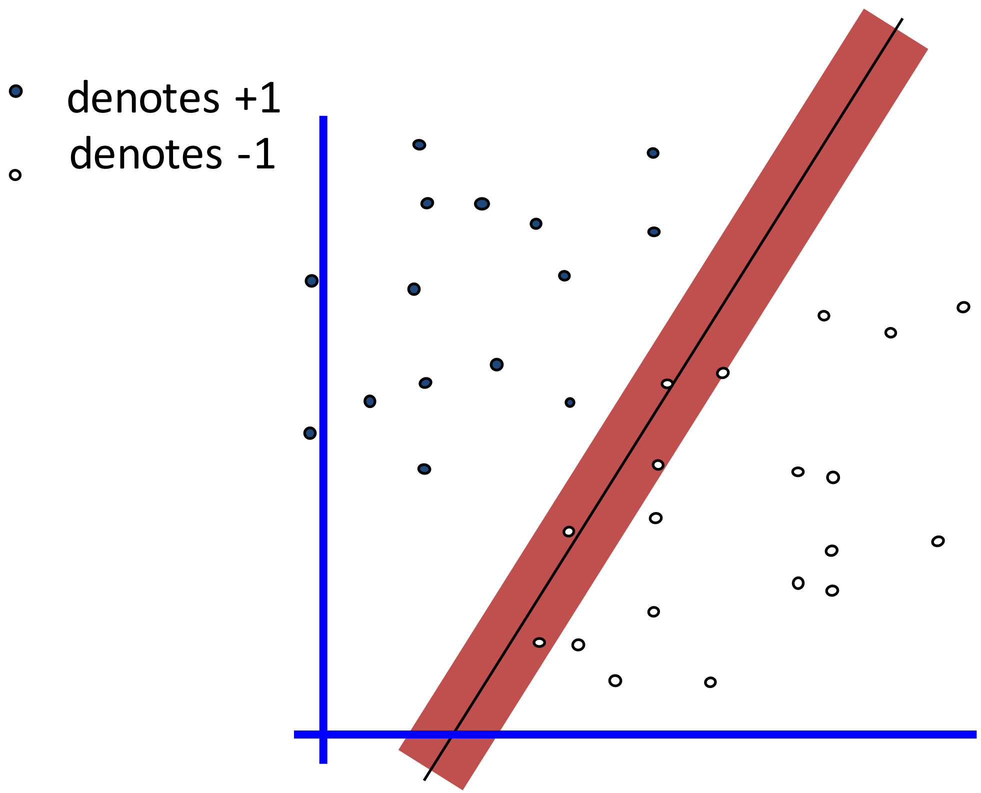


The maximum margin linear classifier is the linear classifier with the maximum margin.

This is the simplest kind of SVM (called an Linear SVM)

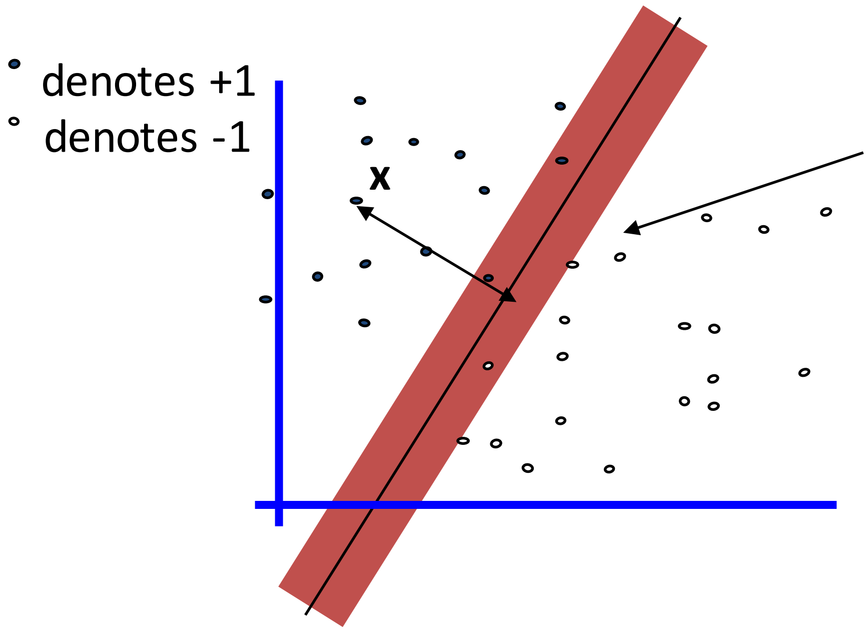
Why Maximum Margin ?

$$f(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}^\top \mathbf{x} + b)$$



1. 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.
2. There's some theory (using VC dimension) that is related to (but not the same as) the proposition that this is a good thing.
3. Empirically it works very very well.

Estimate the Margin

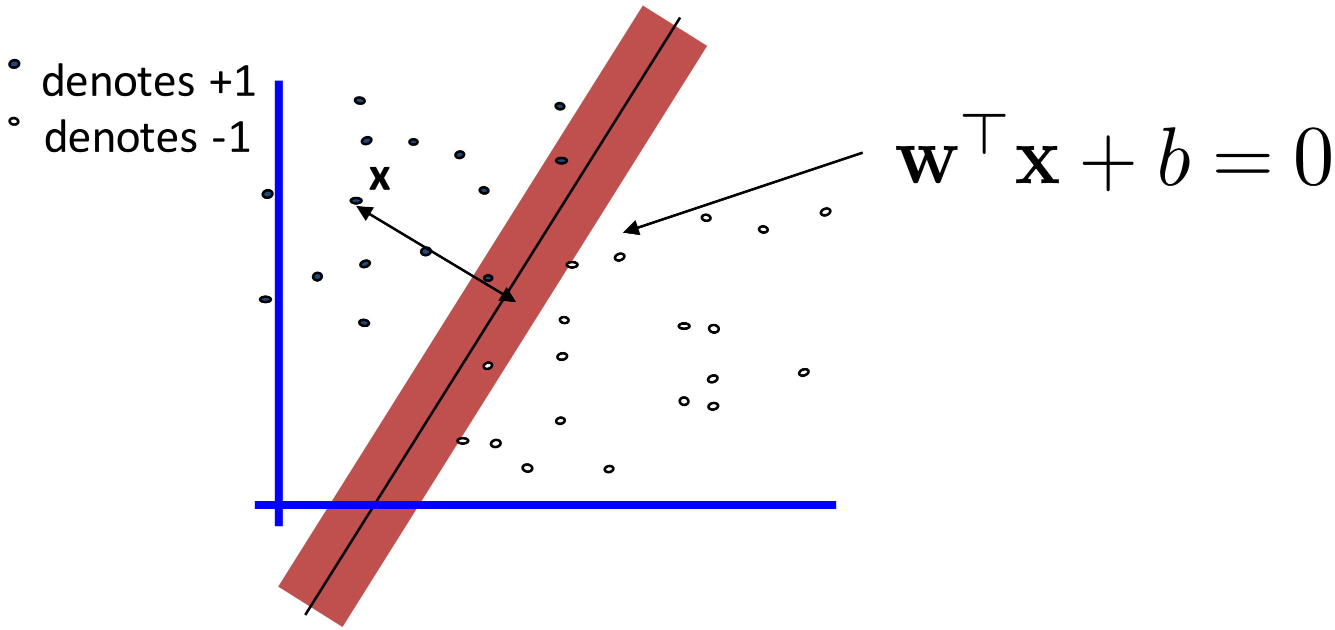


$$\mathbf{w}^\top \mathbf{x} + b = 0$$

- What is the distance expression for a point **x** to a line **wx**+b= 0?

$$d(\mathbf{x}, \mathbf{w}, b) = \frac{|\mathbf{x}^\top \mathbf{w} + b|}{|\mathbf{w}|_2} = \frac{|\mathbf{x}^\top \mathbf{w} + b|}{\sqrt{\sum_{j=1}^{d} w_j^2}}$$

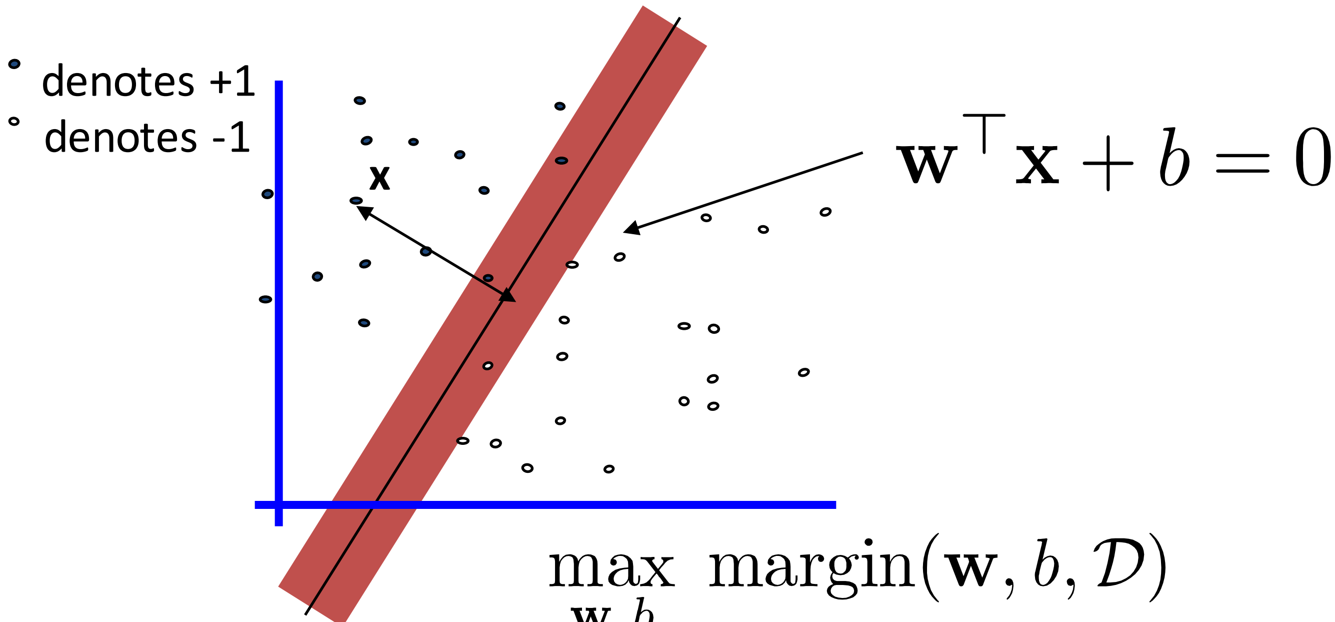
Estimate the Margin



- What is the classification margin for **wx**+b= 0?

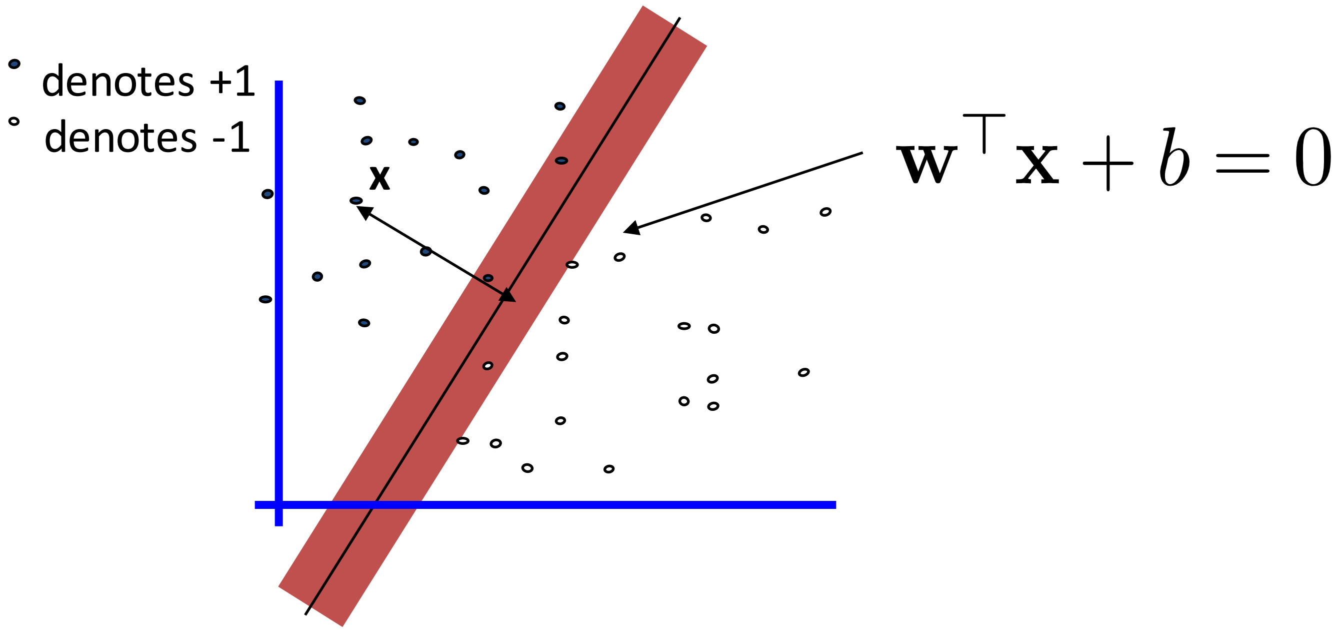
$$\text{margin}(\mathbf{w}, b, \mathcal{D}) = \min_{\mathbf{x}_i \in \mathcal{D}} d(\mathbf{x}_i, \mathbf{w}, b)$$

Maximize the Classification Margin



$$
\begin{aligned}
& \max_{\mathbf{w}, b} \ \text{margin}(\mathbf{w}, b, \mathcal{D}) \\
= \quad & \max_{\mathbf{w}, b} \min_{\mathbf{x}_i \in \mathcal{D}} d(\mathbf{x}_i, \mathbf{w}, b) \\
= \quad & \max_{\mathbf{w}, b} \min_{\mathbf{x}_i \in \mathcal{D}} \frac{|\mathbf{w}^\top \mathbf{x}_i + b|}{|\mathbf{w}|_2}
\end{aligned}
$$

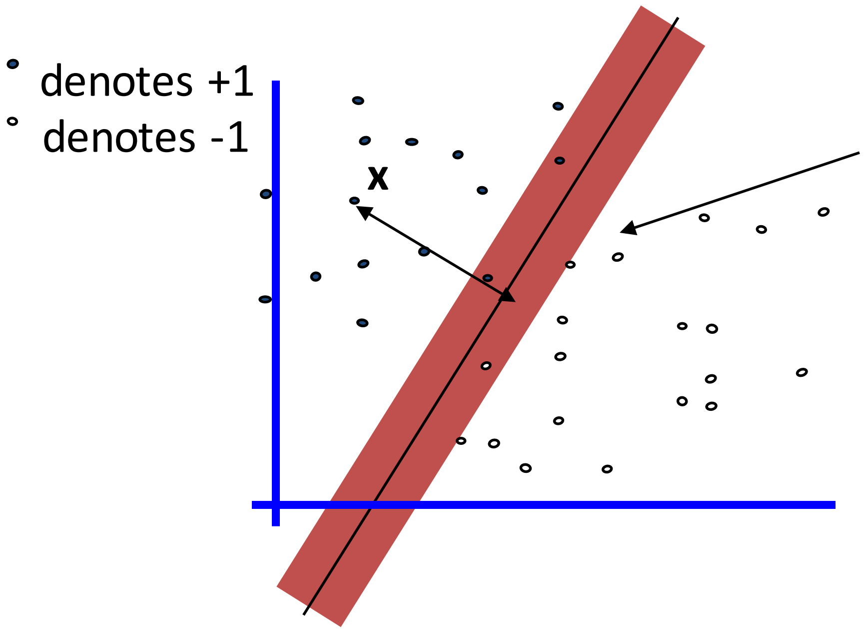
Maximum Margin



$$\max_{\mathbf{w}, b} \quad \min_{\mathbf{x}_i \in \mathcal{D}} \frac{|\mathbf{w}^\top \mathbf{x}_i + b|}{\|\mathbf{w}\|_2}$$

$$\text{s. t.} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) > 0, i = 1, \ldots, N$$

Maximum Margin



$$\mathbf{w}^\top \mathbf{x} + b = 0$$

$$\max_{\mathbf{w},b} \quad \min_{\mathbf{x}_i \in \mathcal{D}} \frac{|\mathbf{w}^\top \mathbf{x}_i + b|}{|\mathbf{w}|_2}$$

$$\text{s. t.} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) > 0, i = 1, \ldots, N$$

$$\min_{\mathbf{x}_i \in \mathcal{D}} |\mathbf{w}^\top \mathbf{x}_i + b| = 1$$

$$\min_{\mathbf{w},b} \quad |\mathbf{w}|_2^2 = \sum_{j=1}^{d} w_j^2$$

$$\text{s. t.} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, i = 1, \ldots, N$$

Maximum Margin

$$\min_{\mathbf{w},b} \quad |\mathbf{w}|_2^2 = \sum_{j=1}^{d} w_j^2$$

$$\text{s. t.} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1, i = 1, \ldots, N$$

Quadratic programming problem

- Quadratic objective function
- Linear equality and inequality constraints
- Well studied problem in OR

Quadratic Programming

Find $\arg\min_{\mathbf{u}} \; c + \mathbf{d}^T\mathbf{u} + \dfrac{\mathbf{u}^T R\mathbf{u}}{2}$ Quadratic criterion

Subject to

$$a_{11}u_1 + a_{12}u_2 + \ldots + a_{1m}u_m \le b_1$$
$$a_{21}u_1 + a_{22}u_2 + \ldots + a_{2m}u_m \le b_2$$
$$\vdots$$
$$a_{n1}u_1 + a_{n2}u_2 + \ldots + a_{nm}u_m \le b_n$$

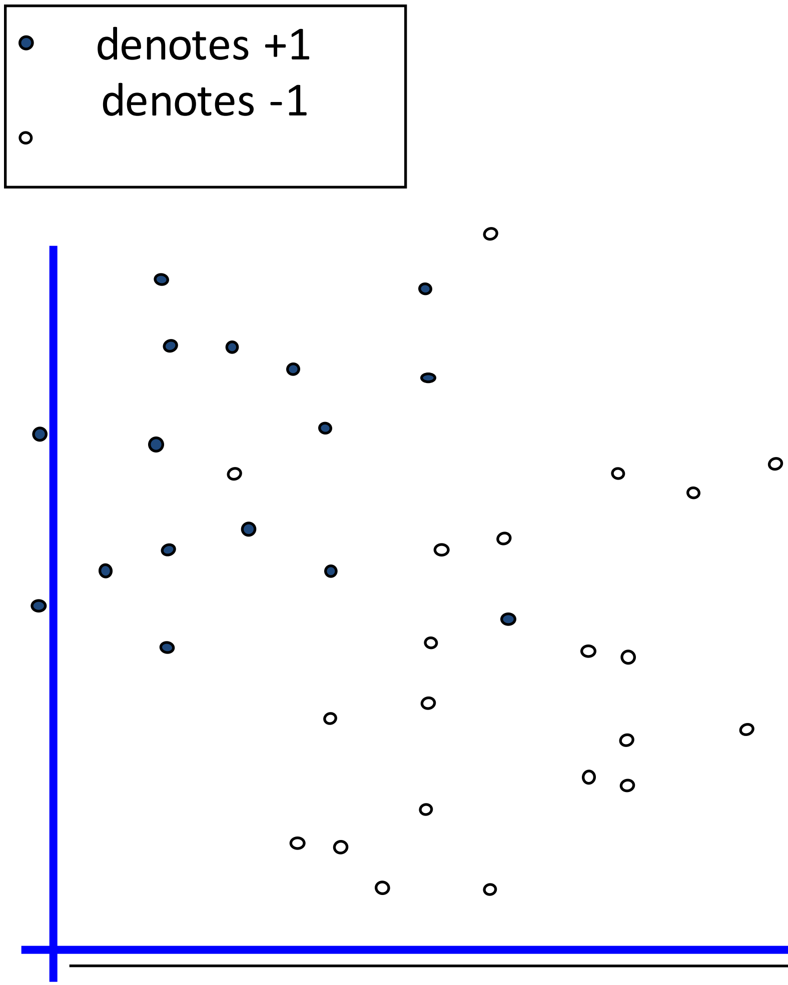
n additional linear inequality constraints

And subject to

$$a_{(n+1)1}u_1 + a_{(n+1)2}u_2 + \ldots + a_{(n+1)m}u_m = b_{(n+1)}$$
$$a_{(n+2)1}u_1 + a_{(n+2)2}u_2 + \ldots + a_{(n+2)m}u_m = b_{(n+2)}$$
$$\vdots$$
$$a_{(n+e)1}u_1 + a_{(n+e)2}u_2 + \ldots + a_{(n+e)m}u_m = b_{(n+e)}$$

e additional linear equality constraints

Linearly Inseparable Case



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

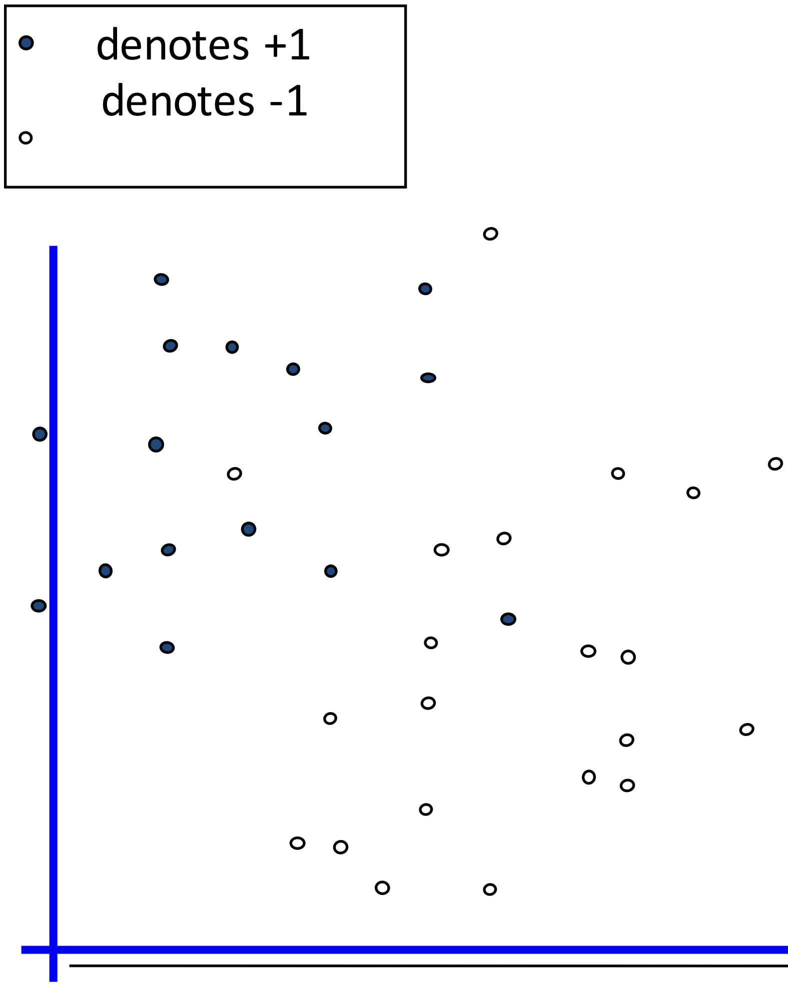
$$\min_{\mathbf{w},b} \quad |\mathbf{w}|_2^2 = \sum_{j=1}^{d} w_j^2$$

$$\text{s. t.} \quad y_1(\mathbf{w}^\top \mathbf{x}_1 + b) \geq 1$$

$$\cdots$$

$$y_N(\mathbf{w}^\top \mathbf{x}_N + b) \geq 1$$

Linearly Inseparable Case



- Relax the constraints
- Penalize the relaxation

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\sum_{j=1}^{d} w_j^2 + C\sum_{i=1}^{N}\varepsilon_i$$

$$\text{s. t.} \quad y_1(\mathbf{w}^\top \mathbf{x}_1 + b) \geq 1 - \varepsilon_1$$

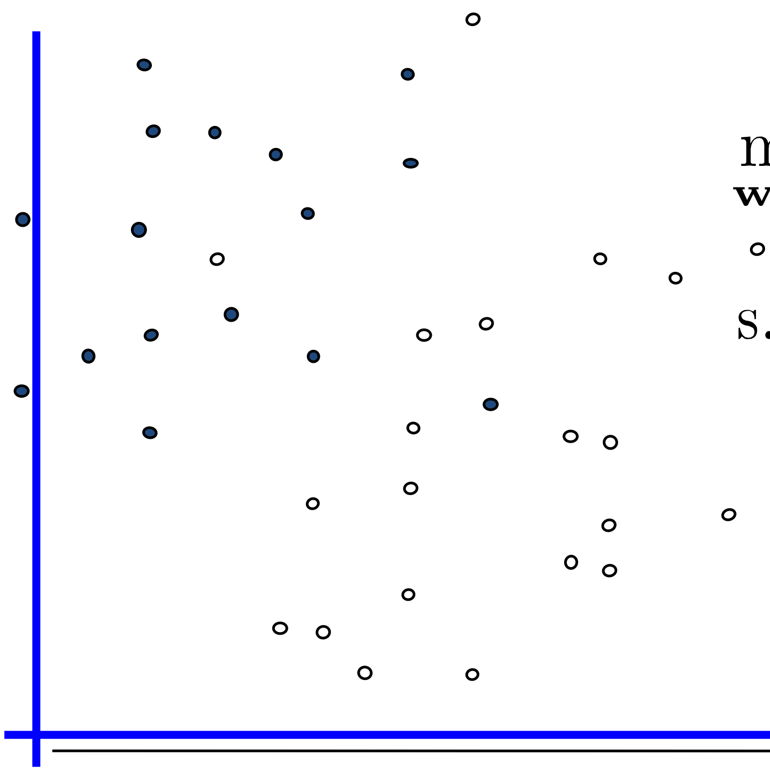
$$\ldots$$

$$y_N(\mathbf{w}^\top \mathbf{x}_N + b) \geq 1 - \varepsilon_N$$

Linearly Inseparable Case

- Relax the constraints
- Penalize the relaxation

denotes +1
denotes -1



$$\min_{\mathbf{w},b,\varepsilon} \quad \frac{1}{2}\sum_{j=1}^{d} w_j^2 + C\sum_{i=1}^{N}\varepsilon_i$$

$$\text{s. t.} \quad y_1(\mathbf{w}^\top \mathbf{x}_1 + b) \geq 1 - \varepsilon_1, \varepsilon_1 \geq 0$$

$$\cdots$$

$$y_N(\mathbf{w}^\top \mathbf{x}_N + b) \geq 1 - \varepsilon_N, \varepsilon_N \geq 0$$

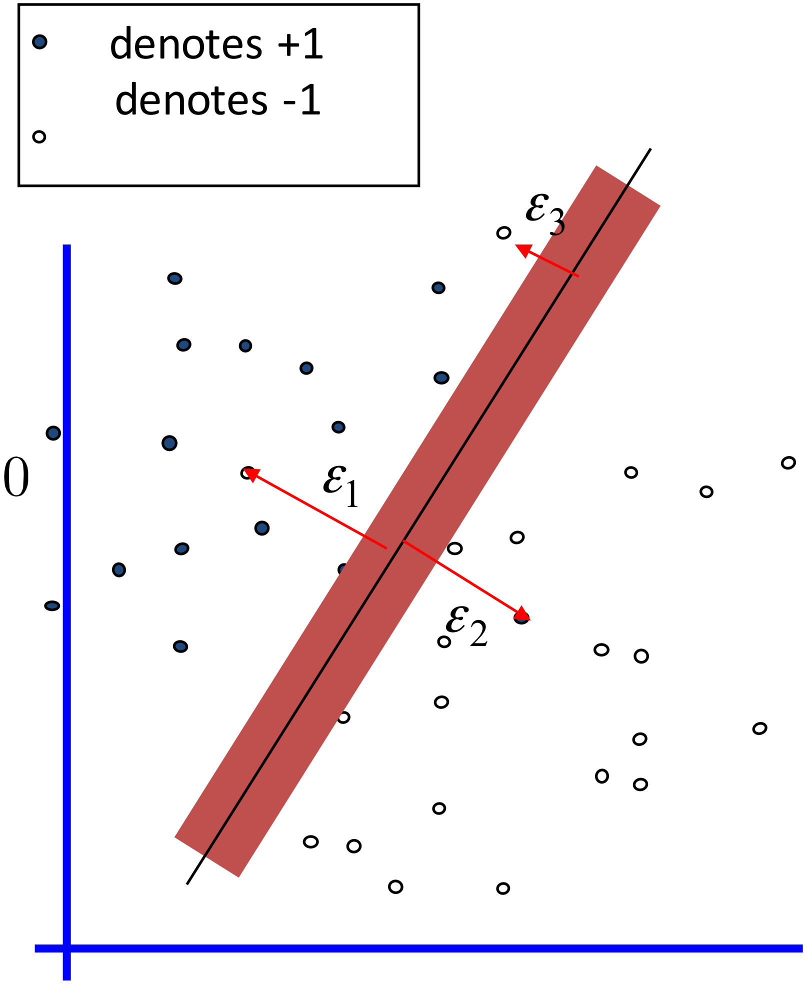
Linearly Inseparable Case

$$\min_{\mathbf{w},b,\varepsilon} \quad \frac{1}{2}\sum_{j=1}^{d} w_j^2 + C\sum_{i=1}^{N}\varepsilon_i$$

$$\text{s. t.} \quad y_1(\mathbf{w}^\top \mathbf{x}_1 + b) \geq 1 - \varepsilon_1, \varepsilon_1 \geq 0$$

$$\cdots$$

$$y_N(\mathbf{w}^\top \mathbf{x}_N + b) \geq 1 - \varepsilon_N, \varepsilon_N \geq 0$$

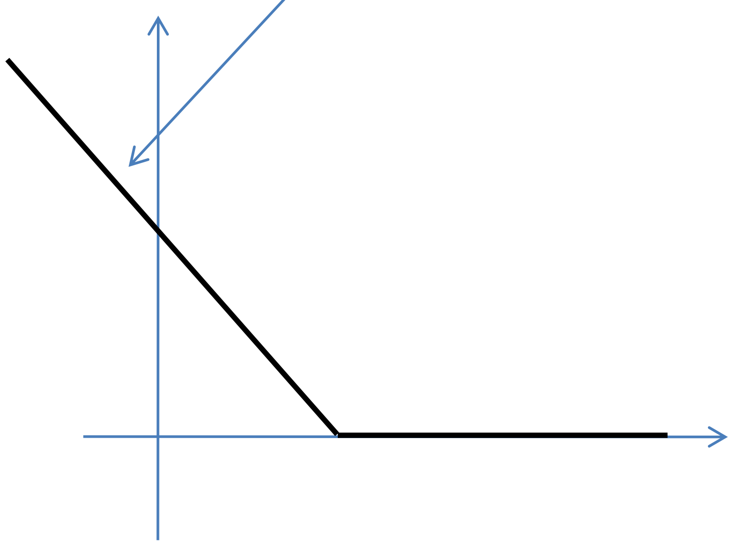


Still a quadratic programming problem

Linearly Inseparable Case

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\sum_{j=1}^{d} w_j^2 + C\sum_{i=1}^{N} \ell(y_i[\mathbf{x}_i^\top \mathbf{w} + b])$$

$$\text{Hinge loss } \ell(z) = \max(0, 1 - z)$$



Linearly Inseparable Case

Support Vector Machine

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\sum_{j=1}^{d} w_j^2 + C\sum_{i=1}^{N}\max(0, 1 - y_i[\mathbf{x}_i^\top \mathbf{w} + b])$$

Regularized logistic regression

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\sum_{j=1}^{d} w_j^2 + C\sum_{i=1}^{N}\ln(1 + \exp(-y_i[\mathbf{x}_i^\top \mathbf{w} + b]))$$

Dual Form of SVM

$$\max_{\alpha_i \in [0,C]} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^\top \mathbf{x}_j) : \sum_{i=1}^{N} \alpha_i y_i = 0 \right\}$$

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

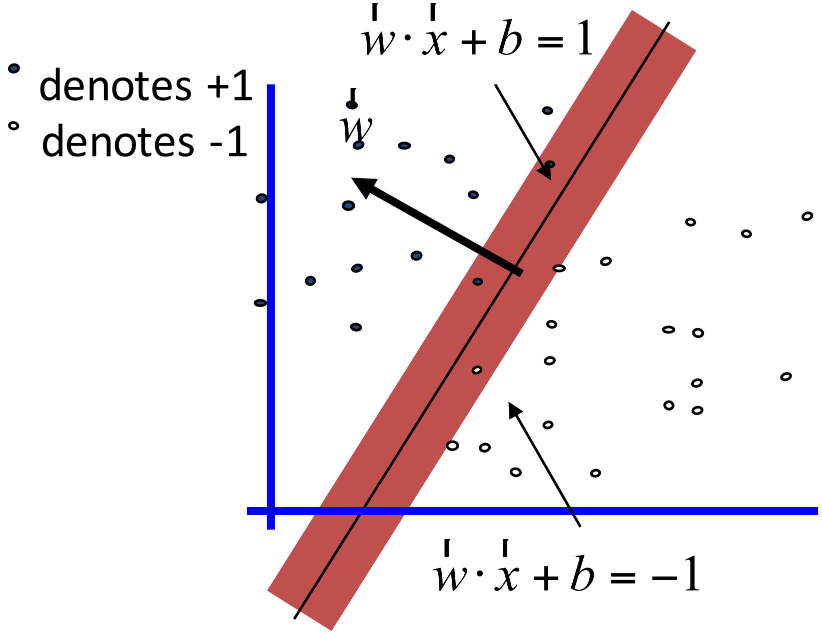
$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \sum_{i=1}^{N} \alpha_i y_i (\mathbf{x}_i^\top \mathbf{x}) + b$$

How to decide b ?

Dual Form of SVM

$$\max_{\alpha_i \in [0,C]} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^\top \mathbf{x}_j) : \sum_{i=1}^{N} \alpha_i y_i = 0 \right\}$$

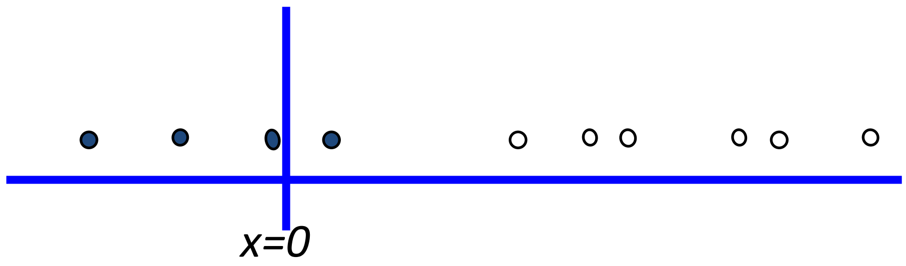
$$\alpha_i \varepsilon_i = 0, i = 1, \ldots, N$$



Support vectors: $\alpha_i > 0$

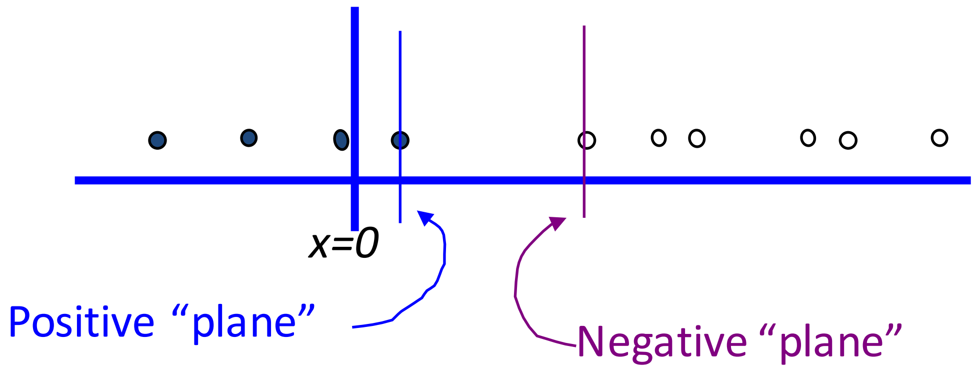
Suppose we're in 1-dimension

What would SVMs do with this data?

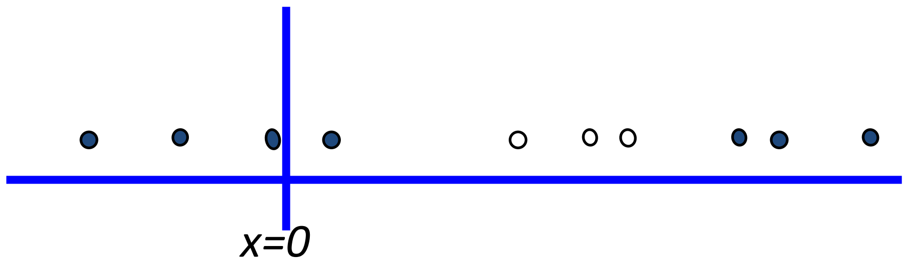


Suppose we're in 1-dimension

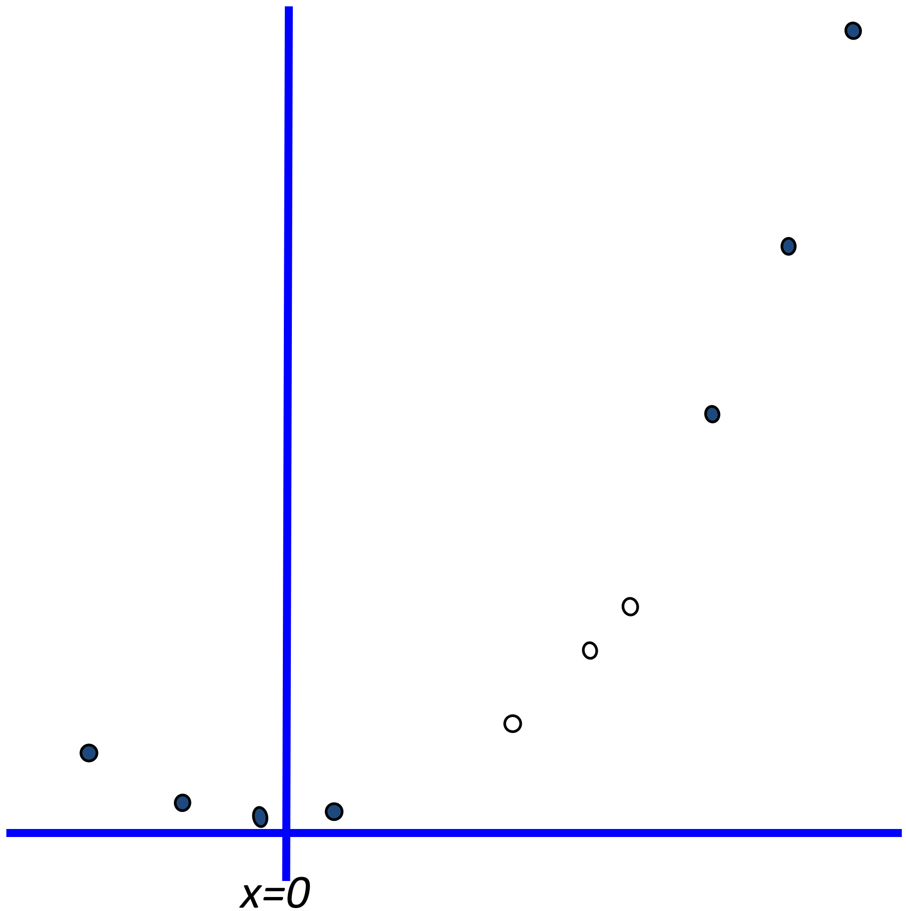
Not a big surprise



Harder 1-dimensional Dataset

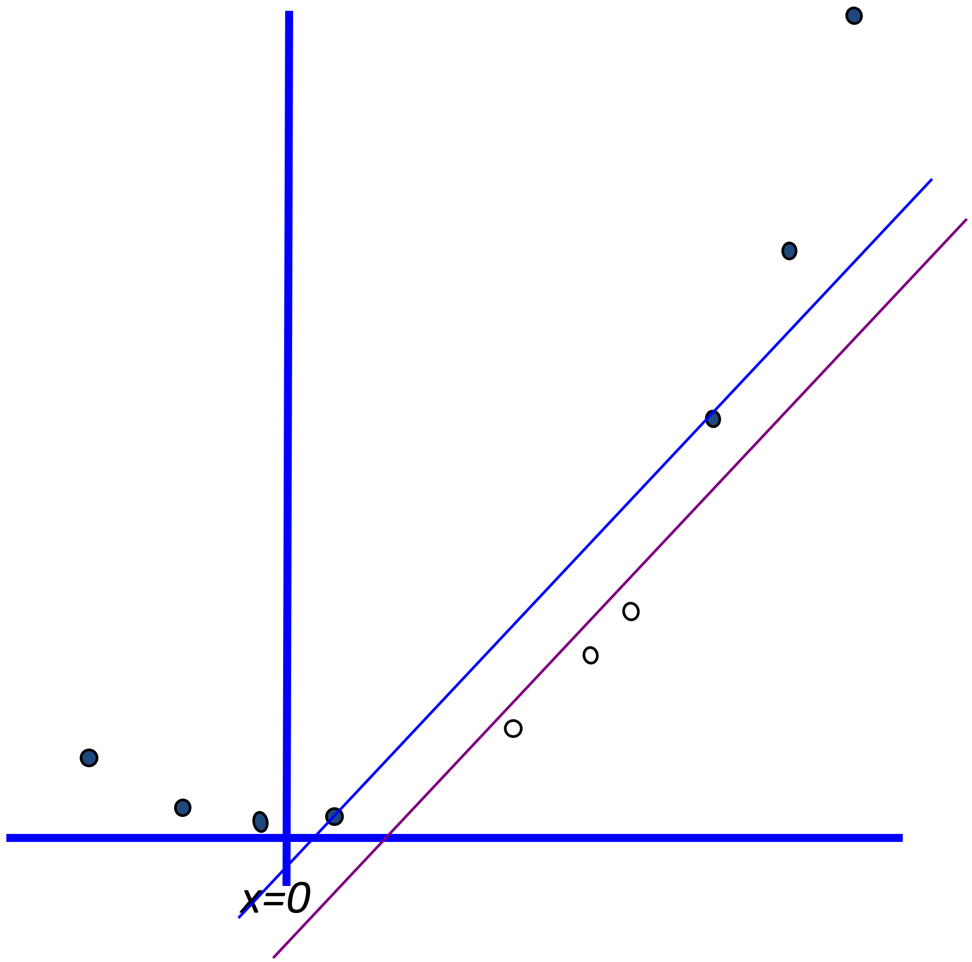


Harder 1-dimensional Dataset



$$x \mapsto (x, x^2)$$

Harder 1-dimensional Dataset



$$x \mapsto (x, x^2)$$

Common SVM Basis Functions

- Polynomial terms of ***x*** of degree 1 to *q*
- Radial (Gaussian) basis functions

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{|\mathbf{x} - \mathbf{c}_j|_2^2}{\sigma^2}\right)$$

Quadratic Basis Functions

$$\Phi(\mathbf{x}) = \begin{pmatrix} 1 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ \vdots \\ \sqrt{2}x_m \\ x_1^2 \\ x_2^2 \\ \vdots \\ x_m^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_1x_3 \\ \vdots \\ \sqrt{2}x_1x_m \\ \sqrt{2}x_2x_3 \\ \vdots \\ \sqrt{2}x_1x_m \\ \vdots \\ \sqrt{2}x_{m-1}x_m \end{pmatrix}$$

Constant Term

Linear Terms

Pure Quadratic Terms

Quadratic Cross-Terms

Number of terms (assuming m input dimensions) = (m+2)-choose-2

= (m+2)(m+1)/2

= $m^2/2$

Dual Form of SVM

$$\max_{\alpha_i \in [0,C]} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^\top \mathbf{x}_j) : \sum_{i=1}^{N} \alpha_i y_i = 0 \right\}$$

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i$$

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \sum_{i=1}^{N} \alpha_i (\mathbf{x}_i^\top \mathbf{x}) + b$$

Dual Form of SVM

$$\max_{\alpha_i \in [0,C]} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j (\vec{\phi}^\top(\mathbf{x}_i)\vec{\phi}(\mathbf{x}_j)) : \sum_{i=1}^{N} \alpha_i y_i = 0 \right\}$$

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \vec{\phi}(\mathbf{x}_i)$$

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \sum_{i=1}^{N} \alpha_i (\vec{\phi}^\top(\mathbf{x}_i)\vec{\phi}(\mathbf{x})) + b$$

It is computationally expensive to calculate $\vec{\phi}^\top(\mathbf{x}_i)\vec{\phi}(\mathbf{x}_j)$

Quadratic Dot Products

$$\Phi(\mathbf{a}) \bullet \Phi(\mathbf{b}) = \begin{pmatrix} 1 \\ \sqrt{2}a_1 \\ \sqrt{2}a_2 \\ : \\ \sqrt{2}a_m \\ a_1^2 \\ a_2^2 \\ : \\ a_m^2 \\ \sqrt{2}a_1a_2 \\ \sqrt{2}a_1a_3 \\ : \\ \sqrt{2}a_1a_m \\ \sqrt{2}a_2a_3 \\ : \\ \sqrt{2}a_1a_m \\ : \\ \sqrt{2}a_{m-1}a_m \end{pmatrix} \bullet \begin{pmatrix} 1 \\ \sqrt{2}b_1 \\ \sqrt{2}b_2 \\ : \\ \sqrt{2}b_m \\ b_1^2 \\ b_2^2 \\ : \\ b_m^2 \\ \sqrt{2}b_1b_2 \\ \sqrt{2}b_1b_3 \\ : \\ \sqrt{2}b_1b_m \\ \sqrt{2}b_2b_3 \\ : \\ \sqrt{2}b_1b_m \\ : \\ \sqrt{2}b_{m-1}b_m \end{pmatrix}$$

$$1$$

$$+$$

$$\sum_{i=1}^{m} 2a_ib_i$$

$$+$$

$$\sum_{i=1}^{m} a_i^2b_i^2$$

$$+$$

$$\sum_{i=1}^{m} \sum_{j=i+1}^{m} 2a_ia_jb_ib_j$$

Quadratic Dot Products

$$\Phi(\mathbf{a}) \bullet \Phi(\mathbf{b}) =$$

$$1 + 2\sum_{i=1}^{m} a_i b_i + \sum_{i=1}^{m} a_i^2 b_i^2 + \sum_{i=1}^{m} \sum_{j=i+1}^{m} 2 a_i a_j b_i b_j$$

Just out of casual, innocent, interest, let's look at another function of ***a*** and ***b***:

$$(\mathbf{a}.\mathbf{b} + 1)^2$$

$$= (\mathbf{a}.\mathbf{b})^2 + 2\mathbf{a}.\mathbf{b} + 1$$

$$= \left(\sum_{i=1}^{m} a_i b_i \right)^2 + 2\sum_{i=1}^{m} a_i b_i + 1$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{m} a_i b_i a_j b_j + 2\sum_{i=1}^{m} a_i b_i + 1$$

$$= \sum_{i=1}^{m} (a_i b_i)^2 + 2\sum_{i=1}^{m} \sum_{j=i+1}^{m} a_i b_i a_j b_j + 2\sum_{i=1}^{m} a_i b_i + 1$$

Kernel Trick

$$\max_{\alpha_i \in [0,C]} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j (\vec{\phi}^\top(\mathbf{x}_i) \vec{\phi}(\mathbf{x}_j)) : \sum_{i=1}^{N} \alpha_i y_i = 0 \right\}$$

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \vec{\phi}(\mathbf{x}_i)$$

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \sum_{i=1}^{N} \alpha_i (\vec{\phi}^\top(\mathbf{x}_i) \vec{\phi}(\mathbf{x})) + b$$

Define a kernel function: $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \vec{\phi}^\top(\mathbf{x}_i) \vec{\phi}(\mathbf{x}_j)$

Kernel Trick

$$\max_{\alpha_i \in [0,C]} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) : \sum_{i=1}^{N} \alpha_i y_i = 0 \right\}$$

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \vec{\phi}(\mathbf{x}_i)$$

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \sum_{i=1}^{N} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) + b$$

Define a kernel function: $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \vec{\phi}^\top(\mathbf{x}_i)\vec{\phi}(\mathbf{x}_j)$

SVM Kernel Functions

- Polynomial kernel function

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j + 1)^q$$

- Radial basis kernel function (universal kernel)

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\lambda |\mathbf{x}_i - \mathbf{x}_j|_2^2\right)$$

Kernel Tricks

- Replacing dot product with a kernel function

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \vec{\phi}^\top(\mathbf{x}_i)\vec{\phi}(\mathbf{x}_j)$$

- Not all functions are kernel functions
- Are they kernel functions ?

$$\kappa(\vec{a}, \vec{b}) = \sum_{i=1}^{d}(a_i - b_i)^3$$

$$\kappa(\vec{a}, \vec{b}) = \sum_{i=1}^{d}(a_i - b_i)^4(a_i + b_i)^2$$

Kernel Tricks

Mercer's condition

To expand Kernel function k(x,y) into a dot product, i.e. $k(x,y)=\Phi(x)\cdot\Phi(y)$, k(x, y) has to be positive semi-definite function, i.e., for any function f(x) whose $\int f^2(x)dx$ is finite, the following inequality holds

$$\int d\mathbf{x}_a d\mathbf{x}_b f(\mathbf{x}_a)\kappa(\mathbf{x}_a, \mathbf{x}_b) f(\mathbf{x}_b) \geq 0$$

Kernel Tricks

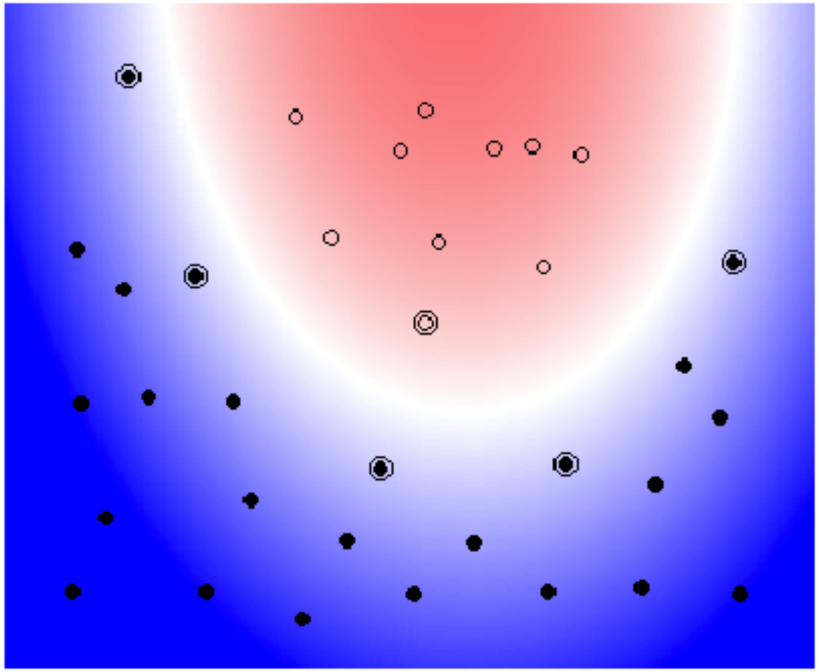
- Introducing nonlinearity into the model
- Computationally efficient

Nonlinear Kernel (I)

Example: SVM with Polynomial of Degree 2

Kernel: $K(\vec{x}_i, \vec{x}_j) = [\vec{x}_i \cdot \vec{x}_j + 1]^2$

plot by Bell SVM applet

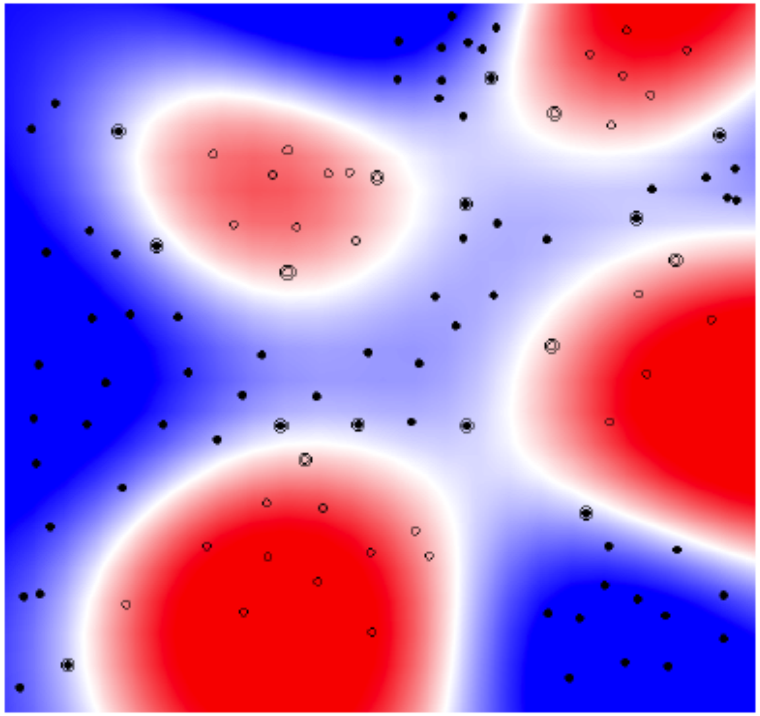


Nonlinear Kernel (II)

Example: SVM with RBF-Kernel

Kernel: $K(\vec{x}_i, \vec{x}_j) = \exp(-|\vec{x}_i - \vec{x}_j|^2 / \sigma^2)$

plot by Bell SVM applet



Reproducing Kernel Hilbert Space (RKHS)

- Reproducing Kernel Hilbert Space *H*
  - Eigen decomposition:

$$\kappa(\mathbf{x}_a, \mathbf{x}_b) = \sum_{i=1}^{\infty} \gamma_i \phi_i(\mathbf{x}_a)\phi_i(\mathbf{x}_b) \qquad \gamma_i \geq 0, \sum_{i=1}^{\infty} \gamma_i^2 < \infty$$

  - Elements of space *H*:

$$f(\mathbf{x}) = \sum_{i=1}^{\infty} c_i \phi_i(\mathbf{x}) \qquad \langle f, f \rangle_{\mathcal{H}} = \sum_{i=1}^{\infty} c_i^2 / \gamma_i < \infty$$

- Reproducing property

$$\langle \kappa(\mathbf{x}, \cdot), f(\cdot) \rangle_{\mathcal{H}} = f(\mathbf{x})$$

Reproducing Kernel Hilbert Space (RKHS)

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\sum_{j=1}^{d} w_j^2 + C\sum_{i=1}^{N} \max(0, 1 - y_i[\mathbf{x}_i^\top \mathbf{w} + b])$$

$$\downarrow$$

$$\min_{f\in\mathcal{H}} \quad \frac{1}{2}\langle f, f\rangle_{\mathcal{H}} + C\sum_{i=1}^{N} \max(0, 1 - y_i f(\mathbf{x}_i))$$

- Representer theorem

Kernelize Logistic Regression

- How can we introduce nonlinearity into the logistic regression model?

$$p(y|\mathbf{x}) = \frac{1}{1+\exp(-y\mathbf{x}^\top \mathbf{w})}$$

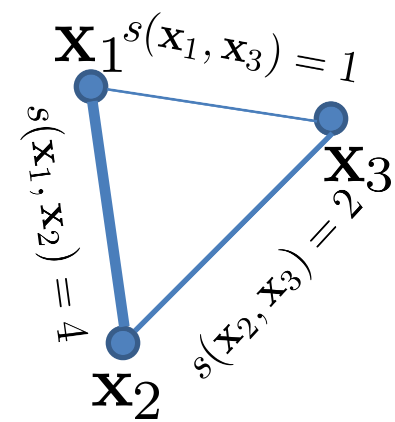
Diffusion Kernel

- Kernel function describes the correlation or similarity between two data points
- Given that a similarity function s(x,y)
  - Non-negative and symmetric
  - Does not obey Mercer's condition
- How can we generate a kernel function based on this similarity function?

A graph theory approach …

Diffusion Kernel

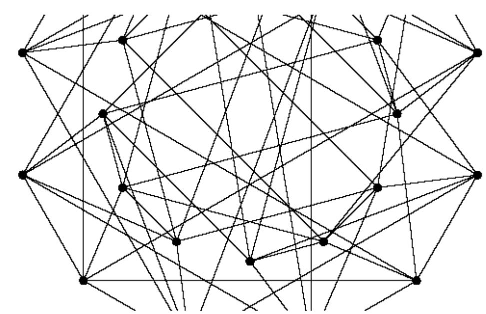
- Create a graph for all the training examples
  - Each vertex corresponds to a data point
  - The weight of each edge is the similarity s(x,y)



- Graph Laplacian

$$L_{i,j} = \begin{cases} s(\mathbf{x}_i, \mathbf{x}_j) & i \neq j \\ -\sum_{k \neq i} s(\mathbf{x}_i, \mathbf{x}_k) & i = j \end{cases}$$

  - Negative semi-definite



Diffusion Kernel

Consider a simple Laplacian

$$L_{i,j} = \begin{cases} 1 & i \neq j \\ -\sum_{k \in \mathcal{N}_i} 1 & i = j \end{cases}$$

Consider L^2, L^3, …

What do these matrices represent?

A diffusion kernel $K_\beta = e^{\beta L} = \lim_{n \to \infty} \left(I + \frac{\beta}{n} L \right)^n$

Diffusion Kernel: Properties

$$K_\beta = \exp(\beta L)$$

- Positive definite
- How to compute the diffusion kernel ?

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.

Kernel Learning

$$\max_{\alpha_i \in [0,C]} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) : \sum_{i=1}^{N} \alpha_i y_i = 0 \right\}$$

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \sum_{i=1}^{N} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}) + b$$

What if we have multiple kernel functions

$$\kappa_1(\mathbf{x}, \mathbf{x}), \kappa_2(\mathbf{x}, \mathbf{x}), \ldots, \kappa_m(\mathbf{x}, \mathbf{x})$$

- Which one is the best ?
- How can we combine multiple kernels ?

Kernel Learning

$$\kappa(\mathbf{x}, \mathbf{x}; \gamma) = \sum_{k=1}^{m} \gamma_k \kappa_k(\mathbf{x}, \mathbf{x})$$

$$\gamma = (\gamma_1, \ldots, \gamma_m) \in \Delta = \left\{ \gamma \in \mathbb{R}_+^m : \sum_{k=1}^{m} \gamma_k = 1 \right\}$$

$$\min_{\gamma \in \Delta} \max_{\alpha_i \in [0, C]} \left\{ \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j; \gamma) : \sum_{i=1}^{N} \alpha_i y_i = 0 \right\}$$

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b = \textstyle\sum_{i=1}^{N} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}; \gamma) + b$$

References

- An excellent tutorial on VC-dimension and Support Vector Machines:

  C.J.C. Burges. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 2(2):955-974, 1998. http://citeseer.nj.nec.com/burges98tutorial.html

- The VC/SRM/SVM Bible: (Not for beginners including myself)

  Statistical Learning Theory by Vladimir Vapnik, Wiley-Interscience; 1998

- Software: SVM-light, http://svmlight.joachims.org/, free download